Politecnico di Torino

DEPARTMENT OF MECHANICAL AND AEROSPACE ENGINEERING
Master of Science in Aerospace Engineering
Academic Year 2021/2022

Application of Optimal Control Techniques to the Parafoil Flight of Space Rider

Terminal Guidance Phase Optimal Control

In collaboration with:

SENER Aeroespacial

Author:
Michele Lucrezia

Supervisor:
Prof. Elisa Capello
Politecnico di Torino

Mentor:
Francesco Cacciatore
SENER Aeroespacial

THIS PROJECT HAS BEEN DEVELOPED UNDER THE SUPERVISION OF SENER AEROESPACIAL. THE CONTENTS OF THIS WORK DO NOT REPRESENT THE ALGORITHMS DEVELOPED BY THE TEAM OF SENER AEROESPACIAL FOR THE GNC SUBSYSTEM OF THE SPACE RIDER MISSION. THOSE ILLUSTRATED ARE INSTEAD POSSIBLE ALTERNATIVE SOLUTIONS THAT COULD BE APPLIED TO THE SPACE RIDER CASE. ANY REFERENCE TO PARAFOIL GNC ALGORITHMS DEVELOPED BY SENER AEROESPACIAL WILL BE REFERRED TO AS "SR-PGNC". THE TERM "PGNC" WILL BE USED FOR THE ALGORITHMS DEVELOPED IN THIS WORK.

A Caterina ed Antonio

Abstract

The design of reusable aerospace vehicles is a challenge that the aerospace industry has been pursuing for several years in order to reduce the cost of access to space and the environmental impact also in terms of orbital debris. The Space Rider program of the European Space Agency (ESA) falls within the framework of such initiatives. Among the greatest challenges for this type of mission is the design of the Guidance, Navigation & Control (GNC) subsystem for the atmospheric re-entry phase. This subsystem is responsible for collecting and processing data from onboard sensors so that the state of the spacecraft can be used to generate a feasible trajectory in compliance with mission requirements. The requirements for this kind of mission are generally expressed in terms of landing accuracy and touchdown velocity with further possible constraints related, for instance, to the presence of no-fly zones. The final descent and landing phase of the Space Rider mission consists of an autonomous flight under parafoil that must guarantee a smooth and precise landing. The GNC subsystem plays a key role in ensuring adequate performance for a safe landing that does not result in damage to the vehicle and allows it to be reused for subsequent launches. This task is quite challenging due to the limited control authority provided by the parafoil and the high sensitivity to environmental conditions. In particular, a medium-large parafoil is characterized by an airspeed of 10-20 m/s, which may be of the same order of magnitude as the windspeed generating high uncertainty in trajectory generation and tracking. In addition, no propeller or thrusters are available for flight under parafoil, so the actuation is based exclusively on the asymmetric and symmetric air-brake deflection used for lateral and vertical control respectively. A further major limitation for trajectory generation is the very slow dynamics of the parafoil in the latero-directional plane, which results in a limited maximum turning rate that for large parafoils can be in the order of 5-10 deg/s. Considering the numerous constraints that characterize parafoil re-entry, one of the most critical stages is what is commonly referred to as the Terminal Guidance (TERGUID) phase. This is the final part of the descent where

the vehicle performs the final approach to the designated landing point (LP) trying to counterbalance the unknown effect of the wind.

The study presented in this thesis was developed at the AOCS/GNC department of SENER Aeroespacial (Madrid, Spain) and the objective is to design a complete solution for managing the Terminal Guidance phase of a Space Rider type case. This includes a guidance algorithm to generate an optimal solution for the TERGUID trajectory, a path tracking procedure, and a guidance logic that allows for a correct implementation within the whole GNC software. For trajectory generation, the approach presented in [1] was selected. The authors apply a direct method based on inverse dynamics in the virtual domain to select the optimal trajectory given a specific two-point boundary-value problem (TPBVP). This study proposes several modifications to the aforementioned method to adapt the algorithm to the specific case of Space Rider. The efficiency of such an algorithm allows frequent recalculation of the optimal trajectory guaranteeing the mitigation of the effects of the unknown wind. In addition, several possible approaches for trajectory tracking have been investigated, including a simple PID controller and a more accurate Model Predictive Control. The optimal terminal guidance algorithm has then been implemented within the 6DOF simulator developed by SENER Aeroespacial to verify its performance in terms of landing accuracy through a series of Monte Carlo simulations. For this purpose, it became necessary to plan a precise guidance strategy and design an ad hoc logic to manage the terminal guidance phase and the associated submode transitions.

The results obtained demonstrated an excellent functioning of the guidance algorithm for the proposed problem especially when supported by a robust logic that takes into account various potential scenarios in terms of boundary conditions and environmental disturbances.

Keywords:

Parafoil, GNC, Guidance, Control, Space Rider, Re-entry, MATLAB, Simulink, Model Predictive Control, Linear Quadratic Regulator, Direct Methods, Terminal Guidance, Trajectory Optimization, Modelling, 6DOF, Linearization, Simulator.

Acknowledgments

Il lavoro presentato in questa tesi è stato sviluppato in collaborazione con SENER Aeroespacial. Per prima cosa vorrei quindi ringraziare il team di Space Rider e tutti gli altri miei colleghi, persone con cui sono felice di condividere questi primi passi della mia carriera professionale nel mondo dell'ingegneria aerospaziale. Un ringraziamento particolare va al mio tutor aziendale Francesco Cacciatore ed al mio compagno di disavventure Jorge Cardin, due persone il cui supporto è risultato davvero prezioso in questi ultimi mesi passati a Madrid.
Colgo poi l'occasione per esprimere la mia sincera gratitudine verso la Prof. Elisa Capello che, nel ruolo di relatrice, ha saputo garantirmi sempre supporto ed aiuto, nonostante la distanza.
Un ringraziamento speciale va poi a tutte quelle persone che in tutti questi anni, nonostante il mio vagabondare, sono rimaste presenti nella mia vita. Tra queste, sicuramente, tutti i parenti (acquisiti e non) la cui presenza sono convinto non vada mai data per scontata. Oltre a loro ci sono ex compagni di squadra, di università, di scuola, di casa e non solo che ora definirei semplicemente amici. Cambiare spesso città in generale allontana le persone ma, in alcuni preziosi casi, allo stesso tempo le avvicina. E più mi perdo, più mi rendo conto di quanto valore abbia tutto ciò.
Voglio ringraziare poi una persona il cui supporto è stato per me vitale in questi ultimi anni. Clara, con la sua infinita pazienza (di cui dichiaro qui ufficialmente di essere consapevole) è stata un pilastro davvero importante nel tortuoso cammino che mi ha portato fino a qui.
Non posso non ringraziare inoltre Federica che in una calda giornata estiva, con quel per me indimenticabile "perché non riinizi l'università", mi ha dato l'input di cui avevo bisogno, come solo una sorella sa fare.
Infine, la mia più profonda gratitudine va ai miei genitori che mi hanno sempre appoggiato e sostenuto e che costituiscono per me un importante punto di riferimento ed esempio.

Madrid, July 2022

Contents

List of Figures

List of Tables

List of Acronyms

AF	Air-path Frame
AI	Apparent Inertia
ALEK	AVUM Life Extension Kit
AM	Apparent Mass
AOM	AVUM Orbital Module
AVUM	Attitude Vernier Upper Module
BF	Body-fixed Frame
CAM	Collision Avoidance Maneuvers
CF	Canopy Frame
CFTOC	Constrained Finite-Time Optimal Control
CoAM	Center of Apparent Mass
CoM	Center of Mass
D&L	Descent and Landing
DCM	Direct Cosine Matrix
DKE	Dynamics, Kinematics, and Environment
DOF	Degrees of Freedom
DSF	Design Simulation Facility
ENEMNG	Energy Management
ESA	European Space Agency
FPA	Flight Path Angle
GNC	Guidance, Navigation & Control
GPS	Global Positioning System
HDG	Heading
HDR	Heading Rate
HW	Hardware
IMU	Inertial Measurement Unit
IXV	Intermediate eXperimental Vehicle
LP	Landing Point
LPF	Landing Point Frame
LUT	LookUp Table
LV	Local Vertical
MC	Monte Carlo
MPC	Model Predictive Control
MPCB	Multi-Purpose Cargo Bay
NED	North-East-Down

NF	Navigational Frame
NFZ	No-Fly Zone
ODE	Ordinary Differential Equations
PADS	Precision Aerial Delivery System
ParCoM	Parafoil Center of Mass
PayCoM	Payload Center of Mass
PCON	Parafoil Control
PGNC	Parafoil Guidance, Navigation & Control
PGUI	Parafoil Guidance
PID	Proportional–Integral–Derivative
PMNG	Parafoil submode Manager
PNAV	Parafoil Navigation
RADALT	Radar Altimeter
RM	Re-entry Module
SISO	Single Input Single Output
SPO	Single Parameter Optimization
SR	Space Rider
SW	Software
SysCoM	System Center of Mass
TAEM	Terminal Area Energy Management
TBD	To Be Defined
TD	TouchDown
TERGUID	Terminal Guidance
TPBVP	Two-Point Boundary-Value Problem
TPO	Two-Parameter Optimization
WF	Wind-fixed Frame
WP	Waypoint

Nomenclature

$\boldsymbol{R}_{ji}$	rotation matrix from frame i to frame j
$\mathrm{s}_\theta, \mathrm{c}_\theta$	sine of θ, cosine of θ
ψ, θ, ϕ	yaw angle, pitch angle, roll angle
α, β	angle of attack, sideslip angle
χ, γ, ϕ	heading angle, flight path angle, bank angle
χ_a, γ_a, ϕ_a	heading angle, flight path angle, bank angle (air)
δ_a, δ_s	dimensional asymmetric, symmetric deflection
$\bar{\delta}_a, \bar{\delta}_s$	non-dimensional asymmetric, symmetric deflection
δ_r, δ_l	dimensional right, left deflection
$\boldsymbol{X} = [x, y, z]^T$	position vector in LV
$\boldsymbol{X}_a = [x_a, y_a, z_a]^T$	air position vector in WF
$\boldsymbol{V}_a = [V_{xa}, V_{ya}, V_{za}]^T$	airspeed vector in LV
$\boldsymbol{v}_a = [\ u_a, v_a, w_a]^T$	airspeed vector in BF
$\boldsymbol{V} = [V_x, V_y, V_z]^T$	groundspeed vector in LV
$\boldsymbol{v} = [u, v, w]^T$	groundspeed vector in BF
$\hat{\boldsymbol{v}}_{ac} = [\hat{u}_{ac}, \hat{v}_{ac}, \hat{w}_{ac}]^T$	canopy airspeed vector in CF
$Q = \rho V_a^2/2$	dynamic pressure
$\boldsymbol{V}_{ha}, \boldsymbol{V}_h$	horizontal airspeed, horizontal groundspeed in LV
$\boldsymbol{V}_{va}, \boldsymbol{V}_v$	vertical airspeed, vertical groundspeed in LV
$\boldsymbol{V}_{va0}, \boldsymbol{V}_{ha0}$	vertical, horizontal airspeed steady-state values in LV
$\boldsymbol{W} = [W_x, W_y, W_z]^T$	wind speed vector in LV
$\boldsymbol{w} = [\ w_x, w_y, w_z]^T$	wind speed vector in BF
$\boldsymbol{\omega} = [p, q, r]^T$	angular rate vector of BF wrt LV in BF
$\tilde{\boldsymbol{\omega}} = \left[\dot{\phi}, \dot{\theta}, \dot{\psi}\right]^T$	angular rate vector of BF wrt LV in LV
$\hat{\boldsymbol{\omega}}_c = [\hat{p}_c, \hat{q}_c, \hat{r}_c]^T$	canopy angular rate vector in CF
$\boldsymbol{\Phi}$	rotation matrix from $\boldsymbol{\omega}$ to $\tilde{\boldsymbol{\omega}}$
$\boldsymbol{F}^{LV}, \boldsymbol{M}^{LV}$	external force, external momentum in LV
$\boldsymbol{F}^{C}, \boldsymbol{M}^{C}$	external force, external momentum in CF
$\boldsymbol{F}, \boldsymbol{M}$	external force, external momentum in BF
L, D	lift, drag in AF
$\boldsymbol{0}_{nxn}$	null matrix with size nxn
$\boldsymbol{I}_{nxn}$	identity matrix with size nxn
$\boldsymbol{I}_t$	inertia tensor
$\boldsymbol{I}_{xx},\ \boldsymbol{I}_{yy},\ \boldsymbol{I}_{zz}$	moments of inertia
$\boldsymbol{I}_{xy},\ \boldsymbol{I}_{yz},\ \boldsymbol{I}_{xz}$	products of inertia
$\boldsymbol{S}(\boldsymbol{x})$	skew-symmetric matrix applied to $\boldsymbol{x}$
C_{ij}	aerodynamic coefficients (see [2])
S_p	parafoil canopy area

$c, \bar{c}$	chord, mean aerodynamic chord of the canopy
b, t	span, thickness of the canopy
ρ	air density
m	system mass
g	universal gravitational constant
M_r	mass ratio
s_0	linearization steady-state vector
$T_\psi, K_\psi, T_\phi, K_\phi, K_{\dot{\psi}}$	weighting coefficients for ψ, ϕ dynamic models
T_m	target maneuver duration
τ	virtual variable
$\bar{\tau} = \tau/\tau_f$	normalized virtual variable
a_m^η, b_n^η	direct method trajectory coefficients
$\lambda = d\tau/dt$	derivative of τ with respect to time t
$x' = dx/d\tau$	derivative in the virtual domain
$J, (J^*)$	(optimal) cost function
$\varDelta\dot{\psi}_J$	yaw rate penalty term
$\varDelta\psi_{Ji}, \varDelta\psi_{Jf}$	initial/final yaw penalty terms
$\varDelta N_J$	no-fly zones penalty terms
$k_{\dot{\psi}}, k_\psi, k_N$	weighting coefficients for penalty terms
TR	time ratio
MT	missing ratio
$\dot{\psi}_{lim1}, \dot{\psi}_{lim2}$	yaw rate threshold values
A, B, C, D	continuous-time state-space matrices
A_d, B_d, C_d, D_d	discrete-time state-space matrices
$\boldsymbol{x}_k$	state vector at k-th sampling time
y_k	single output at k-th sampling
$u_k, (u_k^*)$	(optimal) single control input at k-th sampling
$\boldsymbol{x}_k$	state vector at k-th sampling
H_p	prediction horizon
$\boldsymbol{U}, (\boldsymbol{U}^*)$	(optimal) control input sequence vector
$\boldsymbol{Y^d}, \tilde{\boldsymbol{Y}}$	desired, estimated output sequence vector
$\widetilde{\boldsymbol{X}} = [\tilde{\boldsymbol{x}}_{k+1}, \dots, \tilde{\boldsymbol{x}}_{k+H_p}]$	estimated state vector sequence matrix
$\mathcal{A}, \mathcal{B}, \bar{\mathcal{A}}, \bar{\mathcal{B}}, \mathcal{F}$	LQR matrices
$T_{vertical}, T_{horizontal}$	vertical, horizontal time to the final point
$d_{vertical}, d_{horizontal}$	vertical, horizontal distance from the final point in WF
$h_{ThrsE2T}, h_{ThrsT2FC}$	ENEMNG, TERGUID exit altitude thresholds
h_{LimE2T}	TERGUID forced triggering altitude threshold
$\Delta HDG_{ThrsT2FC}$	TERGUID exit delta heading thresho

1. Introduction

The purpose of this work is the analysis of possible solutions applicable to the design of the Guidance, Navigation, and Control (GNC) subsystem for the final phases of an autonomous flight under parafoil. The framework considered for this study is the one of the Space Rider mission conducted by the European Space Agency (ESA). In particular, this study is aimed at developing an effective guidance strategy for managing the delicate phase of the approach to the designated landing point, also referred to as terminal guidance (TERGUID). Such a strategy involves the design of a guidance algorithm capable of determining the optimal trajectory online, a specific control technique for trajectory tracking, and a dedicated logic for implementing these elements within the complete GNC software. Several challenges characterize this type of problem. Among them are certainly the strong influence of atmospheric conditions in terms of wind, the low control authority provided by parafoil systems, and the stringent requirements for landing accuracy.

This research was carried out at the department of AOCS/GNC of SENER Aeroespacial, Madrid, Spain. Throughout the entire period in which the present study was conducted, the author worked in the aforementioned department as an AOCS/GNC Engineer. Although SENER Aeroespacial has provided technical support and the opportunity to use the simulator developed in-house for the Space Rider project, the work described in this thesis is to be considered solely as a student project with no commercial purpose and thus falls within the legal framework of student activities. All information provided in this dissertation is not related to the algorithms developed by SENER Aeroespacial for the Space Rider program, but rather possible alternative solutions applicable to that mission. Any reference to SENER Aeroespacial proprietary algorithms for the Space Rider flight under parafoils will be specifically referred to by the abbreviation SR-PGNC. Otherwise, the generic term PGNC will be used.

1.1. Thesis Overview

The thesis is divided into the following chapters:

- Chapter 1:

  A brief introduction including scope of work, description of Space Rider mission, and role of SENER Aeroespacial in order to provide an overview of the working background.

- Chapter 2:

  Description of the mission sequence for the Space Rider re-entry mission with a focus on the GNC subsystem and the hardware available to that subsystem.

- Chapter 3:

  Derivation of parafoil-payload system models with different DOF used for the development of PGNC algorithms. Model analytical and numerical linearization for control purposes.

- Chapter 4:

  Description of the guidance problem at hand (TPBVP) and the algorithm selected for optimal trajectory generation with a focus on the modifications applied to adapt it to the Space Rider case.

- Chapter 5:

  Overview of possible approaches for trajectory tracking, including PID, Model Predictive Control (MPC), and Linear Quadratic Regulator (LQR).

- Chapter 6:

  Description of the strategy and the logic implemented for the terminal guidance phase to introduce the algorithms developed in the previous chapters within the complete GNC software.

- Chapter 7:

  Main results of the Monte Carlo simulations carried out with the 6DOF simulator developed by SENER Aeroespacial.

- Chapter 8:

  Conclusions on the current state of the work, challenges faced, and suggestions for further investigation and improvement.

1.2. Space Rider Mission

The Space Reusable Integrated Demonstrator for Europe Return (Space Rider) program was approved in December 2016 by the European Space Agency (ESA) as a follow-up to the Intermediate eXperimental Vehicle (IXV) mission that had flown in February 2015. Space Rider is an uncrewed spacecraft that "aims to provide Europe with an affordable, independent, reusable end-to-end integrated transportation system for routine access and return from low orbit. It will transport payloads for an array of applications, orbit altitudes, and inclinations" [3]. The objective of the program is to launch a vehicle capable of remaining in low Earth orbit (approximately 500 Km) for more than two months in order to conduct scientific experiments in microgravity and with radiation exposure (free flyer), educational activities, or in-orbit tests for demonstration and validation of new technologies. Thus, the purpose of the mission is also to provide access to space to entities and companies not directly involved in the aerospace sector. This is made possible by the presence of a Multi-Purpose Cargo Bay (MPCB) with a volume of 1200 liters capable of carrying up to 800 kg providing a power service for up to 600 Watt along with thermal, data-handling, and telemetry capabilities [4]. The spacecraft will be launched from ESA's Spaceport in Kourou, French Guiana, and will be able to perform a controlled re-entry with a ground landing on European soil. The design of the entire vehicle is focused on the ability to perform smooth reentry that allows the spacecraft to be reused for multiple flights. The re-entry phase is particularly delicate not only because of the challenges related to thermal protection during atmospheric flight in hypersonic and supersonic regimes but also because of the performance required for subsonic flight, especially related to safe landing and reusability of the spacecraft. The Space Rider project, however, can rely on knowledge gained from the IXV program, from which it inherits the design of a good number of subsystems. An innovation compared to the IXV, is the final flight under parafoil, the design of which was tackled from scratch and is a key part of the re-entry phase. Space Rider was developed to be fully integrated with the European launcher Vega-C, and the first flight is scheduled for 2023.

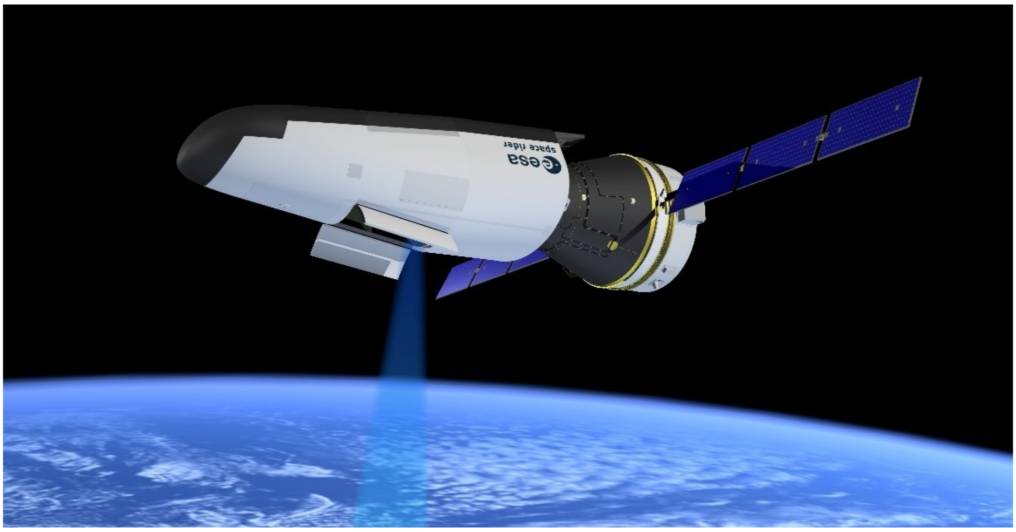

Figure 1-1: Space Rider rendering CREDIT: ESA-Jacky Huart *[5]*

This Space Rider spacecraft (shown in Figure 1-1) is composed of two different modules:

- AVUM Orbital Module (AOM)
- Re-entry Module (RM)

The service module AOM of Space Rider consists of a modified version of the Vega-C Attitude Vernier Upper Module Plus (AVUM+) with the addition of the AVUM Life Extension Kit (ALEK). AVUM+ was designed by the Italian company Avio to place the launcher payload into orbit by providing propulsion and attitude control capabilities. Likewise, the AOM performs the attitude control and all the orbital maneuvers required until the separation of the two modules during the re-entry phase. The service module also ensures electrical power supply to the SR spacecraft for the entire duration of the orbital mission using a two wings deployable solar array as the energy source. Finally, the AOM ensures standard data link and thermal control services [6]. The re-entry module RM is a lifting body based on the design of IXV and is equipped with the Multi-Purpose Cargo Bay which contains the mission payload. This is the reusable part of the spacecraft that during the reentry phase separates from the AOM to perform a controlled atmospheric flight culminating in the precision landing guaranteed by a parafoil. On the other hand, the AOM performs a safe destructive re-entry in open ocean waters.

A possible mission sequence for Space Rider is presented in Figure 1-2. It shows an initial launch phase with Vega-C and ascent to the desired LEO orbit at an altitude of about 400 km. After the commissioning phase, which is completed thanks to AOM support, the core stage of the mission begins, the orbital phase. The first part of that phase (about 4 weeks) is the Microgravity Operational Mode in which the spacecraft is characterized by free drift motion in which attitude is not kept constant and the control effort is minimized [6]. This is followed by the Space Observation and Earth Observation phases, each lasting about two weeks. For the former, the typical attitude mode is Bay to Zenith while in the latter it is Bay to Nadir. In any case, for all operational modes listed above, the pointing to the sun of the solar arrays is ensured to guarantee sufficient energy supply. Next, the de-orbiting phase begins. The AOM provides a final boost to the RM before the separation of the two modules. Subsequently, the AOM begins the Collision Avoidance Maneuvers (CAM) phase, which allows safe reentry into oceanic waters. On the other hand, the RM begins the coasting phase which is followed by the descent & landing phase. For more details on the re-entry phase, please refer to Section 2.1

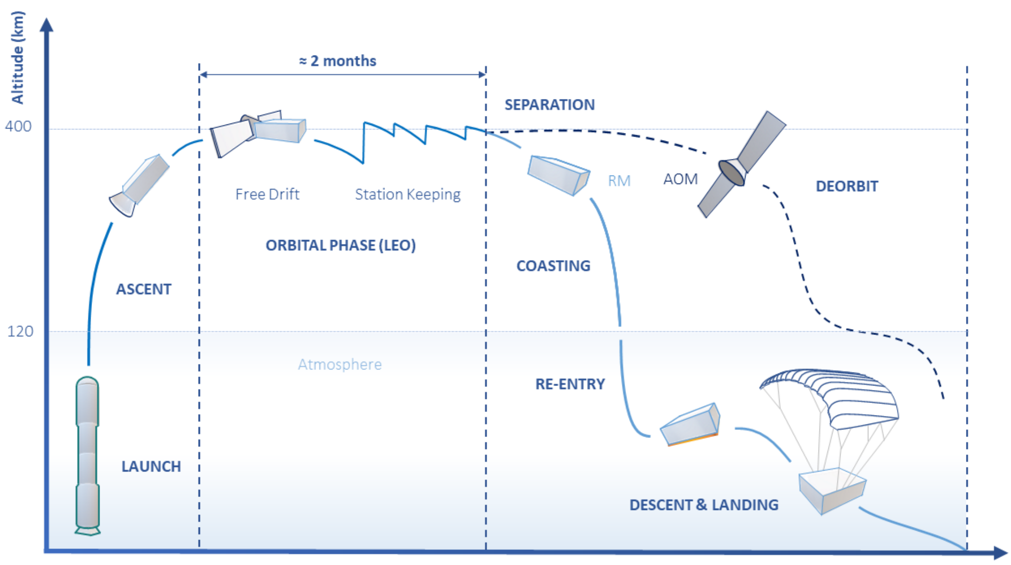

Figure 1-2: Space Rider typical mission profile

1.3. SENER Aeroespacial

On December 9, 2020, "Thales Alenia Space (Thales 67%, Leonardo 33%), and AVIO as co-contractor, signed a contract with the European Space Agency (ESA) for the development of the automated reusable Space Rider transportation system, designed for deployment by the new Vega C light launcher into low Earth orbit (LEO). The total worth of the contract is 167M€" [7]. In May 2021, Thales Alenia Space, as the design responsible for the IXV-derived Re-entry Module, signed a contract with SENER Aeroespacial for the design of the GNC subsystem of the RM. Elecnor-Deimos and Deimos-Space Romania operate as subcontractors for this project. SENER Aeroespacial, as design authority for the full GNC of the Space Rider Re-entry Module, is in charge of coordinating all the activities of the entities participating in the GNC consortium in addition to the actual development of the software based on the heritage of IXV. Among others, SENER Aeroespacial is responsible for developing the GNC algorithms for the flight under parafoil (SR-PGNC) that was not present for the IXV mission. For this purpose, the company leverages a complex infrastructure to perform simulations and test campaigns. This tool has been developed in the MATLAB/Simulink environment and is named Design Simulation Facility (DSF). The DSF ensures quick setup of different types of simulations with different scenarios (including, for instance, Monte Carlo simulations) through a user-friendly interface. The simulator includes both a 3DOF and a 6DOF model. The latter is the simulator used within the scope of this work and henceforth we will refer to it simply as DSF. The tool also contains a set of built-in post-processing functions that enable comprehensive analysis of the results obtained from simulations.

2. Space Rider Re-entry Phase

This chapter describes a possible mission profile for the reentry phase of a Space Rider-type spacecraft. This operational sequence is not intended to represent the exact one planned for the Space Rider mission but is intended to provide an overview of the main re-entry phases useful for contextualizing the present work. An overall picture of the type of hardware and software architecture generally used for this type of mission is also provided.

2.1. Re-entry Mission Profile

This section describes the typical mission sequence for a mission such as Space Rider with a special focus on the flight phase under parafoil, which is the one being examined in this work. Figure 1-2 shows the aforementioned sequence. The re-entry phase begins with a de-orbiting boost performed on the RM by the AOM just before separation. This maneuver allows the RM to insert itself into orbit with a specific perigee that ensures that the desired landing site is reached. The RM remains passive until Coasting begins. The objective of this phase is a controlled loss of altitude by maintaining a given attitude. Coasting must ensure the achievement of a specific final condition in terms of position, velocity, and attitude that coincides with the initial condition designated for the Entry phase. In the latter SR goes through a critical phase from the aerothermodynamic point of view. Around 120-100 km the atmosphere begins to become denser and the spacecraft, which is initially traveling at hypersonic speed, is slowed by its impact with it. This phenomenon generates shock waves and a plasma layer around the aircraft that prevents electromagnetic wave transmission. This proves to be critical from the point of view of the GNC subsystem because there is a time window (generally indicated between 100 and 50 km of altitude) in which there is a blackout of the Global Positioning System (GPS) signal and telemetry data transmission. Once the supersonic low regime is reached (Mach equal to about 1.6), Terminal Area Energy Management (TAEM) phase begins. Here SR follows a specific trajectory designed to transform potential energy into kinetic energy going from supersonic to subsonic regime. When Mach reaches a certain threshold (usually around 0.7), the Descent and Landing (D&L) phase is triggered. This can be split into two different main stages:

- Drogue parachute deceleration
- Parafoil flight

The first part of the D&L is passive. A circular drogue parachute further slows down the spacecraft in order to avoid unwanted aerodynamic effects when opening the parafoil. Parafoil deployment coincides with the beginning of the final active phase of the mission. The flight under parafoil is certainly a particularly critical phase of the re-entry mission as it is subject to stringent requirements in terms of safety and accuracy. The entire design of the Space Rider program is based on the reusability of the RM thus, non-compliance with the landing requirements may jeopardize the entire program. The challenges associated with flying under parafoil are mainly related to the strong influence of the effect of disturbances and uncertainties (such as wind and sensor errors). In addition, the dynamics of large parafoils are in general quite slow and the control authority provided by them is rather low. Moreover, there are strict requirements for accuracy at landing both in terms of position precision and velocity components at touchdown. The latter are essential to reduce the risk of damage caused by ground impact.

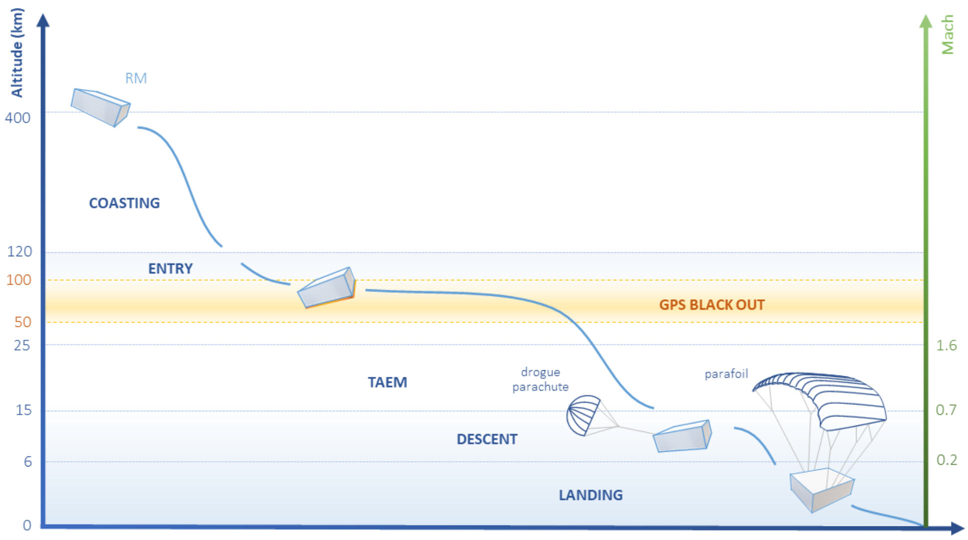

Figure 2-1: Space Rider possible re-entry mission profile

The main stages of the parafoil descent, which is the one examined in this thesis, are now outlined. The phases described here do not represent those adopted by SENER Aeroespacial for the Space Rider mission but are those typically employed for flights of this nature. The main purpose here is to place the Terminal Guidance phase, i.e., the subject of this study, within a realistic mission sequence so as to understand what its purpose is and the boundary conditions to be considered. In Figure 2-2 the main stages of the flight are shown, starting from the deployment of the parafoil to the touchdown instant near the landing point. The reference system used in the figure is Local Vertical (LV) centered at the Landing Point (LP) for which, as described in detail in Section 3.3.1, the x, y, and z axes point north, east, and down, respectively. Henceforth, the different phases of the parafoil flight will also be referred to as "submodes" since each of them is usually associated with a specific submode of the Parafoil GNC (PGNC) system.

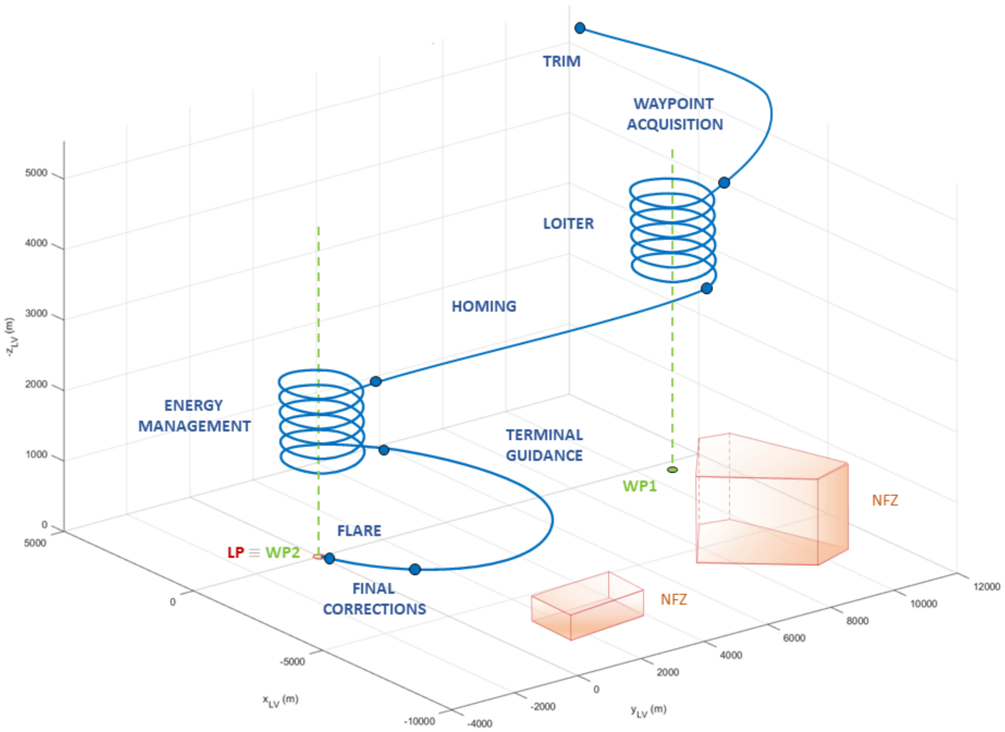

Figure 2-2: PGNC typical sequence

The main objectives of the PGNC are:

- Landing in the vicinity of the LP with a certain precision
- Minimization of velocity to touchdown

An accuracy requirement of 150 meters from the LP is considered in this work. The approach used for the design of PGNC submodes associated with these phases is generally waypoint-based for most of the flight. Algorithms are thus designed to guide the spacecraft along a trajectory that connects a series of waypoints selected a priori as reference. The exception is generally the Terminal Guidance (TERGUID) phase for which different types of approaches can be selected. The one investigated in this work, for instance, is a path-based algorithm that relies on the continuous generation of a reference optimal trajectory given specific boundary conditions.

The flight sequence under parafoil can be broken down into:

- <u>Parafoil deployment and trim</u>:

  The drogue parachute is jettisoned and parafoil deployment takes place, which is associated with an inflation period in which the ram-air parachute goes from the stowed configuration within the RM to the nominal flight configuration. There is then a dynamic transition in which the payload-parafoil system compensates for the aerodynamic effects associated with parafoil deployment until the trimmed flight condition is reached in which the angular moments applied to the system are null.

- <u>Waypoint Acquisition</u>:

  This is the first actual phase of PGNC i.e., the first submode. RM heads to the first of two waypoints (WP1). This reference point is chosen based on several factors including the geographic features of the landing site, the planned position for parafoil deployment, and the possible presence of no-fly zones (NFZ). They are generally related to security concerns such as the presence of buildings. In Figure 2-2, as an example, a couple of no-fly zones are shown in orange.

- Loiter:

  This submode involves a major loss of altitude following a spiral trajectory centered in WP1. Once enough altitude is lost, the triggering for the next phase is performed. This allows potential energy to be lost in a controlled manner while keeping the vehicle within a narrow ground area.

- Homing:

  This is a maneuver similar to Waypoint Acquisition in which the PGNC guides the RM to the second waypoint (WP2). Again, the choice of waypoint depends on many parameters. For the purpose of this study, WP2 is considered to be coincident with the LP. This choice allows the RM to remain in close proximity to the LP when the Terminal Guidance phase is triggered so as to favor the presence of a solution for the TERGUID trajectory generation.

- Energy Management:

  The Energy Management (ENEMNG) submodule performs a function quite similar to that of the loiter, namely, to lose altitude in a controlled manner. The difference with respect to the loiter lies in the exit conditions. They depend on the strategy adopted for TERGUID (for the present work, they are presented in Section 6.1).

- Terminal Guidance:

  This is the core phase of parafoil flight as well as the most critical from a PGNC design perspective. If the previous phases were based on the acquisition of specific Earth-fixed waypoints, here the task is to head the RM toward the LP taking into account the final desired direction and compensating for all disturbance and uncertainty effects. In addition, not too restrictive time margins are allowed for the earlier phases whereas TERGUID is characterized by very strict time limitations. This is due to the fact that since it is the final phase of flight it must point exactly to the LP. Therefore, most of the accuracy requirements depend on it. For a better understanding, considering a horizontal velocity of 15 m/s, a time error of only 5 seconds leads to a horizontal distance of about 75 meters which can lead to non-compliance with the landing accuracy requirement. In addition, the possible presence of wind, which can have an order of magnitude of 5-10 m/s, can further worsen the performance. The wind is indeed the major hurdle for the TERGUID phase. Its effects are unpredictable and the parafoil dynamics tend to be very sensitive to them. In some cases, such as the one under investigation in this work, the ground station provides a wind table generally based on data acquired shortly before the start of the re-entry mission. This table allows for the potential effects of wind to be accounted for, but with an accuracy that is always limited by the temporal variability of wind. In this study, however, the wind direction on the ground at LP will be considered as a known fixed parameter. This aspect is crucial to the development of TERGUID algorithms. Indeed, one of the mission requirements is to minimize velocity at touchdown, which results in the need for an upwind landing in order to minimize the ground speed. For this reason, TERGUID algorithms often make use of the Landing Point Frame (LPF), an LP-centered reference system with x-axis direction coincident with that of the wind on the ground. Please refer to Section 3.3.1 for more details about LPF. There are several possible types of strategies for the TERGUID submode but generally, all fall into the path-based category, i.e., those relying on the continuous online computation of a trajectory that satisfies the conditions of the problem.

- Final Corrections:

  Right after Terminal Guidance, sometimes a short corrective phase is included. During this phase, some final corrections can be made in terms of direction at landing, horizontal distance to be traveled, and lateral velocity, that is, the component of the horizontal groundspeed perpendicular to the x-axis of the Body-Fixed frame (BF). This provides a minimum margin of accuracy to the TERGUID.

- Flare:

  The Flare is the last maneuver before touchdown. It consists of instantaneously commanding maximum symmetric deflection to take advantage of the dynamic response of the parafoil to minimize the vertical velocity at touchdown. This phase is generally associated with an earlier Pre-flare maneuver in which the parafoil is stabilized to prepare it for the flare.

2.2. GNC Physical Architecture

This section briefly describes the typical hardware available to the GNC subsystem during the re-entry phase. The available sensors and actuators are shown in Table 1 [8].

Type	Unit	Number	Comments
Sensor	GPS	2	
Sensor	IMU	2	
Sensor	RADALT	2	
Sensor	Star Tracker	2	Only on AOM
Actuator	Elevon	2	
Actuator	Thruster	4	
Actuator	Parafoil	1	

Table 1: Summary of sensor and actuators for the re-entry phase

Concerning sensors, the general presence of double redundancy is observed. The GPS receiver is intended to provide the horizontal position of the RM but also its altitude for a certain portion of the re-entry phase. In fact, the radar altimeter (RADALT), equipment of aeronautic provenance, can provide sufficiently accurate altitude measurements below a certain altitude threshold, which generally varies between 750 and 1500 meters (about 2500 and 5000 feet). The RADALT then comes into operation in the final part of the flight under parafoil. A key element regarding GPS is the presence of a blackout temporal range between 100 and 50 km due to aerothermodynamic phenomena (see Section 2.1). In addition, partial loss, or degradation of performance due to the vibration environment may occur during TAEM. The GNC must therefore take into account these elements.
Regarding the orientation of the RM in space, two IMU units in hot-redundancy are installed on board. The latter typically provide measurements of acceleration, angular velocity, and attitude through the use of gyroscopes and accelerometers. For the purpose of attitude determination during the orbital phase, there are also two star trackers

units that are, however, mounted on the AOM. Therefore, the data provided by the star trackers are only available in the very early phase of re-entry where the separation between AOM and RM has not yet occurred. They can be used, for instance, for computations related to the boost maneuver performed by the AOM on the RM. An important aspect of the PGNC design is the total absence of airspeed sensors such as Pitot tubes. As a result, no direct measurement of airspeed is provided, which must therefore be estimated indirectly by navigation making use of data given by the other sensors. Furthermore, wind speed, given by the difference between airspeed and ground speed, likewise can only be estimated indirectly. However, for the purpose of the PGNC, the most important information is the one concerning the wind trend below the current flight altitude. This information is provided by the ground station through the wind table and enables the prediction of the effect of wind in terms of displacement from the conditions with the total absence of wind.

As for the actuators, the RM is equipped with a pair of elevons inherited from IXV with some upgrades made using data collected during that mission. A similar discussion applies to thrusters. The RM will in fact mount the same type of unit but with a modified configuration based on lessons learned from the IXV mission. Both elevons and thrusters are disabled during the entire flight under parafoil, which is unpowered. While the drogue parachute plays a passive role without any kind of control authority, the parafoil constitutes a true actuator. In fact, the parafoil can be controlled by the PGNC through a deflection of the lines that connect it to the RM generating a dynamic response that is described in more detail in Section 3.2. However, the absence of other actuation systems during the flight under parafoil makes the control authority very low, which is one of the most impactful restrictions in the design of the PGNC.

2.3. PGNC Functional Architecture

This section briefly describes the architecture of a typical Parafoil GNC (PGNC). Familiarization with its structure is required to understand the logic developed for the Terminal Guidance phase described in Section 6.3. Depending on the design chosen, the PGNC may be integrated within the complete GNC code and considered as a standard submode or constitute a separate segment activated only during the flight phase under parafoil. In the latter case, which is the one considered here, the PGNC may have a different structure than the GNC. The architecture of the Parafoil GNC examined in this study is that shown in Figure 2-3. The hardware part consists of actuators and sensors. The latter provide the PGNC with flight data that is received and processed by the navigation (PNAV). The navigation output, describing the complete state of the system, is then sent to the submode manager (PMNG), which checks the condition of the running submode and verifies whether the conditions for triggering the next submode are met. System status information is also sent to guidance (PGUI) and control (PCON). Guidance algorithms aim to calculate the reference trajectory. These algorithms are naturally dependent on the flight phase, meaning the active submode. The output of the PGUI, in the case of the design under consideration, consists of only two elements:

- Heading Rate (HDR)
- Flight Path Angle (FPA)

The output values of HDR and FPA are the target values to be selected to follow the intended trajectory, controlling the vehicle in the latero-directional plane and longitudinal plane, respectively. These values are sent to the control, which processes them into appropriate command signals to be sent to the actuators. The input required by the latter can be of the type of symmetric/asymmetric deflection or left/right line deflection. The PCON must ensure that the values required by the guidance are physically feasible taking into account the dynamics of the system. This often requires a linearized model of the payload-parafoil dynamic system. Section 3.6 discusses the topic of model linearization. The commands derived from the control are then sent to the actuators, which with their action will influence the plant. These variations will then be detected by sensors and taken into account by navigation. The PGNC described here thus provides a classic close-loop control. The presence of system state feedback is certainly necessary to ensure compliance with the requirements especially considering the strong impact of uncertainties and disturbances.

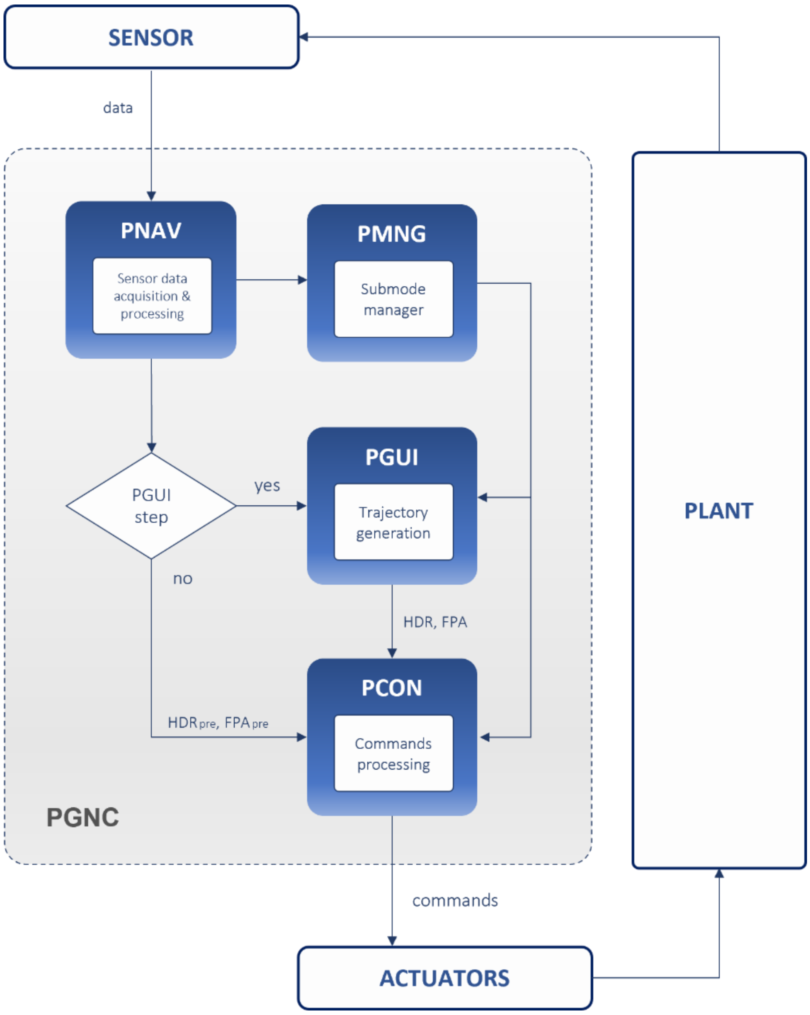

Figure 2-3: PGNC architecture

When implementing a simulator such as the Design Simulation Facility (DSF), it is required to take into account all the elements described above. For this reason, it is required to develop appropriate models to simulate the behavior of the hardware (sensors and actuators) and the dynamic response of the system (plant). The latter is implemented through the so-called Dynamics, Kinematics, and Environment (DKE). Some possible models that can be employed for this purpose are described in Section 3.5.

This study, however, focuses on the development of some possible solutions to be adopted for the PMNG, PGUI and PCON applied specifically to the Terminal Guidance phase. These elements, together with PNAV, collectively constitute the PGNC software. A crucial element in the development of such software is the CPU load budget, especially in the space sector where hardware with low computational capability is employed. One impactful factor for CPU load is the operating frequency. Guidance is generally the component with the highest computational cost, so it is common practice to allocate it a lower frequency compared to the remaining functions. The following frequencies were considered for this study:

- PGUI frequency: 2.5 Hz
- PGNC frequency: 25 Hz

The use of different frequencies for PGNC functions leads to the need to develop special logic to manage their execution over time. Chapter 6 discusses the approach used for the algorithms developed in this study.

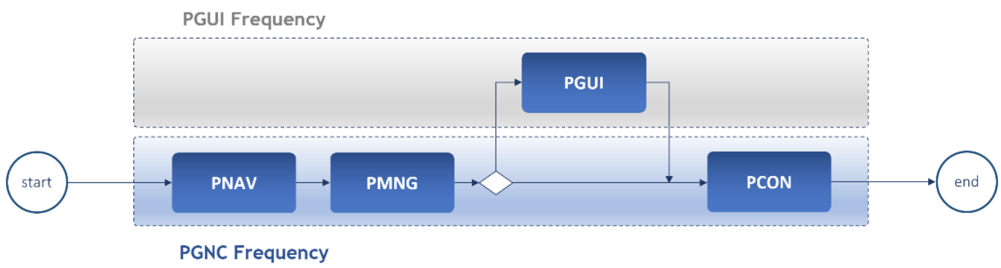

Figure 2-4: PGNC frequencies

3. Parafoil Dynamics

This section provides a brief description of how to derive and linearized a dynamic model of the parafoil outlining the main steps and the assumptions made.

3.1. Autonomous Precision Aerial Delivery Systems

Precision aerial delivery systems (PADS) generally refer to aerospace systems designed to deliver airborne payloads while meeting specific requirements that depend on the type of mission. Among these, there are simple systems such as round canopies that allow uncontrolled re-entry at a constant rate of descent, or more complex systems such as parafoil parachutes. The latter guarantee a re-entry with controlled gliding and a certain degree of steerability. The first such systems were introduced in the first half of the 1960s. In the following years, the wide success of this technology, especially in military applications, led to the rapid development of a rather standard architecture consisting of a ram-air parafoil controlled by electrically driven actuators. Compared with an aircraft, this kind of system is generally devoid of propulsion systems and thus exhibits limited actuation capacity in terms of range of motion and dynamics speed. Despite this, these devices allow autonomous controlled re-entry by means of the design of a suitable GNC subsystem. GNC provides online trajectory planning based on data from sensors and in some cases on information sent from the ground station. There are multiple challenges to be faced in the development of such a subsystem. PADS are generally required to meet stringent requirements in terms of landing accuracy and touchdown velocity, taking into account disturbances such as wind, which can have a velocity of the same order of magnitude as the parafoil airspeed.
Space Rider makes use of these kinds of systems for its final descent and landing phase. Another pivotal project that utilized parafoil parachute was NASA's X-38 program. Developed since 1995, despite its cancellation in 2022 due to budget cuts, X-38 has ensured the collection of a large amount of experimental data obtained through a long series of tests.

3.2. Parafoil Physical Operation

The parafoil is controlled utilizing strokes that are applied through a system of winches. This system is able to convert the signal from the control input of GNC into a command to the parafoil lines. Two types of command can be applied to the parafoil:

- Symmetric deflection δ_s
- Asymmetric deflection δ_a

The symmetric command δ_s of the lines provides control in the longitudinal plane with an effect on the Flight Path Angle (FPA), while the asymmetric command δ_a guarantees control in the latero-directional plane through the heading rate (HDR). It is noted that hereafter, unless otherwise stated, FPA and HDR refer to the Flight Path Angle and Heading Rate relative to the airspeed vector i.e., FpaAir and HdgAir, respectively (see Section 3.3.2 for further details). Although the PGNC works with symmetric and asymmetric stroke commands (δ_s and δ_a), the input to the actuators must be translated into left and right strokes (δ_l and δ_r). A simplified scheme of the operation of the parafoil commands is shown in Figure 3-1 .

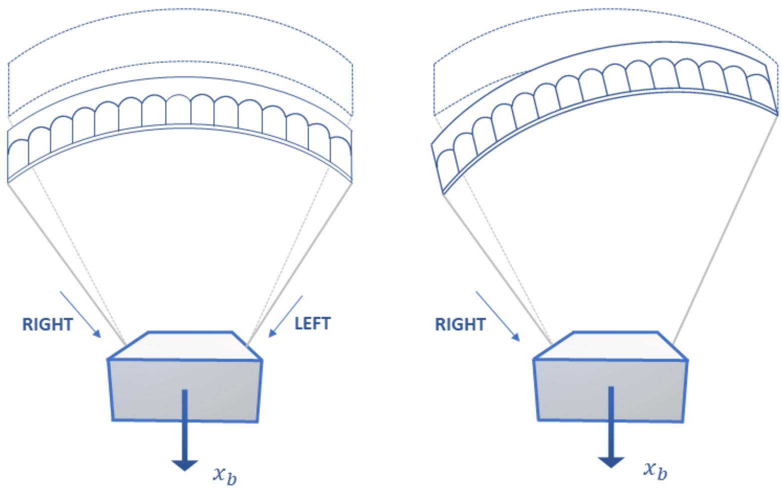

Figure 3-1: Left to right, symmetric and asymmetric deflection

The control authority provided by the parafoil system is generally very limited and usually, for large parafoils, ensures a maximum HDR on the order of 5-10 deg/s. Longitudinal control, in the same way, is able to provide rather limited vertical velocity variations. It is also important to highlight that longitudinal and lateral control authority are mutually dependent. Hence, set a certain value of symmetric deflection (associated with a specific value of FPA) there will be a definite range of HDR values achievable with the asymmetric control. This concept is clearly expressed in Figure 3-2. Such a figure can be obtained by running a series of open-loop simulations in which, as the symmetric deflection (hence the FPA) varies, the symmetric deflection is also varied along the entire range of available deflections observing what value of HDR is obtained. This yields a map of the possible HDR values available once the FPA value is fixed. The values shown are those of a typical large parafoil (of the same type as that used by SR). Each point on the map corresponds to a pair of symmetric (δ_s) and asymmetric (δ_a) deflection values. All in-play values depend on parafoil dynamics but also the mass of the parafoil-payload system as well as air density. Varying these parameters results in a new map associated with different values of δ_s and δ_a. The relationship of FPA and HDR with the asymmetric deflection command is shown in Figure 3-3 by adding $\bar{\delta}_a$ as a third dimension to the graph in Figure 3-2, where $\bar{\delta}_a$ denotes the nondimensional asymmetric deflection:

$$\bar{\delta}_a = \frac{\delta_a}{(\delta_s)_{max}} \in [-1,1] \tag{1}$$

It becomes clear that a trade-off is needed between the available control authority in terms of FPA and HDR. During the Terminal Guidance phase, for instance, it may be convenient to narrow the range of available FPAs (thus of δ_s) in favor of expanding the range of HDRs. The low control authority provided by the parafoil is indeed one of the major challenges in PGNC design, especially for the Terminal Guidance phase. The TERGUID algorithms must in fact be able to generate a suitable trajectory taking into account the limitations in terms of HDR that severely restrict the spectrum of available paths. The maximum HDR constitutes a key design parameter (see Chapter 4).

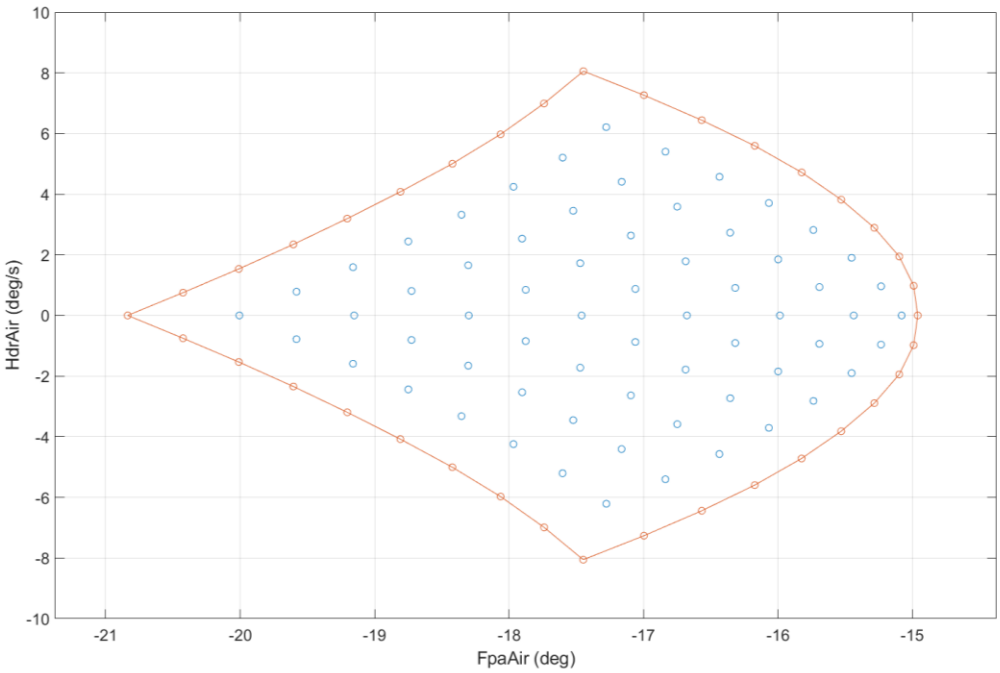

Figure 3-2: Command domain in terms of FPA and HDR

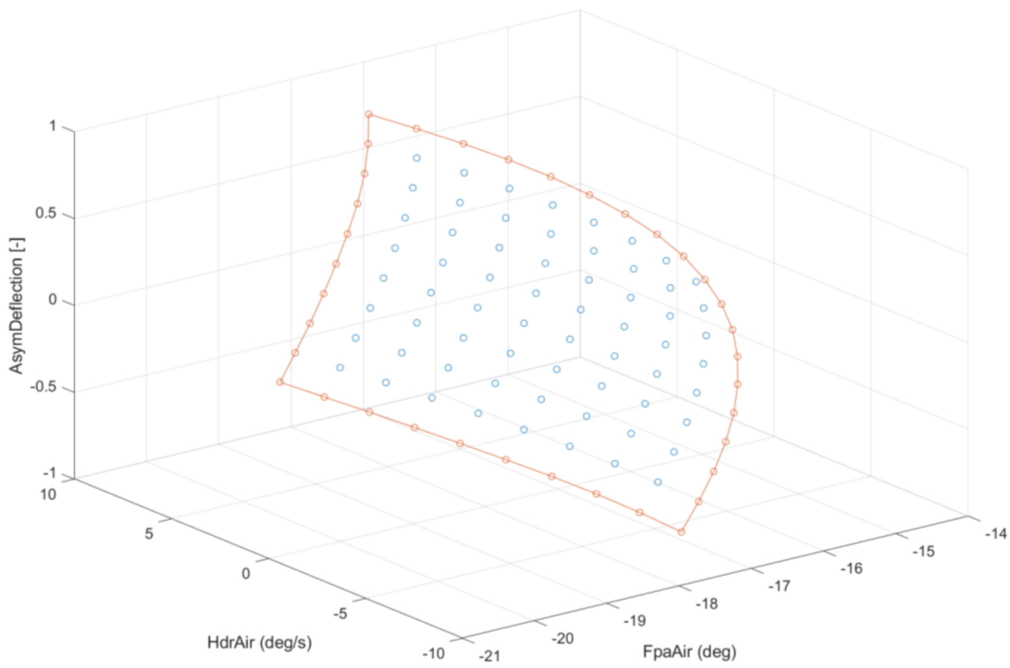

Figure 3-3: Relation HDR-FPA-Asymmetric deflection

3.3. Basic Definitions

This section discusses the basic definitions and fundamentals required for the derivation of the dynamic model of the parafoil.

3.3.1. Reference Frames

To derive the parafoil dynamic model and in general for the scopes of this work, it is first necessary to define the set of right-handed reference frames listed below.

- Local Vertical Frame:

  The Local Vertical (LV) frame is north-east-down (NED) Earth-fixed reference system with the origin in an arbitrary point on the ground and the axes x_{LV}, y_{LV}, z_{LV} (or x, y, z) defined as follows:

  x_{LV}: Positive towards the true North along local meridian
  y_{LV}: Positive towards the East
  z_{LV}: Positive downwards along the ellipsoid Earth model normal

- Navigational Frame:

  The Navigational Frame (NF) has the same orientation as LV, with the origin placed in the center of mass of the system (CoM) and axes:

  x_n: Positive towards the true North along local meridian
  y_n: Positive towards the East
  z_n: Positive downwards along the ellipsoid Earth model normal

- Landing Point Frame:

  The Landing Point Frame (LPF) is a coordinate system with the origin on the ground at the landing point (LP). The orientation of the axes depends on the direction of the horizontal component of the windspeed at LP (ground level). The axes are defined as follows:

  x_{LPF}: Negative in the direction of the horizontal windspeed at LP
  y_{LPF}: Positive direction to comply with the right-hand rule
  z_{LPF}: Positive downwards along the ellipsoid Earth model normal

- Body-Fixed Frame:

  The Body-Fixed (BF) frame is fixed with respect to the nominal geometry of the vehicle with origin in the CoM of the system. The x and z axes are contained in the symmetry plane of the system while the y axis is perpendicular to it. The direction of the axes is determined as follows:

  x_b: Positive towards the nose of the vehicle
  y_b: Positive direction to comply with the right-hand rule
  z_b: Positive toward the lower part of the vehicle

- Air-Path Frame:

  The Air-Path Frame (AF) is a reference frame with origin in the CoM of the system and orientation associated with the direction of the airspeed vector according to the following definitions:

  x_a: Positive in the same direction as the airspeed vector $\boldsymbol{v_a}$
  y_a: Positive direction to comply with the right-hand rule
  z_a: Positive downwards, contained in the symmetry plan

- Canopy Frame:

  The Canopy Frame (CF) is a body-fixed reference frame with origin in the CoM of the parafoil system. The x and z axes are contained in the symmetry plane of the system while the y axis is perpendicular to it. The direction of the axes is determined as follows:

  x_c: Positive forward, parallel to the parafoil aerodynamic ref. line
  y_c: Positive direction to comply with the right-hand rule
  z_c: Positive towards the RM

Taking into account the fact that the parafoil model is used for the final stages of the descent and landing phase, some simplifications can be made due to limited variation in latitude and longitude throughout the entire trajectory and the short duration of the flight (the D&L phase under parafoil can typically take around 10-20 minutes). Under these assumptions, it is possible to consider LV as inertial and approximate the Earth as a flat surface.

Table 2 lists the reference systems described above and the related nomenclature, while Figure 3-4 offers a simplified visual representation of them.

Name	Abbreviation	Symbol	Axes nomenclature
Local Vertical Frame	LV	LV	$[\mathrm{x}, \mathrm{y}, \mathrm{z}]^{\mathrm{T}}$ or $[\mathrm{x_{LV}}, \mathrm{y_{LV}}, \mathrm{z_{LV}}]^{\mathrm{T}}$
Navigational Frame	NF	n	$[\mathrm{x_n}, \mathrm{y_n}, \mathrm{z_n}]^{\mathrm{T}}$
Landing Point Frame	LPF	LPF	$[\mathrm{x_{LPF}}, \mathrm{y_{LPF}}, \mathrm{z_{LPF}}]^{\mathrm{T}}$
Body-Fixed Frame	BF	b	$[\mathrm{x_b}, \mathrm{y_b}, \mathrm{z_b}]^{\mathrm{T}}$
Air-Path Frame	AF	a	$[\mathrm{x_a}, \mathrm{y_a}, \mathrm{z_a}]^{\mathrm{T}}$
Canopy Frame	CF	c	$[\mathrm{x_c}, \mathrm{y_c}, \mathrm{z_c}]^{\mathrm{T}}$

Table 2: Reference frames nomenclature

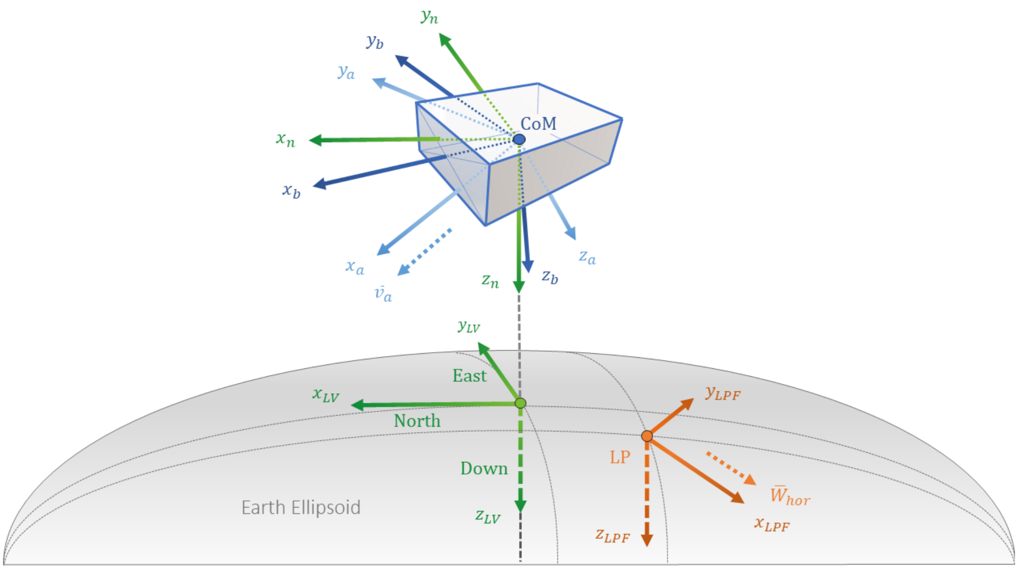

Figure 3-4: Reference frames simplified representation

3.3.2. Frame Rotations

This section describes the rotations between the major reference frames and the angles associated with them. Each set of angles will uniquely define the relative orientation between two specific reference systems. Listed below are the main reference frame transformations.

- <u>Transformation: BF → LV/NF</u>

  The rotation from Body-Fixed Frame (BF) to Local Vertical (LV) or Navigational Frame (NF) is given by the following Euler 321 sequence:

  3) ψ: yaw angle
  2) θ: pitch angle
  1) ϕ: roll angle

- <u>Transformation: BF → AF</u>

  The rotation from Body-Fixed Frame (BF) to Air-Path Frame (AF) is given by the following Euler 32 sequence:

  3) α: angle of attack
  2) β: sideslip angle

- <u>Transformation: AF → LV/NF</u>

  The rotation from Air-Path Frame (AF) to Local Vertical (LV) or Navigational Frame (NF) is given by the following Euler 321 sequence:

  3) χ_a: heading angle air (HdgAir)
  2) γ_a: flight path angle air (FpaAir)
  1) ϕ_a: bank angle air

It is also possible to define a reference frame analogous to the air-path frame but with x-axis direction parallel to the groundspeed instead of airspeed. In the case of no wind, the two reference frames are coincident (see Section 3.3.3.). The three Euler angles for the transformation from such a reference system to Local Vertical (LV) or Navigational Frame (NF) will be simply named heading angle, flight path angle, and bank angle (χ, γ, ϕ respectively).
By means of the aforementioned Euler angles, the respective Direct Cosine Matrix (DCM) can be constructed. The DCM constitutes a geometric operation that projects the axes of the initial reference system onto those of the target system. The convention used in this work to denote the rotation matrix from a reference frame i to a reference frame j is as follows:

$$\boldsymbol{R}_i^j = \boldsymbol{R}_{ji} = \boldsymbol{R}_{i \to j} \tag{2}$$

Following the convention outlined above, it is possible to go from a frame i to a frame j by passing through an intermediate frame k with the following matrix operation:

$$\boldsymbol{R}_{ji} = \boldsymbol{R}_{jk}\boldsymbol{R}_{ki} = \boldsymbol{R}_{kj}^{\boldsymbol{T}}\boldsymbol{R}_{ji} \tag{3}$$

The simplest way to construct DCMs is by appropriate multiplication of single-axis rotation matrices. As an example, below is the Navigational Frame (NF) to Body-fixed Frame (BF) rotation matrix:

$$\begin{gathered}\boldsymbol{R}_{bn} = \boldsymbol{R}_{n\to b} = \boldsymbol{R}_{\phi}\boldsymbol{R}_{\theta}\boldsymbol{R}_{\psi} = \\ = \begin{bmatrix} 1 & 0 & 0 \\ 0 & c_\phi & s_\phi \\ 0 & -s_\phi & c_\phi \end{bmatrix} \begin{bmatrix} c_\theta & 0 & -s_\theta \\ 0 & 1 & 0 \\ s_\theta & 0 & c_\theta \end{bmatrix} \begin{bmatrix} c_\psi & s_\psi & 0 \\ -s_\psi & c_\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} = \\ = \begin{bmatrix} c_\psi\ c_\theta & s_\psi\ c_\theta & -s_\theta \\ c_\psi\ s_\theta\ s_\phi - s_\psi\ c_\phi & s_\psi\ s_\theta\ s_\phi + c_\psi\ c_\phi & c_\theta\ s_\phi \\ c_\psi\ s_\theta\ c_\phi + s_\psi\ s_\phi & s_\psi\ s_\theta\ c_\phi - c_\psi\ s_\phi & c_\theta\ c_\phi \end{bmatrix}\end{gathered} \tag{4}$$

where s_α and c_α denote the sine and cosine of angle α, respectively. Similarly, it is possible to derive the rotation matrices for the other transformations listed above.
An additional important step is the transformation of the angular rates from Local Vertical (LV) to Body-fixed Frame (BF) and vice versa. Taking into account that the Euler angles of yaw, pitch, and roll express the orientation of BF with respect to LV, the corresponding derivatives indicate the angular velocity of BF with respect to LV. The components of the angular velocity vector can then be projected into BF obtaining:

$$\boldsymbol{\omega} = \begin{bmatrix} p \\ q \\ r \end{bmatrix} = \boldsymbol{R}_{\phi} \begin{bmatrix} \dot{\phi} \\ 0 \\ 0 \end{bmatrix} + \boldsymbol{R}_{\phi}\boldsymbol{R}_{\theta} \begin{bmatrix} 0 \\ \dot{\theta} \\ 0 \end{bmatrix} + \boldsymbol{R}_{\phi}\boldsymbol{R}_{\theta}\boldsymbol{R}_{\psi} \begin{bmatrix} 0 \\ 0 \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & 0 & -s_\theta \\ 0 & c_\phi & c_\theta\ s_\phi \\ 0 & -s\phi & c_\theta\ c_\phi \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \tag{5}$$

where $\boldsymbol{\omega}$, with components p, q, r, denotes the angular rate of BF with respect to LV expressed in BF. By inverting the matrix in equation (5), it is possible to obtain the inverse transformation:

$$\begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} 1 & s_\phi\ s_\theta/c_\theta & c_\phi\ s_\theta/c_\theta \\ 0 & c_\phi & -s_\phi \\ 0 & s_\phi/c_\theta & c_\phi/c_\theta \end{bmatrix} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{6}$$

3.3.3. Kinematic and Aerodynamic Variables

This section lists and derives some of the main kinematic and aerodynamic variables used in this work in order to become familiar with the notation employed. The position vector is always expressed in LV, so subscripts are omitted. Furthermore, remembering that the z-axis points downward, the variable z coincides with the opposite of the altitude value h (measured with respect to ground level at the origin chosen for LV):

$$\boldsymbol{X} = [x, y, z]^T = [x, y, -h]^T \tag{7}$$

The groundspeed vector is defined as follows:

$$\boldsymbol{V} = \boldsymbol{V}_a + \boldsymbol{W} \tag{8}$$

where $\boldsymbol{V_a}$ is airspeed and $\boldsymbol{W}$ is the wind speed. It is important to remark that in the case of flight under parafoil, airspeed $\boldsymbol{V_a}$ and wind speed $\boldsymbol{W}$ may have a comparable order of magnitude, so $\boldsymbol{V_a}$ and $\boldsymbol{V}$ may differ significantly from each other. As far as velocities are concerned, uppercase letters will be used for variables with components expressed in LV while lowercase letters will be used for those with components expressed in BF, as shown in Table 3.

Variable name	Components in LV	Components in BF
Groundspeed vector	$\boldsymbol{V} \quad = [V_x, V_y, V_z]^T$	$\boldsymbol{v} \ = [u, v, w]^T$
Airspeed vector	$\boldsymbol{V}_a = [V_{xa}, V_{ya}, V_{za}]^T$	$\boldsymbol{v}_a = [\ u_a, v_a, w_a]^T$
Wind speed vector	$\boldsymbol{W} = [W_x, W_y, W_z]^T$	$\boldsymbol{w} = [\ w_x, w_y, w_z]^T$

Table 3: Linear velocities nomenclature

It is worth mentioning that airspeed expressed in Air-path Frame (AF) has only a component along the x-axis equal to its norm, i.e., $[V_a, 0,0]^T$. Now, recalling the definitions of Euler angles given in Section 3.3.2, it is possible to derive the components of the velocity vectors as a function of Euler angles. An overview of the velocities in play together with the sign convention used for α and β is shown in Figure 3-5.

$$\boldsymbol{v} = \begin{bmatrix} u \\ v \\ w \end{bmatrix} = V \begin{bmatrix} cos\alpha \ cos\beta \\ sin\beta \\ sin\alpha \ cos\beta \end{bmatrix} \tag{9}$$

$$\boldsymbol{V} = \begin{bmatrix} V_x \\ V_y \\ V_z \end{bmatrix} = \begin{bmatrix} V_h cos\chi \\ V_h sin\chi \\ V_v \end{bmatrix} = V \begin{bmatrix} cos\gamma cos\chi \\ cos\gamma sin\chi \\ sin\gamma \end{bmatrix} = \boldsymbol{V}_h + \boldsymbol{V}_v \tag{10}$$

$$\boldsymbol{V}_a = \begin{bmatrix} V_{xa} \\ V_{ya} \\ V_{za} \end{bmatrix} = \begin{bmatrix} V_{ha} cos\chi_a \\ V_{ha} sin\chi_a \\ V_{va} \end{bmatrix} = V_a \begin{bmatrix} cos\gamma_a cos\chi_a \\ cos\gamma_a sin\chi_a \\ sin\gamma_a \end{bmatrix} = \boldsymbol{V}_{ha} + \boldsymbol{V}_{va} \tag{11}$$

where $\boldsymbol{V}_v$ and $\boldsymbol{V}_{va}$ denote the vertical components of $\boldsymbol{V}$ and $\boldsymbol{V}_a$ in LV, while $\boldsymbol{V}_h$ and $\boldsymbol{V}_{ha}$ denote the horizontal components of $\boldsymbol{V}$ and $\boldsymbol{V}_a$ in LV. In addition, the vertical component of V in LV corresponds to:

$$V_v = V_z = Vsin\gamma = -\frac{dh}{dt} \tag{12}$$

From equations (9),(10), and (11) it is then straightforward to obtain the value of the angles involved by reversing the equations. For instance:

$$\gamma_a = -asin\left(\frac{V_{za}}{V_a}\right) \quad \text{and} \quad \chi_a = atan\left(\frac{V_{ya}}{V_{xa}}\right) \tag{13}$$

$$\alpha = atan\left(\frac{V_{za}}{V_{xa}}\right) \quad \text{and} \quad \beta = atan\left(\frac{V_{ya}}{\sqrt{V_{xa}^2+V_{za}^2}}\right) \tag{14}$$

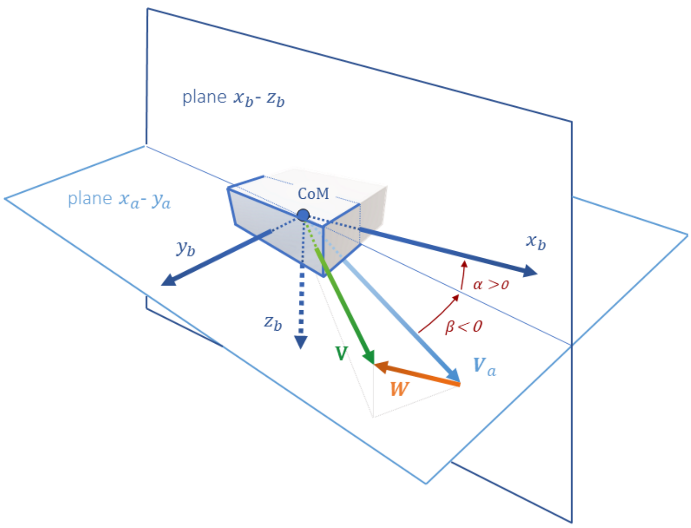

Figure 3-5: Velocity vectors overview

3.4. Translational and Rotational Dynamics

This section reports the main players and equations involved in the generation of a parafoil model.

3.4.1. Dynamical Equations

The starting equations for modeling parafoil dynamics are, as usual, the classical Newton's second law and Euler's equations. The following are, in order, the equations governing translational and rotational dynamics expressed in LV:

$$m\dot{\boldsymbol{V}} = m \begin{bmatrix} \dot{V}_x \\ \dot{V}_y \\ \dot{V}_z \end{bmatrix} = \boldsymbol{F}^{LV} \tag{15}$$

$$\boldsymbol{I_t}\dot{\tilde{\boldsymbol{\omega}}} = \boldsymbol{I_t} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} = \boldsymbol{M}^{LV} \tag{16}$$

where $\boldsymbol{F}^{LV}$ and $\boldsymbol{M}^{LV}$ are the external force and momentum vectors expressed in LV, respectively. $\boldsymbol{I_t}$ is the inertia tensor, which generally for parafoils, given the hypothesis of symmetry with respect to the x-y plane, has the following form:

$$\boldsymbol{I_t} = \begin{bmatrix} I_{xx} & 0 & I_{xz} \\ 0 & I_{yy} & 0 \\ I_{xz} & 0 & I_{zz} \end{bmatrix} \tag{17}$$

where the moments of inertia I_{ii} and the product of inertia I_{ij} can be computed by means of the following integrals:

$$I_{xx} = \int_m (y_b^2 + z_b^2)\, dm, \quad I_{xx} = \int_m (x_b^2 + z_b^2)\, dm, \quad I_{zz} = \int_m (x_b^2 + y_b^2)\, dm \tag{18}$$

$$I_{xz} = -\int_m (x_b z_b)\, dm \tag{19}$$

Considering the noninertial BF, the translational equation (15) and the rotational equation (16) must take into account appropriate additional terms:

$$m\dot{\boldsymbol{v}} + \boldsymbol{\omega} \times m\boldsymbol{v} = m \begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} + m \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} u \\ v \\ w \end{bmatrix} = \boldsymbol{F} \tag{20}$$

$$\boldsymbol{I_t}\dot{\omega} + \boldsymbol{\omega} \times \boldsymbol{I_t}\boldsymbol{\omega} = \boldsymbol{I_t} \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} + \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \boldsymbol{I_t} \begin{bmatrix} p \\ q \\ r \end{bmatrix} = \boldsymbol{M} \tag{21}$$

where $\boldsymbol{F}$ and $\boldsymbol{M}$ are the external force and momentum vectors expressed in BF, respectively. It is also possible to introduce the so-called skew-symmetric matrix $\boldsymbol{S}(\boldsymbol{x})$ in order to obtain more compact writing of the equations in the presence of cross products:

$$\mathbf{S}(\boldsymbol{x}) = \mathbf{S}([x_1, x_2, x_3]) = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix} \tag{22}$$

Applying definition (22) to the equations (20) and (21) gives:

$$m \begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} + m \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \begin{bmatrix} u \\ v \\ w \end{bmatrix} = m \begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} + m\boldsymbol{S}(\boldsymbol{\omega}) \begin{bmatrix} u \\ v \\ w \end{bmatrix} = \boldsymbol{F} \tag{23}$$

$$\boldsymbol{I_t} \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} + \begin{bmatrix} p \\ q \\ r \end{bmatrix} \times \boldsymbol{I_t} \begin{bmatrix} p \\ q \\ r \end{bmatrix} = \boldsymbol{I_t} \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} + \boldsymbol{S}(\boldsymbol{\omega})\boldsymbol{I_t} \begin{bmatrix} p \\ q \\ r \end{bmatrix} = \boldsymbol{M} \tag{24}$$

Keeping the derivative terms on the left-hand side of the equation and bringing the others to the right-hand side, we obtain the Newton-Euler equations in the compact form:

$$\begin{bmatrix} m\boldsymbol{I}_{3x3} & \mathbf{0}_{3x3} \\ \mathbf{0}_{3x3} & \boldsymbol{I_t} \end{bmatrix} \begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \\ \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \boldsymbol{F} - m\boldsymbol{S}(\boldsymbol{\omega}) \begin{bmatrix} u \\ v \\ w \end{bmatrix} \\ \\ \boldsymbol{M} - \boldsymbol{S}(\boldsymbol{\omega})\boldsymbol{I_t} \begin{bmatrix} p \\ q \\ r \end{bmatrix} \end{bmatrix} \tag{25}$$

In the specific case of unpowered PADS, it is also possible to derive an additional form for Newton's second law. Considering the Air-path Frame (AF) and the expressions in Section 3.3.3, the equation (15) becomes [9] :

$$m\begin{bmatrix}\dot{V}_a\\ \dot{\gamma}_a\\ \dot{\chi}_a\end{bmatrix} = \begin{bmatrix} D + mg\ sin\gamma_a \\ \frac{1}{V_a}(L\cos\phi_a - mg\ cos\gamma_a) \\ \frac{1}{V_a}\frac{L\ sin\phi_a}{cos\gamma_a}\end{bmatrix} \tag{26}$$

where L is the lift and D the drag lying on the z-axis and x-axis of AF respectively (both in the negative direction).

3.4.2. Forces and Moments

This paragraph describes the main forces and moments acting on the parafoil-payload system considering the latter as a rigid body with 6 degrees of freedom following the scheme shown in Figure 3-6 (in which airspeed is assumed to be contained in the plane of symmetry). The center of mass to be considered for the application of forces and moments is the one of the complete system and the mass of the system is given by:

$$m = m_{payload} + m_{parafoil} \tag{27}$$

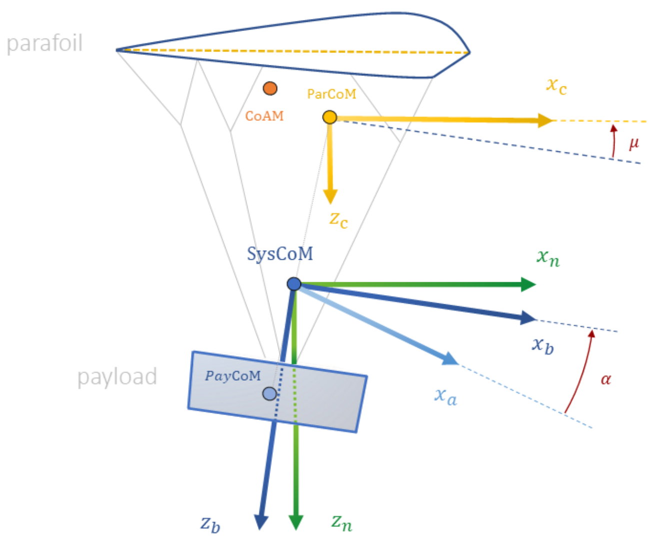

Figure 3-6: Parafoil-payload system 6DOF model

The gravity force vector expressed in BF is as follows:

$$\boldsymbol{F}_G = \begin{bmatrix} -sin\theta \\ cos\theta \ sin\phi \\ cos\theta \ cos\phi \end{bmatrix} mg \tag{28}$$

Regarding aerodynamic forces and moments, there are contributions generated by both the payload and the parafoil. In the case of medium to large canopies, in general, the parafoil contributions are clearly predominant. However, considering BF and combining both contributions into a single aerodynamic model yields [10]:

$$\boldsymbol{F}_A = \boldsymbol{R}_{ba} \frac{\rho V_a^2 S_p}{2} \begin{bmatrix} C_{D0} + C_{D\alpha^2}\alpha^2 + C_{D\delta_s}\bar{\delta}_s \\ C_{Y\beta}\beta \\ C_{L0} + C_{L\alpha}\alpha + C_{L\delta_s}\bar{\delta}_s \end{bmatrix} \tag{29}$$

$$\boldsymbol{M}_A = \frac{\rho V_a^2 S_p}{2} \begin{bmatrix} b\left(C_{l\beta}\beta + \frac{b}{2V_a}C_{lp}p + \frac{b}{2V_a}C_{lr}r + C_{l\delta_a}\bar{\delta}_a\right) \\ \bar{c}\left(C_{m0} + C_{m\alpha}\alpha + \frac{c}{2V_a}C_{mq}q\right) \\ b\left(C_{n\beta}\beta + \frac{b}{2V_a}C_{np}p + \frac{b}{2V_a}C_{nr}r + C_{n\delta_a}\bar{\delta}_a\right. \end{bmatrix} \tag{30}$$

where ρ is the density, S_p the parafoil canopy area and b the span of the inflated parafoil canopy (see Figure 3-7). The parameter $\bar{c}$ is the mean aerodynamic chord and is given by:

$$\bar{c} = \frac{2}{S_p}\int_0^{\frac{b}{2}} c(y)dy \tag{31}$$

The coefficients C_{ij} are the conventional aerodynamic coefficients used in flight mechanics. For their definition, please refer to any book on atmospheric flight and flight mechanics such as [2]. Their values are measured experimentally by specific flight tests. $\bar{\delta}_a$ and $\bar{\delta}_s$ are the control variables and constitute the non-dimensional asymmetric and symmetric deflection values, respectively. They are defined as follows:

$$\bar{\delta}_a = \frac{\delta_a}{(\delta_s)_{max}} \in [-1,1] \quad \text{and} \quad \bar{\delta}_s = \frac{\delta_s}{(\delta_s)_{max}} \in [0,1] \tag{32}$$

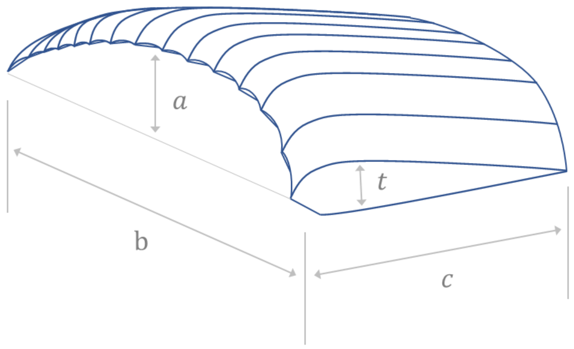

Figure 3-7: Main geometrical parameters of parafoil

3.4.3. Apparent and Enclosed Mass

In the specific case of the ram-air canopy of parafoils, there is the necessity to consider two additional effects for dynamic modeling:

- Entrapped air in the ram-air canopy
- Apparent mass forces and moments

The special shape of the ram-air canopy causes a certain amount of air to be trapped within its structure and move inertially with it. The total mass of entrapped air depends on the size of the parafoil, cannot be neglected for medium to large parafoils, and is generally referred to as enclosed or included mass. There are several models for computing this mass that depend on the type of parafoil. In some cases, this value is provided by the manufacturer. Therefore, the total mass of the system, considering all contributions, becomes:

$$m = m_{payload} + m_{parafoil} + m_{enclosed} \tag{33}$$

The second effect that must be considered is the one associated with the dynamics of the volume of air surrounding the canopy. When a body is in motion within a fluid, local and global accelerations set the fluid itself in motion generating a response in terms of force on the surface of the body, i.e., the so-called apparent mass pressure field. The magnitude of this effect depends on the ratio M_r between the body mass and the displaced fluid mass, thus on the type of vehicle.

For this reason, for conventional aircraft, where the ratio M_r is high, apparent mass pressure is typically assumed negligible. In contrast, for lighter-than-air vehicles, this effect must be taken into account because it has a significant impact on dynamics. In the specific case of parafoil, the canopy has a low mass density associated with an extensive surface that is capable of displacing a considerable amount of air. In addition, the application point of the apparent mass forces may be fairly distant from the center of mass of the system increasing the effects in terms of momentum. For parafoils, it is also possible to approximate the value of the mass ratio with the following formula [10]:

$$M_r = \frac{m}{\rho {S_p}^{3/2}} \tag{34}$$

This value decreases with parafoil size and generally ranges from 0.5 to 6. Computational Fluid Dynamics (CFD) numerical analysis or analytical models based on experimental data can be employed to evaluate the impact of apparent mass on parafoil dynamics. For a more detailed discussion of the topic, see [11] and [12]. A possible analytical formulation for modeling the apparent mass force and apparent inertia moment in CF is the one described in [10], that is:

$$\boldsymbol{F}_{AM}^{C} = -\left(\boldsymbol{I}_{AM} \begin{bmatrix} \dot{\hat{u}}_{ac} \\ \dot{\hat{v}}_{ac} \\ \dot{\hat{w}}_{ac} \end{bmatrix} + \boldsymbol{S}(\hat{\boldsymbol{\omega}}_c) \boldsymbol{I}_{AM} \begin{bmatrix} \hat{u}_{ac} \\ \hat{v}_{ac} \\ \hat{w}_{ac} \end{bmatrix} \right) \tag{35}$$

$$\boldsymbol{M}_{AI}^{C} = -\left(\boldsymbol{I}_{AI} \begin{bmatrix} \dot{\hat{p}}_{c} \\ \dot{\hat{q}}_{c} \\ \dot{\hat{r}}_{c} \end{bmatrix} + \boldsymbol{S}(\hat{\boldsymbol{\omega}}_c) \boldsymbol{I}_{AI} \begin{bmatrix} \hat{p}_{c} \\ \hat{q}_{c} \\ \hat{r}_{c} \end{bmatrix} + \boldsymbol{S}(\hat{\boldsymbol{v}}_{ac}) \boldsymbol{I}_{AM} \begin{bmatrix} \hat{u}_{ac} \\ \hat{v}_{ac} \\ \hat{w}_{ac} \end{bmatrix} \right) \tag{36}$$

where the canopy airspeed vector in CF and the canopy angular rate vector in CF are respectively defined as follows:

$$\hat{\boldsymbol{v}}_{ac} = [\hat{u}_{ac}, \hat{v}_{ac}, \hat{w}_{ac}]^T \tag{37}$$

$$\hat{\boldsymbol{\omega}}_c = [\hat{p}_c, \hat{q}_c, \hat{r}_c]^T \tag{38}$$

The matrices $\boldsymbol{I}_{AM}$ and $\boldsymbol{I}_{AI}$ are named apparent mass matrix and apparent inertia matrix, respectively, and have a diagonal form:

$$\boldsymbol{I}_{AM} = \begin{bmatrix} A & 0 & 0 \\ 0 & B & 0 \\ 0 & 0 & C \end{bmatrix} \tag{39}$$

$$\boldsymbol{I}_{AI} = \begin{bmatrix} I_A & 0 & 0 \\ 0 & I_B & 0 \\ 0 & 0 & I_C \end{bmatrix} \tag{40}$$

For the calculation of the components A, B, C and I_A, I_B, I_C of the two matrices above, please refer to the in-depth discussion by Lissaman and Brown [11].

3.5. Parafoil-Payload System Models

This section draws some possible solutions for modeling the dynamics of a parafoil considering different degrees of freedom. Simplified models with 3 or 4 DOF are often adequate for design and verification of the functioning of guidance algorithms. In other cases, however, it proves necessary to use a more complex model with more degrees of freedom that can more accurately describe the dynamic evolution of the system. The Dynamics, Kinematics, and Environment (DKE) model of a simulator used for verification and validation of a GNC software, needs, for instance, a model with at least 6DOF. The simulator developed by SENER Aeroespacial employed for testing the algorithms described in this work contains a 6DOF model of the payload-parafoil system. Table 4 shows the kinematic and dynamic variables taken into account for the different models proposed in this chapter (3, 4, 6 DOF). Translational motion is defined as the motion of the center of mass of the system in LV. Rotational motion, on the other hand, refers to the rotation of BF with respect to LV (or NF). The main assumptions made for each of them are also summarized in the same table.

Model	Translational motion	Rotational motion	Main assumptions
3DOF	(x, y),(u, w)	ψ	$\beta \approx 0, V_{ha} \approx V_{ha0}, V_{va} \approx V_{va0}$
4DOF	(x, y),(u, w)	(ψ, ϕ)	$\beta \approx 0, \theta \approx 0, \dot{\theta} \approx 0$
6DOF	(x, y, z),(u, v, w)	(ψ, ϕ, θ),(p, q, r)	$CoAM \equiv ParCoM$

Table 4: Dynamic and kinematic variables for different parafoil models

3.5.1. 3DOF Model

The simplest flight mechanical model for the parafoil-payload systems is the 3-degree-of-freedom model, which is based on the kinematic equations alone. This model is useful for describing only the translational motion of the center of mass CoM of the system without providing any information about the attitude. The translational kinematics of a vehicle as a rigid body in 3 dimensions can be expressed as follows:

$$\boldsymbol{V} = \boldsymbol{V}_a + \boldsymbol{W} = \boldsymbol{R}_{bn}^T \boldsymbol{v} \tag{41}$$

Explicitly expressing the velocity components in the two reference frames LV and BF leads to the following:

$$\begin{bmatrix} V_x \\ V_y \\ V_z \end{bmatrix} = \begin{bmatrix} V_{xa} \\ V_{ya} \\ V_{za} \end{bmatrix} + \begin{bmatrix} W_x \\ W_y \\ W_z \end{bmatrix} = \begin{bmatrix} V_{ha} cos\chi_a \\ V_{ha} sin\chi_a \\ V_{va} \end{bmatrix} + \begin{bmatrix} W_x \\ W_y \\ W_z \end{bmatrix} = \begin{bmatrix} V_a cos\gamma_a cos\chi_a \\ V_a cos\gamma_a sin\chi_a \\ V_a sin\gamma_a \end{bmatrix} + \begin{bmatrix} W_x \\ W_y \\ W_z \end{bmatrix} \tag{42}$$

with:

$$\begin{bmatrix} V_x \\ V_y \\ V_z \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \boldsymbol{R}_{bn}^T \begin{bmatrix} u \\ v \\ w \end{bmatrix} = \boldsymbol{R}_{an}^T \begin{bmatrix} V_a \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} W_x \\ W_y \\ W_z \end{bmatrix} \tag{43}$$

A typical assumption in PADS modeling is to neglect the sideslip angle so that the yaw angle coincides with the heading angle. A further common assumption concerns the norm of the horizontal and vertical airspeeds that are considered constant along the trajectory. Under this assumption, the L/D ratio is considered constant throughout the trajectory, neglecting elements such as variation in air density and dynamic effects of motion in the longitudinal plane. The dynamic effects of motion in the latero-directional plane are also disregarded in this discussion. The above assumptions are summarized in the expressions (44), (45), and (46) which are valid for the entire trajectory.

$$\beta \approx 0 \quad \rightarrow \quad \chi_a \approx \psi \tag{44}$$

$$V_{ha} \approx V_{ha0} \tag{45}$$

$$V_{va} \approx V_{va0} \tag{46}$$

Considering the assumptions above, the equation (42) becomes:

$$\begin{bmatrix} V_x \\ V_y \\ V_z \end{bmatrix} = \begin{bmatrix} V_a cos\gamma_a cos\chi_a \\ V_a cos\gamma_a sin\chi_a \\ V_a sin\gamma_a \end{bmatrix} + \begin{bmatrix} W_x \\ W_y \\ W_z \end{bmatrix} \approx \begin{bmatrix} V_{ha0}\ cos\psi \\ V_{ha0}\ sin\psi \\ V_{va0} \end{bmatrix} + \begin{bmatrix} W_x \\ W_y \\ W_z \end{bmatrix} \tag{47}$$

It is then possible to improve the model by considering for latero-directional dynamics the relationship between asymmetric deflection and yaw rate. Following the approach proposed by Jann in [13], the following first-order dynamics of the yaw rate can be introduced:

$$T_\psi \ddot{\psi} + \dot{\psi} = K_\psi \delta_a \tag{48}$$

where K_ψ and T_ψ are weighting coefficients given by flight data. The equation (48) can also be expressed in the state-space form:

$$\begin{bmatrix} \dot{\psi} \\ \ddot{\psi} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & T_\psi^{-1} \end{bmatrix} \begin{bmatrix} \psi \\ \dot{\psi} \end{bmatrix} + T_\psi^{-1} \begin{bmatrix} 0 \\ K_\psi \end{bmatrix} \delta_a \tag{49}$$

A final measure to improve the accuracy of the 3 DOF model includes considering the influence of turn rate (i.e., yaw rate) on vertical velocity by introducing an additional term in the (47):

$$\begin{bmatrix} V_x \\ V_y \\ V_z \end{bmatrix} = \begin{bmatrix} V_{ha0}\ cos\psi \\ V_{ha0}\ sin\psi \\ V_{va0} + K_{\dot{\psi}} \dot{\psi} \end{bmatrix} + \begin{bmatrix} W_x \\ W_y \\ W_z \end{bmatrix} \tag{50}$$

where $K_{\dot{\psi}}$ is a coefficient given by flight data. Equations (49) and (50) provide a simplified 3DOF model for the parafoil-payload system. It is highlighted that, although the dynamical equations have not been considered, some dynamic effects were introduced with the two last steps described. In addition, the aerodynamics of the parafoil is included in the choice of parameters K_ψ, T_ψ, $K_{\dot{\psi}}$ and V_{ha0}, V_{va0}. In particular, for the airspeed, the values that characterize the trimmed flight under conditions similar to those intended to be simulated are chosen. Despite the simplicity of this model, there are several possible applications as, for instance, testing guidance algorithms such as the one described in Section 4. In fact, imposing the value of the yaw rate over time using the one provided by the guidance algorithm ($\dot{\psi}_c$), the model allows to verify the correct generation of the control command by checking the response of the system:

$$\dot{\psi}(t) = \dot{\psi}_c(t) \tag{51}$$

3.5.2. 4DOF Model

To improve the accuracy of the model, a possible further step is to add a degree of freedom consisting of roll angle ϕ by including translational dynamics in the form of Newton's second law (23). For the sake of simplicity, the case of negligible crosswind will be considered, but it is straightforward to generalize the formulation to the case with wind in any direction. Keeping the assumption of negligible sideslip gives now:

$$v \approx v_a \approx 0 \tag{52}$$

The equation (23), expressed in BF, thus becomes:

$$\frac{\boldsymbol{F}}{m} \approx \begin{bmatrix} \dot{u} \\ 0 \\ \dot{w} \end{bmatrix} + \boldsymbol{S}(\boldsymbol{\omega}) \begin{bmatrix} u \\ 0 \\ w \end{bmatrix} = \begin{bmatrix} \dot{u} \\ 0 \\ \dot{w} \end{bmatrix} + \begin{bmatrix} 0 & -r & q \\ r & 0 & -p \\ -q & p & 0 \end{bmatrix} \begin{bmatrix} u \\ 0 \\ w \end{bmatrix} = \begin{bmatrix} \dot{u} \\ 0 \\ \dot{w} \end{bmatrix} + \begin{bmatrix} qw \\ ru - pw \\ -qu \end{bmatrix} \tag{53}$$

It is now assumed that the pitch angle and pitch rate are negligible:

$$\theta \approx 0 \text{ and } \dot{\theta} \approx 0 \tag{54}$$

Under these assumptions, equation (5) becomes:

$$\begin{bmatrix} p \\ q \\ r \end{bmatrix} = \begin{bmatrix} 1 & 0 & -s_\theta \\ 0 & c_\phi & c_\theta\, s_\phi \\ 0 & -s_\phi & c_\theta\, c_\phi \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \end{bmatrix} \approx \begin{bmatrix} 1 & 0 & -\theta \\ 0 & c_\phi & s_\phi \\ 0 & -s_\phi & c_\phi \end{bmatrix} \begin{bmatrix} \dot{\phi} \\ 0 \\ \dot{\psi} \end{bmatrix} = \begin{bmatrix} \dot{\phi} - \dot{\psi}\theta \\ \dot{\psi}\, s_\phi \\ \dot{\psi}\, c_\phi \end{bmatrix} \approx \begin{bmatrix} \dot{\phi} \\ \dot{\psi}\, s_\phi \\ \dot{\psi}\, c_\phi \end{bmatrix} \tag{55}$$

This leads to rewriting (53) as follows:

$$\begin{bmatrix} \dot{u} \\ 0 \\ \dot{w} \end{bmatrix} = \frac{\mathbf{F}}{m} - \begin{bmatrix} qw \\ ru - pw \\ -qu \end{bmatrix} \approx \frac{\mathbf{F}}{m} - \begin{bmatrix} (s_\phi \dot{\psi})\ w \\ (c_\phi \dot{\psi})\ u - (\dot{\phi})w \\ -(s_\phi \dot{\psi})\ u \end{bmatrix} \tag{56}$$

Neglecting the apparent mass force, with the other assumptions given above, the vector of external forces in BF will be:

$$\boldsymbol{F} = \boldsymbol{F}_G + \boldsymbol{F}_A = \begin{bmatrix} s_\alpha L - c_\alpha D \\ 0 \\ -c_\alpha L \ - s_\alpha D \end{bmatrix} + m \begin{bmatrix} 0 \\ s_\phi g \\ c_\phi g \end{bmatrix} \tag{57}$$

Substituting the (57) into the (56) yields:

$$\begin{bmatrix} \dot{u} \\ 0 \\ \dot{w} \end{bmatrix} = \frac{1}{m} \begin{bmatrix} s_\alpha L - c_\alpha D \\ 0 \\ -c_\alpha L \ - s_\alpha D \end{bmatrix} + \begin{bmatrix} 0 \\ s_\phi g \\ c_\phi g \end{bmatrix} - \begin{bmatrix} s_\phi \dot{\psi} \ w \\ c_\phi \dot{\psi} \ u - \dot{\phi} \ w \\ -s_\phi \dot{\psi} \ u \end{bmatrix} = \begin{bmatrix} \dfrac{s_\alpha L - c_\alpha D}{m} - s_\phi \dot{\psi} \ w \\ s_\phi g - c_\phi \dot{\psi} \ u + \dot{\phi} \ w \\ -\dfrac{c_\alpha L + s_\alpha D}{m} + c_\phi g + s_\phi \dot{\psi} \ u \end{bmatrix} \tag{58}$$

and taking $\dot{\psi}$ to the left-hand side in the second equation:

$$\begin{bmatrix} \dot{u} \\ \dot{\psi} \\ \dot{w} \end{bmatrix} = \begin{bmatrix} (s_\alpha L - c_\alpha D)/m - s_\phi \dot{\psi} \ w \\ (s_\phi g + \dot{\phi} \ w)/u c_\phi \\ -(c_\alpha L + s_\alpha D)/m + c_\phi g + s_\phi \dot{\psi} \ u \end{bmatrix} \tag{59}$$

The following simplified models, which do not take into account the angle of attack α, can be employed for lift L and drag D [13] :

$$D = \frac{1}{2} \rho V_a^2 S_p (C_{D0} + C_{D\delta s} \delta_s) \neq f(\alpha) \tag{60}$$

$$L = \frac{1}{2} \rho V_a^2 S_p (C_{L0} + C_{L\delta s} \delta_s) \neq f(\alpha) \tag{61}$$

Here again, the model can be made more accurate by considering a first-order dynamic model not only for yaw rate $\dot{\psi}$ (as done in section 3.5.1), but also for roll angle ϕ, that is introducing [13]:

$$T_\psi \ddot{\psi} + \dot{\psi} = K_\psi \delta_a \tag{62}$$

$$T_\phi \dot{\phi} + \phi = K_\phi \delta_a \tag{63}$$

where $T_\psi, K_\psi, T_\phi, K_\phi$ are coefficients determined from flight data that take into account the aerodynamic behavior of the parafoil. In conclusion, equations (59) through (63) provide a 4DOF model that, compared with that of Section 3.5.2, takes into account translational dynamics and roll angle, in addition to yaw angle. The vehicle attitude description is provided by the simplified expressions (62) and (63) where the pitch angle is considered negligible.

3.5.3. 6DOF Model

A complete 6DOF model is now considered. The payload-parafoil system is assumed to be a rigid body (as shown in Figure 3-6) for which the relative motion between payload and parafoil is neglected. The starting point for the development of this model are the Newton-Euler dynamical equations (25). In this case, in addition to gravitational and aerodynamic forces and moments, effects related to apparent mass will also be included. The expressions introduced in Section 3.4.3 give the apparent mass forces and moments in Canopy Frame (CF). These contributions act in the apparent mass center $CoAM$ which generally does not coincide with the center of mass of the parafoil $ParCoM$. Henceforth, for simplicity, $CoAM$ and $ParCoM$ will be assumed to be coincident. Under this assumption, it is possible to express the apparent mass force (35) and apparent inertia moment (36) in BF [14]:

$$\boldsymbol{F}_{AM} = \boldsymbol{R}_{cb}^T \boldsymbol{F}_{AM}^C \tag{64}$$

$$\boldsymbol{M}_{AI} = \boldsymbol{R}_{cb}^T \boldsymbol{M}_{AI}^C + \boldsymbol{r}_{bm} \times \boldsymbol{F}_{AM} \tag{65}$$

where $\boldsymbol{r}_{bm}$ is the vector from the system center of mass $\mathrm{Sys}CoM$ to $CoAM$ and $\boldsymbol{R}_{\mathrm{cb}}$ is the rotation matrix from BF to CF. The latter represents a simple rotation in the plane of symmetry of the rigging angle μ (see Figure 3-6):

$$\boldsymbol{R}_{cb} = \begin{bmatrix} cos\mu & 0 & -sin\mu \\ 0 & 1 & 0 \\ sin\mu & 0 & cos\mu \end{bmatrix} \tag{66}$$

To develop equations (64) and (65) (expressed in BF), it is first needed to derive the relationships between linear and angular velocities and accelerations in CF and BF, recalling that the system is a rigid body [14]:

$$\begin{bmatrix} \hat{p}_c \\ \hat{q}_c \\ \hat{r}_c \end{bmatrix} = \boldsymbol{R}_{cb} \begin{bmatrix} p_c \\ q_c \\ r_c \end{bmatrix} = \boldsymbol{R}_{cb} \begin{bmatrix} p \\ q \\ \boldsymbol{r} \end{bmatrix} \tag{67}$$

$$\begin{bmatrix} \dot{\hat{p}}_c \\ \dot{\hat{q}}_c \\ \dot{\hat{r}}_c \end{bmatrix} = \boldsymbol{R}_{cb} \begin{bmatrix} \dot{p}_c \\ \dot{q}_c \\ \dot{r}_c \end{bmatrix} = \boldsymbol{R}_{cb} \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} \tag{68}$$

$$\begin{bmatrix} \hat{u}_{ac} \\ \hat{v}_{ac} \\ \hat{w}_{ac} \end{bmatrix} = \boldsymbol{R}_{cb} \left(\begin{bmatrix} u \\ v \\ w \end{bmatrix} + \boldsymbol{S}(\boldsymbol{\omega})\boldsymbol{r}_{bm} - \boldsymbol{R}_{cb}\boldsymbol{W} \right) = \boldsymbol{R}_{cb} \left(\begin{bmatrix} u \\ v \\ w \end{bmatrix} - \boldsymbol{S}(\boldsymbol{r}_{bm}) \begin{bmatrix} p \\ q \\ r \end{bmatrix} - \boldsymbol{R}_{cb}\boldsymbol{W} \right) \tag{69}$$

$$\begin{bmatrix} \dot{\hat{u}}_{ac} \\ \dot{\hat{v}}_{ac} \\ \dot{\hat{w}}_{ac} \end{bmatrix} = \boldsymbol{R}_{cb} \left(\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} + \boldsymbol{S}(\dot{\boldsymbol{\omega}})\ \boldsymbol{r}_{bm} \right) = \boldsymbol{R}_{cb} \left(\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} - \boldsymbol{S}(\boldsymbol{r}_{bm}) \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} \right) \tag{70}$$

where it is useful to recall the following two properties of the skew-symmetric matrices that can be applied to the case at hand:

$$\boldsymbol{\omega} \times \boldsymbol{r}_{bm} = \boldsymbol{S}(\boldsymbol{\omega})\boldsymbol{r}_{bm} = -\ \boldsymbol{S}(\boldsymbol{r}_{bm})\boldsymbol{\omega} \tag{71}$$

$$\boldsymbol{S}(\hat{\boldsymbol{\omega}}_c) = \boldsymbol{S}(\boldsymbol{R}_{cb}\ \hat{\boldsymbol{\omega}}_c) = \boldsymbol{S}(\boldsymbol{R}_{cb}\ \boldsymbol{\omega}) = \boldsymbol{R}_{cb}\ \boldsymbol{S}(\boldsymbol{\omega})\ \boldsymbol{R}_{cb}^T \tag{72}$$

Using the expressions from (67) to (72) allows to develop equations (64) and (65) highlighting velocity and acceleration variables with their components expressed in BF:

$$\begin{aligned} \boldsymbol{F}_{AM} = \boldsymbol{R}_{cb}^T \boldsymbol{F}_{AM}^C = -\boldsymbol{R}_{cb}^T &\left(\boldsymbol{I}_{AM} \begin{bmatrix} \dot{\hat{u}}_{ac} \\ \dot{\hat{v}}_{ac} \\ \dot{\hat{w}}_{ac} \end{bmatrix} + \boldsymbol{S}(\hat{\boldsymbol{\omega}}_c)\boldsymbol{I}_{AM} \begin{bmatrix} \hat{u}_{ac} \\ \hat{v}_{ac} \\ \hat{w}_{ac} \end{bmatrix} \right) \\ = -\boldsymbol{R}_{cb}^T &\left(\boldsymbol{I}_{AM}\boldsymbol{R}_{cb} \left(\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} - \boldsymbol{S}(\boldsymbol{r}_{bm}) \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} \right) \right. \\ &\left. + (\boldsymbol{R}_{cb}\boldsymbol{S}(\boldsymbol{\omega})\boldsymbol{R}_{cb}^T)\ \boldsymbol{I}_{AM}\boldsymbol{R}_{cb} \left(\begin{bmatrix} u \\ v \\ w \end{bmatrix} - \boldsymbol{S}(\boldsymbol{r}_{bm}) \begin{bmatrix} p \\ q \\ r \end{bmatrix} - \boldsymbol{R}_{cb}\boldsymbol{W} \right) \right) \\ = -\bar{\boldsymbol{I}}_{AM} &\begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} + \bar{\boldsymbol{I}}_{AM}\boldsymbol{S}(\boldsymbol{r}_{bm}) \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} - \boldsymbol{S}(\boldsymbol{\omega})\bar{\boldsymbol{I}}_{AM} \begin{bmatrix} u \\ v \\ w \end{bmatrix} \\ &+ \boldsymbol{S}(\boldsymbol{\omega})\bar{\boldsymbol{I}}_{AM}\boldsymbol{S}(\boldsymbol{r}_{bm}) \begin{bmatrix} p \\ q \\ r \end{bmatrix} + \boldsymbol{S}(\boldsymbol{\omega})\bar{\boldsymbol{I}}_{AM}\boldsymbol{R}_{cb}\boldsymbol{W} \end{aligned} \tag{73}$$

$$
\begin{aligned}
&\boldsymbol{M}_{AI} = \boldsymbol{R}_{cb}^{T}\boldsymbol{M}_{AI}^{C} + \boldsymbol{r}_{bm} \times \boldsymbol{F}_{AM} \\
&= \boldsymbol{R}_{cb}^{T}\left(\boldsymbol{I}_{AI}\begin{bmatrix}\dot{\hat{p}}_c\\ \dot{\hat{q}}_c\\ \dot{\hat{r}}_c\end{bmatrix} + \boldsymbol{S}(\hat{\boldsymbol{\omega}}_c)\boldsymbol{I}_{AI}\begin{bmatrix}\hat{p}_c\\ \hat{q}_c\\ \hat{r}_c\end{bmatrix}\right) + \boldsymbol{S}(\boldsymbol{r}_{bm})\boldsymbol{F}_{AM} \\
&= \boldsymbol{R}_{cb}^{T}\left(\boldsymbol{I}_{AI}\boldsymbol{R}_{cb}\begin{bmatrix}\dot{p}\\ \dot{q}\\ \dot{r}\end{bmatrix} + (\boldsymbol{R}_{cb}\boldsymbol{S}(\boldsymbol{\omega})\boldsymbol{R}_{cb}^{T})\ \boldsymbol{I}_{AI}\boldsymbol{R}_{cb}\begin{bmatrix}p\\ q\\ r\end{bmatrix}\right) \\
&+ \boldsymbol{S}(\boldsymbol{r}_{bm})\boldsymbol{F}_{AM} = \\
&= -\bar{\boldsymbol{I}}_{AI}\begin{bmatrix}\dot{p}\\ \dot{q}\\ \dot{r}\end{bmatrix} + \boldsymbol{S}(\boldsymbol{\omega})\bar{\boldsymbol{I}}_{AI}\begin{bmatrix}p\\ q\\ r\end{bmatrix} \\
&- \boldsymbol{S}(\boldsymbol{r}_{bm})\left(-\bar{\boldsymbol{I}}_{AM}\begin{bmatrix}\dot{u}\\ \dot{v}\\ \dot{w}\end{bmatrix} + \bar{\boldsymbol{I}}_{AM}\boldsymbol{S}(\boldsymbol{r}_{bm})\begin{bmatrix}\dot{p}\\ \dot{q}\\ \dot{r}\end{bmatrix} - \boldsymbol{S}(\boldsymbol{\omega})\bar{\boldsymbol{I}}_{AM}\begin{bmatrix}u\\ v\\ w\end{bmatrix}\right. \\
&\left. + \boldsymbol{S}(\boldsymbol{\omega})\bar{\boldsymbol{I}}_{AM}\boldsymbol{S}(\boldsymbol{r}_{bm})\begin{bmatrix}p\\ q\\ r\end{bmatrix} + \boldsymbol{S}(\boldsymbol{\omega})\bar{\boldsymbol{I}}_{AM}\boldsymbol{R}_{cb}\boldsymbol{W}\right)
\end{aligned}
\tag{74}
$$

where it was used:

$$\boldsymbol{R}_{cb}^{T}\boldsymbol{R}_{cb} = \boldsymbol{I}_{3x3} \tag{75}$$

$$\boldsymbol{R}_{cb}^{T}\boldsymbol{I}_{AM}\ \boldsymbol{R}_{cb} = \bar{\boldsymbol{I}}_{AM} \tag{76}$$

$$\boldsymbol{R}_{cb}^{T}\boldsymbol{I}_{AI}\ \boldsymbol{R}_{cb} = \bar{\boldsymbol{I}}_{AI} \tag{77}$$

In addition, for $\boldsymbol{M}_{AI}^{C}$, one of the terms included in the equation (36) has been neglected since it is assumed [14] that the effects associated with it, are contained within the aerodynamic momentum contribution $\boldsymbol{M}_A$:

$$\boldsymbol{S}(\hat{\boldsymbol{v}}_{ac})\boldsymbol{I}_{AM}\begin{bmatrix}\hat{u}_{ac}\\ \hat{v}_{ac}\\ \hat{w}_{ac}\end{bmatrix} \approx 0 \tag{78}$$

Keeping in mind that the total mass of the system includes the payload and parafoil mass and the enclosed mass (according to equation (33)), the Newton-Euler equations (25) are now rewritten considering all force and momentum contributions:

$$\begin{bmatrix}m\ \boldsymbol{I}_{3x3} & \boldsymbol{0}_{3x3}\\ \boldsymbol{0}_{3x3} & \boldsymbol{I_t}\end{bmatrix}\begin{bmatrix}\dot{u}\\ \dot{v}\\ \dot{w}\\ \dot{p}\\ \dot{q}\\ \dot{r}\end{bmatrix} = \begin{bmatrix}\boldsymbol{F} - m\ \boldsymbol{S}(\boldsymbol{\omega})\begin{bmatrix}u\\ v\\ w\end{bmatrix}\\ \boldsymbol{M} - \boldsymbol{S}(\boldsymbol{\omega})\boldsymbol{I_t}\begin{bmatrix}p\\ q\\ r\end{bmatrix}\end{bmatrix} = \begin{bmatrix}\boldsymbol{F}_G + \boldsymbol{F}_A + \boldsymbol{F}_{AM} - m\ \boldsymbol{S}(\boldsymbol{\omega})\begin{bmatrix}u\\ v\\ w\end{bmatrix}\\ \boldsymbol{M}_A + \boldsymbol{M}_{AI} - \boldsymbol{S}(\boldsymbol{\omega})\boldsymbol{I_t}\begin{bmatrix}p\\ q\\ r\end{bmatrix}\end{bmatrix} \tag{79}$$

By explicitly introducing the expressions (73) and (74) for $\boldsymbol{F}_{AM}$ and $\boldsymbol{M}_{AI}$, taking the terms that multiply $\dot{\boldsymbol{v}}$ and $\dot{\boldsymbol{\omega}}$ to the left-hand side of the equation, the following result is obtained:

$$\begin{bmatrix} m\ \boldsymbol{I}_{3x3} + \bar{\boldsymbol{I}}_{AM} & -\bar{\boldsymbol{I}}_{AM}\boldsymbol{S}(\boldsymbol{r}_{bm}) \\ \boldsymbol{S}(\boldsymbol{r}_{bm})\bar{\boldsymbol{I}}_{AM} & \boldsymbol{I}_t + \bar{\boldsymbol{I}}_{AI} - \boldsymbol{S}(\boldsymbol{r}_{bm})\bar{\boldsymbol{I}}_{AM}\boldsymbol{S}(\boldsymbol{r}_{bm}) \end{bmatrix} \begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \\ \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} \boldsymbol{P}_1 \\ \boldsymbol{P}_2 \end{bmatrix} \tag{80}$$

with:

$$\begin{aligned} \boldsymbol{P}_1 = \boldsymbol{F}_A + \boldsymbol{F}_G - \boldsymbol{S}(\boldsymbol{\omega})(m\boldsymbol{I}_{3x3} + \bar{\boldsymbol{I}}_{AM}) \begin{bmatrix} u \\ v \\ w \end{bmatrix} + \boldsymbol{S}(\boldsymbol{\omega})\bar{\boldsymbol{I}}_{AM}\boldsymbol{S}(\boldsymbol{r}_{bm}) \begin{bmatrix} p \\ q \\ r \end{bmatrix} \\ + \boldsymbol{S}(\boldsymbol{\omega})\bar{\boldsymbol{I}}_{AM}\boldsymbol{R}_{cb}\boldsymbol{W} \end{aligned} \tag{81}$$

$$\begin{aligned} \boldsymbol{P}_2 = \boldsymbol{M}_A - [\boldsymbol{S}(\boldsymbol{\omega})(\boldsymbol{I_t} + \bar{\boldsymbol{I}}_{AI}) - \boldsymbol{S}(\boldsymbol{r}_{bm})\boldsymbol{S}(\boldsymbol{\omega})\bar{\boldsymbol{I}}_{AM}\boldsymbol{S}(\boldsymbol{r}_{bm})] \begin{bmatrix} p \\ q \\ r \end{bmatrix} \\ - \boldsymbol{S}(\boldsymbol{r}_{bm})\boldsymbol{S}(\boldsymbol{\omega})\bar{\boldsymbol{I}}_{AM} \begin{bmatrix} u \\ v \\ w \end{bmatrix} + \boldsymbol{S}(\boldsymbol{r}_{bm})\boldsymbol{S}(\boldsymbol{\omega})\bar{\boldsymbol{I}}_{AM}\boldsymbol{R}_{cb}\boldsymbol{W} \end{aligned} \tag{82}$$

Equations (80) through (82) describe the translational and rotational dynamics of the 6DOF payload-parafoil model. Equations (41) and (6), which constitute the translational and rotational kinematics, complete the model. The model obtained, despite considering the system as a 6DOF rigid body, is a reasonably accurate model in describing the translational motion and attitude of the system also given the absence of highly restrictive assumptions. With its implementation in a Simulink environment, for instance, it is possible to test complete GNC architectures applied to the flight under parafoils with good accuracy. For even more accurate results, additional degrees of freedom can be introduced in order to also describe the relative motion between parafoil and payload. Among the most widely used is the 9DOF model. The latter allows for relative free rotation between parafoil and payload while taking into account the couplings given by the torques generated by the lines connecting the two elements. A full description of these kinds of models is beyond the scope of this work. Please refer to [15] for a detailed discussion of this topic.

3.6. Linearized Models

All models analyzed in Section 3.5 constitute nonlinear models of the parafoil-payload system. In some cases, such as for stability analysis or control purposes, the need arises to develop linearized models. For the intent of this study, this need is related to the use of Model Predictive Control (MPC) techniques for control sequence generation. As a matter of fact, for the use of MPC, it is required to provide the linearized system in the continuous-time state-space form:

$$\begin{cases} \dot{x} = Ax + Bu \\ y = Cx + Du \end{cases} \tag{83}$$

where x is the state vector, y the output vector, u the control vector, and A, B, C, D are respectively the state matrix, the input matrix, the output matrix, and the feedforward matrix. This system will then be converted to discrete time as discussed in Section 5.2. The guidance strategy proposed in this work requires in particular a SISO-type system with the value of yaw angle as the output and the value of asymmetric deflection as the control variable. In Section 3.6.1, an analytical linearization of a simplified model is proposed. The resulting model, however, is adequate in terms of accuracy for the purposes of verification of guidance algorithms. A more accurate 6DOF linearized model is then briefly described in Section 3.6.2.

3.6.1. Reduced-order Linearized Model

In this section, a simplified analytical approach to obtaining a linearized model of the parafoil-payload system is presented. This model is based on the rotational kinematics and dynamics equations taking into account the following assumptions:

- Slow turn rate: negligible sideslip and roll angles
- No vertical wind
- Constant horizontal airspeed norm
- Constant vertical airspeed norm
- No apparent mass effects
- No enclosed mas effects

The hypotheses above translate into the following expressions:

$$\beta \approx 0, \phi \approx 0 \;\rightarrow\; \chi_a \approx \psi \tag{84}$$

$$W_z \approx 0 \tag{85}$$

$$V_z \approx \mathrm{V}_{va} \approx \mathrm{V}_{va0} \quad \text{with} \quad k_{\dot{\psi}} \approx 0 \tag{86}$$

$$V_{ha} \approx V_{ha0} \tag{87}$$

$$\boldsymbol{F}_{AM}, \boldsymbol{M}_{AI} \approx 0 \tag{88}$$

$$m_{enclosed} \approx 0 \tag{89}$$

The chosen state variables are the angular values and rates describing the orientation of BF with respect to LV except for pitch angle and pitch rate. The output of the system is the yaw angle ψ, and the control variable is asymmetric deflection $\bar{\delta_a}$. The objective of the linearization process is then to derive the A and B matrices of the following system in state-space form (written in terms of variations):

$$\begin{cases} \begin{bmatrix} \delta\dot{\phi} \\ \delta\dot{\psi} \\ \delta\dot{p} \\ \delta\dot{r} \end{bmatrix} = \begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} \\ A_{21} & A_{22} & A_{23} & A_{24} \\ A_{31} & A_{32} & A_{33} & A_{34} \\ A_{41} & A_{42} & A_{43} & A_{44} \end{bmatrix} \begin{bmatrix} \delta\phi \\ \delta\psi \\ \delta p \\ \delta r \end{bmatrix} + \begin{bmatrix} B_1 \\ B_2 \\ B_3 \\ B_4 \end{bmatrix} \delta\bar{\delta}_a \\ \delta\psi = [0, 1, 0, 0] \begin{bmatrix} \delta\phi \\ \delta\psi \\ \delta p \\ \delta r \end{bmatrix} \end{cases} \tag{90}$$

The approach provided below is as described in [10]. It is a first-order linearization about the steady-state with zero roll angle, angular rates, asymmetric deflection, and pitch angle fixed:

$$s_0 = [\phi_0, \theta_0, p_0, q_0, r_0, \delta_{a0}]^T = [0, \theta_0, 0{,}0{,}0{,}0]^T \tag{91}$$

The starting equations to obtain the linear model are the following:

a. Rotational kinematics: equation (6)
b. Rotational dynamics: rotational part of equation (80)

a. Analytical linearization of the kinematics equations

The first step is to modify the starting kinematic equation by adding terms of infinitesimal variation to all the elements involved:

$$\begin{bmatrix}\dot{\phi}\\ \dot{\theta}\\ \dot{\psi}\end{bmatrix}=\begin{bmatrix}1 & sin\phi\ tan\theta & cos\phi\ tan\theta\\ 0 & cos\phi & -sin\phi\\ 0 & sin\phi/cos\theta & cos\phi/cos\theta\end{bmatrix}\begin{bmatrix}p\\ q\\ r\end{bmatrix}=\boldsymbol{\Phi}\begin{bmatrix}p\\ q\\ r\end{bmatrix} \tag{92}$$

$$\begin{bmatrix}\dot{\phi}+\delta\dot{\phi}\\ \dot{\theta}+\delta\dot{\theta}\\ \dot{\psi}+\delta\dot{\psi}\end{bmatrix}=\boldsymbol{\Phi}^{\delta}\begin{bmatrix}p+\delta p\\ q+\delta q\\ r+\delta r\end{bmatrix}$$

$$=\begin{bmatrix}1 & sin\ (\phi+\delta\phi)tan\ (\theta+\delta\theta) & cos\ (\phi+\delta\phi)tan\ (\theta+\delta\theta)\\ 0 & cos\ (\phi+\delta\phi) & -sin\ (\phi+\delta\phi)\\ 0 & sin\ (\phi+\delta\phi)/cos\ (\theta+\delta\theta) & cos\ (\phi+\delta\phi)/cos\ (\theta+\delta\theta)\end{bmatrix}\begin{bmatrix}p+\delta p\\ q+\delta q\\ r+\delta r\end{bmatrix} \tag{93}$$

Subtracting (92) to (93) gives:

$$\begin{bmatrix}\delta\dot{\phi}\\ \delta\dot{\theta}\\ \delta\dot{\psi}\end{bmatrix}=\boldsymbol{\Phi}^{\delta}\begin{bmatrix}p+\delta p\\ q+\delta q\\ r+\delta r\end{bmatrix}-\boldsymbol{\Phi}\begin{bmatrix}p\\ q\\ r\end{bmatrix}=\ (\boldsymbol{\Phi}^{\delta}-\boldsymbol{\Phi})\begin{bmatrix}p\\ q\\ r\end{bmatrix}+\boldsymbol{\Phi}^{\delta}\begin{bmatrix}\delta p\\ \delta q\\ \delta r\end{bmatrix} \tag{94}$$

It may be useful to recall that for trigonometric functions it is possible to treat infinitesimal variations as follows:

$$sin(\varphi_0+\delta\varphi)=sin\varphi_0\ cos\delta\varphi+sin\delta\varphi\ cos\varphi_0\approx sin\varphi_0+\delta\varphi\ cos\varphi_0 \tag{95}$$

$$cos(\varphi_0+\delta\varphi)=cos\varphi_0\ cos\delta\varphi+sin\delta\varphi\ sin\varphi_0\approx cos\varphi_0+\delta\varphi\ sin\varphi_0 \tag{96}$$

$$tan(\varphi_0+\delta\varphi)=\frac{tan\varphi_0+tan\delta\varphi}{1-tan\varphi_0 tan\delta\varphi}\approx\frac{tan\varphi_0+\delta\varphi}{1-tan\varphi_0\ \delta\varphi} \tag{97}$$

with:

$$sin\,\delta\varphi\approx\delta\varphi,\ \ cos\,\delta\varphi\approx 1,\ \ tan\,\delta\varphi\ \approx\delta\varphi\quad \text{for}\quad \delta\varphi\to 0 \tag{98}$$

Taking into account what above, it is possible to evaluate the matrix $\boldsymbol{\Phi}^{\delta}$ at the reference steady-state s_0:

$$\begin{aligned}
\boldsymbol{\Phi}^{\delta}(s_0) &= \begin{bmatrix} 1 & sin(\phi_0+\delta\phi)\,tan(\theta_0+\delta\theta) & cos(\phi_0+\delta\phi)\,tan(\theta_0+\delta\theta) \\ 0 & cos(\phi_0+\delta\phi) & -\,sin(\phi_0+\delta\phi) \\ 0 & \dfrac{sin(\phi_0+\delta\phi)}{cos(\theta_0+\delta\theta)} & \dfrac{cos(\phi_0+\delta\phi)}{cos(\theta_0+\delta\theta)} \end{bmatrix} \\
&= \begin{bmatrix} 1 & sin(\delta\phi)\,tan(\theta_0+\delta\theta) & cos(\delta\phi)\,tan(\theta_0+\delta\theta) \\ 0 & cos(\delta\phi) & -\,sin(\delta\phi) \\ 0 & \dfrac{sin(\delta\phi)}{cos(\theta_0+\delta\theta)} & \dfrac{cos(\delta\phi)}{cos(\theta_0+\delta\theta)} \end{bmatrix} \\
&\approx \begin{bmatrix} 1 & \delta\phi\ tan(\theta_0+\delta\theta) & tan\ (\theta_0+\delta\theta) \\ 0 & 1 & -\delta\phi \\ 0 & \delta\phi/cos\ (\theta_0+\delta\theta) & 1/cos\ (\theta_0+\delta\theta) \end{bmatrix} \\
&\approx \begin{bmatrix} 1 & \delta\phi\ \dfrac{tan\theta_0+\delta\theta}{1-tan\theta_0\ \delta\theta} & \dfrac{tan\theta_0+\delta\theta}{1-tan\theta_0\ \delta\theta} \\ 0 & 1 & -\delta\phi \\ 0 & \delta\phi/(cos\theta_0+\delta\theta\ sin\theta_0) & 1/(cos\theta_0+\delta\theta\ sin\varphi_0) \end{bmatrix}
\end{aligned} \tag{99}$$

Hence, neglecting the second-order or higher terms, eq. (95) becomes:

$$\begin{aligned}
\begin{bmatrix} \delta\dot{\phi} \\ \delta\dot{\theta} \\ \delta\dot{\psi} \end{bmatrix} &= (\boldsymbol{\Phi}^{\delta}-\boldsymbol{\Phi}) \begin{bmatrix} p_0 \\ q_0 \\ r_0 \end{bmatrix} + \boldsymbol{\Phi}^{\delta} \begin{bmatrix} \delta p \\ \delta q \\ \delta r \end{bmatrix} = \boldsymbol{\Phi}^{\delta} \begin{bmatrix} \delta p \\ \delta q \\ \delta r \end{bmatrix} \\
&\approx \begin{bmatrix} 1 & \delta\phi\ \dfrac{tan\theta_0+\delta\theta}{1-tan\theta_0\ \delta\theta} & \dfrac{tan\theta_0+\delta\theta}{1-tan\theta_0\ \delta\theta} \\ 0 & 1 & -\delta\phi \\ 0 & \delta\phi/(cos\theta_0+\delta\theta\ sin\theta_0) & 1/(cos\theta_0+\delta\theta\ sin\varphi_0) \end{bmatrix} \begin{bmatrix} \delta p \\ \delta q \\ \delta r \end{bmatrix} \\
&\approx \begin{bmatrix} 1 & 0 & tan\theta_0 \\ 0 & 1 & 0 \\ 0 & 0 & 1/cos\theta_0 \end{bmatrix} \begin{bmatrix} \delta p \\ \delta q \\ \delta r \end{bmatrix}
\end{aligned} \tag{100}$$

From the second equation, it is noted that pitch motion depends only on the angular rate q and is decoupled from yaw and roll motions. This justifies the use of only ϕ and ψ variables for the reduced model under consideration. In particular, under the hypotheses taken, the variation of pitch rate $\delta\dot{\theta}$ corresponds with that of the angular rate δq:

$$\delta\dot{\theta} = \delta q \tag{101}$$

b. Analytical linearization of the dynamic equations

Here again, the starting point is adding infinitesimal variation terms to the original rotational dynamic equation:

$$\begin{aligned} &\boldsymbol{I}_t + \bar{\boldsymbol{I}}_{AI} - \boldsymbol{S}(\boldsymbol{r}_{bm})\bar{\boldsymbol{I}}_{AM}\boldsymbol{S}(\boldsymbol{r}_{bm}) \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = \boldsymbol{J} \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} \\ &= \boldsymbol{M}_A - [\boldsymbol{S}(\boldsymbol{\omega})(\boldsymbol{I_t} + \boldsymbol{I}'_{a.i.}) - \boldsymbol{S}(\boldsymbol{r}_{BM})\boldsymbol{S}(\boldsymbol{\omega})\boldsymbol{I}'_{a.m.}\boldsymbol{S}(\boldsymbol{r}_{BM})] \begin{bmatrix} p \\ q \\ r \end{bmatrix} \\ &- \boldsymbol{S}(\boldsymbol{r}_{BM})\boldsymbol{S}(\boldsymbol{\omega})\boldsymbol{I}'_{a.m.} \left(\begin{bmatrix} u \\ v \\ w \end{bmatrix} - \boldsymbol{R}_\mu \boldsymbol{W} \right) \end{aligned} \tag{102}$$

$$\begin{aligned} &\boldsymbol{I}_t + \bar{\boldsymbol{I}}_{AI} - \boldsymbol{S}(\boldsymbol{r}_{bm})\bar{\boldsymbol{I}}_{AM}\boldsymbol{S}(\boldsymbol{r}_{bm}) \begin{bmatrix} \dot{p} + \delta\dot{p} \\ \dot{q} + \delta\dot{q} \\ \dot{r} + \delta\dot{r} \end{bmatrix} = \boldsymbol{J} \begin{bmatrix} \dot{p} + \delta\dot{p} \\ \dot{q} + \delta\dot{q} \\ \dot{r} + \delta\dot{r} \end{bmatrix} \\ &= \boldsymbol{M}_A^\delta - [\boldsymbol{S}(\boldsymbol{\omega} + \delta\boldsymbol{\omega})(\boldsymbol{I_t} + \boldsymbol{I}'_{a.i.}) - \boldsymbol{S}(\boldsymbol{r}_{BM})\boldsymbol{S}(\boldsymbol{\omega} + \delta\boldsymbol{\omega})\boldsymbol{I}'_{a.m.}\boldsymbol{S}(\boldsymbol{r}_{BM})] \begin{bmatrix} p + \delta p \\ q + \delta q \\ r + \delta r \end{bmatrix} \\ &- \boldsymbol{S}(\boldsymbol{r}_{BM})\boldsymbol{S}(\boldsymbol{\omega} + \delta\boldsymbol{\omega})\boldsymbol{I}'_{a.m.} \left(\begin{bmatrix} u \\ v \\ w \end{bmatrix} - \boldsymbol{R}_\mu \boldsymbol{W} \right) \end{aligned} \tag{103}$$

where the following notation was used:

$$\boldsymbol{J} = \boldsymbol{I}_t + \bar{\boldsymbol{I}}_{AI} - \boldsymbol{S}(\boldsymbol{r}_{bm})\bar{\boldsymbol{I}}_{AM}\boldsymbol{S}(\boldsymbol{r}_{bm}) \tag{104}$$

The aerodynamic moment vectors $\boldsymbol{M}_A$and $\mathbf{M}_A^\delta$, considering null angle of sideslip β, are given by the following expressions:

$$\begin{aligned} \boldsymbol{M}_A &= \frac{\rho V_a^2 S_p}{2} \begin{bmatrix} b\left(C_{l\beta}\beta + \frac{b}{2V_a}C_{lp}p + \frac{b}{2V_a}C_{lr}r + C_{l\delta_a}\bar{\delta}_a\right) \\ \bar{c}\left(C_{m0} + C_{m\alpha}\alpha + \frac{c}{2V_a}C_{mq}q\right) \\ b\left(C_{n\beta}\beta + \frac{b}{2V_a}C_{np}p + \frac{b}{2V_a}C_{nr}r + C_{n\delta_a}\bar{\delta}_a\right) \end{bmatrix} \\ &= QS_p \begin{bmatrix} b\left(\frac{b}{2V_a}C_{lp}p + \frac{b}{2V_a}C_{lr}r + C_{l\delta_a}\bar{\delta}_a\right) \\ \bar{c}\left(C_{m0} + C_{m\alpha}\alpha + \frac{c}{2V_a}C_{mq}q\right) \\ b\left(\frac{b}{2V_a}C_{np}p + \frac{b}{2V_a}C_{nr}r + C_{n\delta_a}\bar{\delta}_a\right) \end{bmatrix} \end{aligned} \tag{105}$$

$$\boldsymbol{M}_A^\delta = QS_p \begin{bmatrix} b\left(\dfrac{b}{2V_a}C_{lp}(p+\delta p)+\dfrac{b}{2V_a}C_{lr}(r+\delta r)+C_{l\delta_a}(\bar{\delta}_a+\delta\bar{\delta}_a)\right) \\ \bar{c}\left(C_{m0}+C_{m\alpha}\alpha+\dfrac{c}{2V_a}C_{mq}(q+\delta q)\right) \\ b\left(\dfrac{b}{2V_a}C_{np}(p+\delta p)+\dfrac{b}{2V_a}C_{nr}(r+\delta r)+C_{n\delta_a}(\bar{\delta}_a+\delta\bar{\delta}_a)\right) \end{bmatrix} \tag{106}$$

with the following notation for dynamic pressure:

$$Q = \frac{1}{2}\rho V_a^2 \tag{107}$$

The skew-symmetric matrix $\boldsymbol{S}$ applied to vector $(\boldsymbol{\omega}+\delta\boldsymbol{\omega})$, on the other hand, results as follows:

$$\boldsymbol{S}(\boldsymbol{\omega}+\delta\boldsymbol{\omega}) = \begin{bmatrix} 0 & -(r+\delta r) & (q+\delta q) \\ (r+\delta r) & 0 & -(p+\delta p) \\ -(q+\delta q) & (p+\delta p) & 0 \end{bmatrix} \tag{108}$$

Subtracting (102) to (103), neglecting the second-order terms, gives:

$$\boldsymbol{J}\begin{bmatrix}\delta\dot{p}\\ \delta\dot{q}\\ \delta\dot{r}\end{bmatrix} = \boldsymbol{J}\begin{bmatrix}\dot{p}+\delta\dot{p}\\ \dot{q}+\delta\dot{q}\\ \dot{r}+\delta\dot{r}\end{bmatrix} - \boldsymbol{J}\begin{bmatrix}\dot{p}\\ \dot{q}\\ \dot{r}\end{bmatrix} \approx \boldsymbol{M}_A^\delta - \boldsymbol{M}_A = \delta\boldsymbol{M}_A \tag{109}$$

where:

$$\begin{aligned} \delta\boldsymbol{M}_A &= \boldsymbol{M}_A - \boldsymbol{M}_A^\delta \\ &= QS_p \begin{bmatrix} b\left(\dfrac{b}{2V_a}C_{lp}\delta p+\dfrac{b}{2V_a}C_{lr}\delta r+C_{l\delta_a}\delta\bar{\delta}_a\right) \\ \bar{c}\left(C_{m0}+C_{m\alpha}\alpha+\dfrac{c}{2V_a}C_{mq}\delta q\right) \\ b\left(\dfrac{b}{2V_a}C_{np}\delta p+\dfrac{b}{2V_a}C_{nr}\delta r+C_{n\delta_a}\delta\bar{\delta}_a\right) \end{bmatrix} \\ &= \begin{bmatrix} \dfrac{\rho S_p b^2 V_a}{4}C_{lp}\delta p+\dfrac{\rho S_p b^2 V_a}{4}C_{lr}\delta r+QS_p b\ C_{l\delta_a}\delta\bar{\delta}_a \\ QS_p\bar{c}\left(C_{m0}+C_{m\alpha}\alpha+\dfrac{c}{2V_a}C_{mq}\delta q\right) \\ \dfrac{\rho S_p b^2 V_a}{4}C_{np}\delta p+\dfrac{\rho S_p b^2 V_a}{4}C_{nr}\delta r+QS_p b\ C_{n\delta_a}\delta\bar{\delta}_a \end{bmatrix} \end{aligned} \tag{110}$$

The equation (109) then becomes:

$$\begin{bmatrix} \delta\dot{p} \\ \delta\dot{q} \\ \delta\dot{r} \end{bmatrix} = \boldsymbol{J}^{-1}\delta\boldsymbol{M}_A = \begin{bmatrix} B_1 \\ B_2 \\ B_3 \end{bmatrix}$$

$$= \begin{bmatrix} J^i_{11} & 0 & J^i_{13} \\ 0 & J^i_{22} & 0 \\ J^i_{31} & 0 & J^i_{33} \end{bmatrix} \begin{bmatrix} \frac{\rho S_p b^2 V_a}{4} C_{lp}\delta p + \frac{\rho S_p b^2 V_a}{4} C_{lr}\delta r + QS_p b\ C_{l\delta_a}\delta\bar{\delta}_a \\ QS_p\bar{c}\left(C_{m0} + C_{m\alpha}\alpha + \frac{c}{2V_a}C_{mq}\delta q\right) \\ \frac{\rho S_p b^2 V_a}{4} C_{np}\delta p + \frac{\rho S_p b^2 V_a}{4} C_{nr}\delta r + QS_p b\ C_{n\delta_a}\delta\bar{\delta}_a \end{bmatrix} \tag{111}$$

with:

$$\boldsymbol{J}^{-1} = inv\left(\begin{bmatrix} J_{11} & 0 & J_{13} \\ 0 & J_{22} & 0 \\ J_{13} & 0 & J_{33} \end{bmatrix}\right) = \begin{bmatrix} J^i_{11} & 0 & J^i_{13} \\ 0 & J^i_{22} & 0 \\ J^i_{31} & 0 & J^i_{33} \end{bmatrix} \tag{112}$$

and

$$\begin{aligned} B_1 = J^i_{11}&\left(\frac{\rho S_p b^2 V_a}{4} C_{lp}\delta p + \frac{\rho S_p b^2 V_a}{4} C_{lr}\delta r + QS_p b\ C_{l\delta_a}\delta\bar{\delta}_a\right) \\ &+ J^i_{13}\left(\frac{\rho S_p b^2 V_a}{4} C_{np}\delta p + \frac{\rho S_p b^2 V_a}{4} C_{nr}\delta r + QS_p b\ C_{n\delta_a}\delta\bar{\delta}_a\right) \end{aligned} \tag{113}$$

$$B_2 = J^i_{22} QS_p\bar{c}\left(C_{m0} + C_{m\alpha}\alpha + \frac{c}{2V_a}C_{mq}\delta q\right) \tag{114}$$

$$\begin{aligned} B_3 = J^i_{31}&\left(\frac{\rho S_p b^2 V_a}{4} C_{lp}\delta p + \frac{\rho S_p b^2 V_a}{4} C_{lr}\delta r + QS_p b\ C_{l\delta_a}\delta\bar{\delta}_a\right) \\ &+ J^i_{33}\left(\frac{\rho S_p b^2 V_a}{4} C_{np}\delta p + \frac{\rho S_p b^2 V_a}{4} C_{nr}\delta r + QS_p b\ C_{n\delta_a}\delta\bar{\delta}_a\right) \end{aligned} \tag{115}$$

Gathering with respect to the variables in the state vector and the control variable yields the following system:

$$\begin{bmatrix} \delta\dot{p} \\ \delta\dot{q} \\ \delta\dot{r} \end{bmatrix} = \boldsymbol{J^{-1}}\delta\boldsymbol{M}_A \tag{116}$$

$$= S_p \begin{bmatrix} J_{11}^i \frac{\rho b^2 V_a}{4} C_{lp} + J_{13}^i \frac{\rho b^2 V_a}{4} C_{np} & 0 & J_{11}^i \frac{\rho b^2 V_a}{4} C_{lr} + J_{13}^i \frac{\rho b^2 V_a}{4} C_{nr} \\ 0 & J_{22}^i Q \bar{c} \frac{c}{2V_a} C_{mq} & 0 \\ J_{31}^i \frac{\rho b^2 V_a}{4} C_{lp} + J_{33}^i \frac{\rho b^2 V_a}{4} C_{np} & 0 & J_{31}^i \frac{\rho b^2 V_a}{4} C_{lr} + J_{33}^i \frac{\rho b^2 V_a}{4} C_{nr} \end{bmatrix} \begin{bmatrix} \delta p \\ \delta q \\ \delta r \end{bmatrix}$$

$$+ \begin{bmatrix} J_{11}^i S_p Q b \; C_{l\delta_a} + J_{13}^i S_p Q b \; C_{n\delta_a} \\ 0 \\ J_{31}^i S_p Q b \; C_{l\delta_a} + J_{33}^i S_p Q b \; C_{n\delta_a} \end{bmatrix} \delta\bar{\delta}_a + \begin{bmatrix} 0 \\ S_p Q \bar{c} J_{22}^i (C_{m0} + C_{m\alpha}\alpha) \\ 0 \end{bmatrix} \approx$$

$$\approx S_p \begin{bmatrix} J_{11}^i \frac{\rho b^2 V_a}{4} C_{lp} & 0 & J_{13}^i \frac{\rho b^2 V_a}{4} C_{nr} \\ 0 & J_{22}^i Q \bar{c} \frac{c}{2V_a} C_{mq} & 0 \\ J_{31}^i \frac{\rho b^2 V_a}{4} C_{lp} & 0 & J_{33}^i \frac{\rho b^2 V_a}{4} C_{nr} \end{bmatrix} \begin{bmatrix} \delta p \\ \delta q \\ \delta r \end{bmatrix}$$

$$+ S_p Q b \begin{bmatrix} (J_{11}^i C_{l\delta_a} + J_{13}^i C_{n\delta_a}) \\ 0 \\ (J_{31}^i C_{l\delta_a} + J_{33}^i C_{n\delta_a}) \end{bmatrix} \delta\bar{\delta}_a + S_p Q \bar{c} \begin{bmatrix} 0 \\ J_{22}^i (C_{m0} + C_{m\alpha}\alpha) \\ 0 \end{bmatrix}$$

Finally, it is possible to combine the two expressions (100) and (116) by extracting only the equations involving the state variables (in both cases the first and third):

$$\begin{bmatrix} \delta\dot{\phi} \\ \delta\dot{\psi} \\ \delta\dot{p} \\ \delta\dot{r} \end{bmatrix} = \begin{bmatrix} 0 & 0 & 1 & tan\theta_0 \\ 0 & 0 & 0 & 1/cos\theta_0 \\ 0 & 0 & J_{11}^i \bar{K} C_{lp} & J_{13}^i \bar{K} C_{nr} \\ 0 & 0 & J_{31}^i \bar{K} C_{lp} & J_{33}^i \bar{K} C_{nr} \end{bmatrix} \begin{bmatrix} \delta\phi \\ \delta\psi \\ \delta p \\ \delta r \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ Q S_p b \; (J_{11}^i C_{l\delta_a} + J_{13}^i C_{n\delta_a}) \\ Q S_p b \; (J_{31}^i C_{l\delta_a} + J_{33}^i C_{n\delta_a}) \end{bmatrix} \delta\bar{\delta}_a \tag{117}$$

with:

$$\bar{K} = \frac{\rho S_p b^2 V_a}{4} \tag{118}$$

The linearized system in the form (90) was obtained. In [16], the authors propose the introduction of two additional terms in A to remove the position of the center of pressure from the dynamic equations while maintaining the tendency to glide without roll during neutral control:

$$A = \begin{bmatrix} 0 & 0 & 1 & tan\theta_0 \\ 0 & 0 & 0 & 1/cos\theta_0 \\ Q S_p b \; J_{11}^i C_{l\phi} & 0 & J_{11}^i \bar{K} C_{lp} & J_{13}^i \bar{K} C_{nr} \\ Q S_p b \; J_{31}^i C_{l\phi} & 0 & J_{31}^i \bar{K} C_{lp} & J_{33}^i \bar{K} C_{nr} \end{bmatrix} \tag{119}$$

To verify the accuracy of the linearized model obtained in this section with an analytical method, a numerical linearization was performed by means of the tools available within the MATLAB/Simulink environment. For this purpose, a Simulink model (see Figure 3-8) was developed in order to implement the complete nonlinear rotational equations of kinematics and dynamics (equations (5) and (80)). Through MATLAB *linmod* function it is possible to obtain the continuous-time linear state-space model from the nonlinear Simulink model (in this case called 'NonlinSys') by indicating the operation point in the form of state vector X_0 and input vector U_0. The function call within the MATLAB script is as follows:

```
[A,B,C,D] = linmod('NonlinSys',X0,U0);
```
(120)

In the current case study, for X_0 and U_0 values, the steady-state vector described by (91) was considered. In order to verify the analytical linearization, it was naturally required to consider a subset of the rotational variables shown in Figure 3-8, that is, those listed among the state variables, namely ϕ, ψ, p, r.

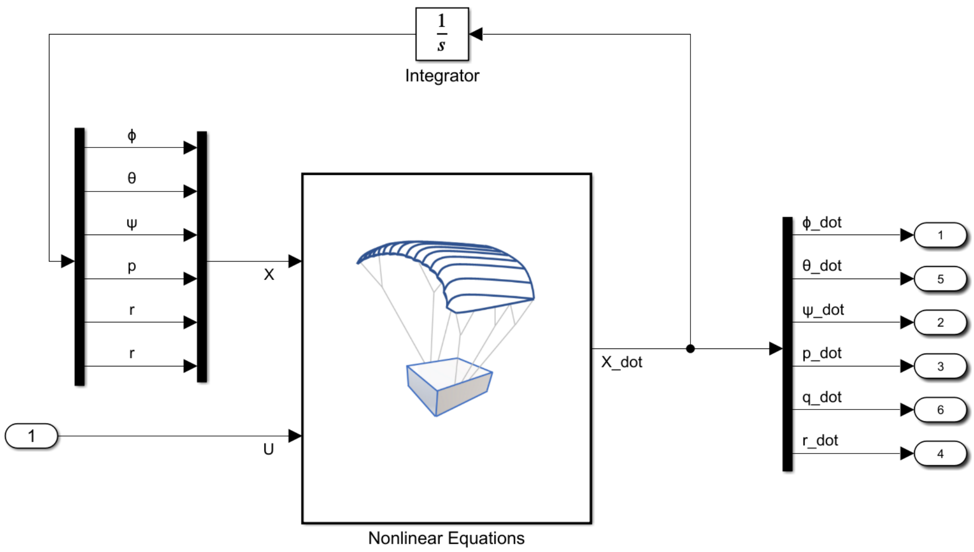

Figure 3-8: Simulink NonlinSys model for numerical linearization

A MATLAB script was finally developed to derive the numerical values of the A and B matrices of the analytically linearized model to compare them with those given by the *linmod* function. For an initial comparative check between the analytical and numerical models, the geometric and aerodynamic parameters of the payload-parafoil system described in [1] were used. The results obtained show a good match between the two different models. Shown as examples, in the table are the A matrices obtained according to the procedure described above.

Model	Matrix
Analytical model (matrix A)	$A = \begin{bmatrix} 0 & 0 & 1 & 0.08748866 \\ 0 & 0 & 0 & 1.00381983 \\ -0.09382090 & 0 & -0.75063026 & -0.16528320 \\ 0.06603074 & 0 & 0.51421441 & -1.95293961 \end{bmatrix}$
Numerical model (matrix A)	$A = \begin{bmatrix} 0 & 0 & 1 & 0.08748867 \\ 0 & 0 & 0 & 1.00381983 \\ -0.09382090 & 0 & -0.74998178 & -0.16734532 \\ 0.06603075 & 0 & 0.52141116 & -1.94949187 \end{bmatrix}$
Relative abs error (%)	$E = \begin{bmatrix} 0 & 0 & 1 & 0.000011 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0.086391 & 1.247628 \\ 0.000015 & 0 & 1.399562 & 0.176541 \end{bmatrix}$

Table 5: Analytical and numerical linearization comparison

For a more in-depth analysis of the accuracy of the two models, a test campaign within a complete 6DOF simulator is required to compare the performance of the two linearized models with that of the corresponding nonlinear one when considering an open-loop maneuver around the operation point at issue (steady-state vector). Such a study is beyond the scope of this work. It is also for this reason that the linearized 6DOF model shown in the next section was chosen to be used (primarily) for the algorithms illustrated in this thesis. In fact, this model has undergone an appropriate validation campaign that makes it reliable for the purposes of this work. It is noted that the numerical linearization method outlined above can also be used directly for generating the linear state-space model. This procedure is particularly useful in cases where analytical linearization is particularly complex or not applicable. An appropriate model validation campaign proves necessary in this case as well.

3.6.2. 6DOF Linearized Model

A 6DOF continuous-time linear model courtesy of SENER Aeroespacial was employed to apply the Model Predictive Control techniques described in Section 5. This model was subjected to a specific validation campaign carried out by SENER Aeroespacial to verify its reliability. The state vector, in this case, consists of nine variables, viz.:

- Attitude Euler angles of BF with respect to LV
- Angular rates of BF with respect to LV in BF
- Airspeed components in BF

On the other hand, the control vector is composed of the following two control variables:

- Symmetric deflection
- Asymmetric deflection

The operational point considered to obtain the model at hand is the one that characterizes the trim condition for the Terminal Guidance phase with constant symmetric deflection. The 6DOF state-space linear model will be of the form:

$$\begin{cases} \begin{bmatrix} \dot{\phi} \\ \dot{\theta} \\ \dot{\psi} \\ \dot{p} \\ \dot{q} \\ \dot{r} \\ \ddot{u}_a \\ \ddot{v}_a \\ \ddot{w}_a \end{bmatrix} = \boldsymbol{A}_{9x9} \begin{bmatrix} \phi \\ \theta \\ \psi \\ p \\ q \\ r \\ \dot{u}_a \\ \dot{v}_a \\ \dot{w}_a \end{bmatrix} + \boldsymbol{B}_{9X2} \begin{bmatrix} \delta_s \\ \delta_a \end{bmatrix} \\ \psi = [0,0,1,0,0,0,0,0,0] \begin{bmatrix} \phi \\ \theta \\ \psi \\ p \\ q \\ r \\ \dot{u}_a \\ \dot{v}_a \\ \dot{w}_a \end{bmatrix} \end{cases} \tag{121}$$

4. Optimal Trajectory Design

This chapter describes the guidance algorithm selected to generate the optimal trajectory to be employed during the Terminal Guidance phase. The guidance problem to be solved is the so-called Two-point Boundary Value Problem (TPBVP) for the solution of which a vast range of possible approaches are proposed in the literature. Some of the main solutions are discussed in Section 4.2. The core algorithm chosen is the one described by Slegers and Yakimenko in [1]. Several changes were then introduced to adapt it to the case at hand.

4.1. Two-Point Boundary Value Problem

A Boundary Value Problem consists of a system of Ordinary Differential Equations (ODE) with values of the solution and derivative specified at a certain number of points. The special case in which solution and derivatives are defined at two boundary points is called Two-Point Boundary Value Problem (TPBVP). A classical solution to this type of problem generally requires a major computational effort because it is necessary to iterate by integrating the differential equations in the domain of interest. In the particular case of linear differential equations, however, it is possible to make some simplifications and estimate a priori the number of iterations required to solve the problem. For some types of TPBVP, some direct methods are available which do not involve the integration of differential equations, reducing the computational cost. However, these methods generally involve a restriction of the solution domain and thus generally provide a near-optimal solution. The TPBVP in its standard form is defined as follows:

Given a system of N coupled first-order ODE in the independent variable $\boldsymbol{x}$, find the solution belonging to the domain $Dom(f)$ that satisfies N_1 boundary conditions at the initial point $\boldsymbol{x}_1$ and $N_2 = N - N_1$ conditions at the final point $\boldsymbol{x}_2$:

$$\dot{\boldsymbol{y}}(\boldsymbol{x}) = f(\boldsymbol{x}, \boldsymbol{y}(\boldsymbol{x})) \tag{122}$$

with:

$$\boldsymbol{x} \in Dom(f) \tag{123}$$

and

$$g_1(\boldsymbol{x_1}, \boldsymbol{y}(\boldsymbol{x_1})) = 0 \tag{124}$$

$$g_2(\boldsymbol{x_2}, \boldsymbol{y}(\boldsymbol{x_2})) = 0 \tag{125}$$

where g_1, g_2 are sets of functions that may involve the derivatives of the dependent variables and define N independent boundary conditions applied in $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$. For the case considered in this study, as explained in detail in Section 4.3, it is possible to reduce the TPBVP to a problem in two spatial dimensions for which the solution is sought in the form of a 2D trajectory in the x and y LV coordinates as a function of the time. The boundary conditions consist of the position, velocity to acceleration vectors at the starting point $\boldsymbol{x}_1$ and the desired final point $\boldsymbol{x}_2$. Figure 4-1 shows the geometric structure of the problem for the Terminal Guidance problem under investigation.

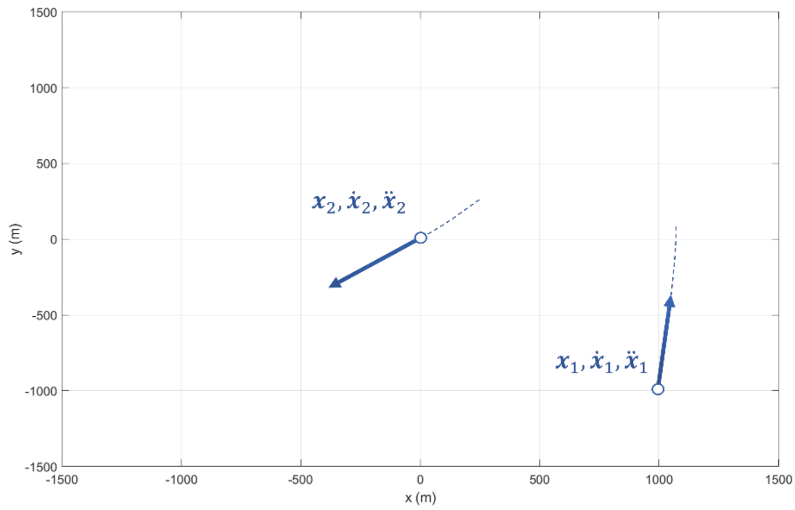

Figure 4-1: TPBVP for the Terminal Guidance problem

Regarding the existence and uniqueness of the solution to Boundary Value Problems, no theorems of general validity are yet available. However, there are several studies in the literature concerning single cases associated with specific methods of solving. For the case at hand, it is possible to demonstrate rigorously that the existence of the solution is not always guaranteed and, where it exists, it is not unique. Such a demonstration is beyond the scope of this work.

4.2. TPBVP Resolution Literature Review

The Boundary Value Problem is a widely studied problem because of its numerous applications in engineering. For this reason, there are many approaches proposed in the literature for its solution. The two main classes of methods for solving TPBVP are as follows:

- Numerical methods
- Direct methods

Numerical methods in general allow an optimal solution to be obtained without special restrictions on its form through the integration of ODEs. Among the most widely used numerical methods for TPBVP are *shooting methods* and *relaxation methods*.
The *shooting methods* require the use of a set of parameters that describe the solution of the problem and are associated with a given number of degrees of freedom (DOF). Some of the parameters are assigned specific values in order to satisfy the boundary conditions at the initial point $\boldsymbol{x_1}$. The remaining degrees of freedom determine the shape of the solution and are initially guessed arbitrarily. The differential equations are then integrated using the chosen parameters. The final conditions obtained in $\boldsymbol{x_2}$ are then compared with the desired ones. Based on the discrepancies obtained, free parameters are then adjusted in subsequent iterations to obtain the target boundary conditions in $\boldsymbol{x_2}$. Due to numerical integration, the solutions found at each iteration always respect the differential equations (122). On the other hand, the boundary conditions given by (124) and (125) are both satisfied only at the end of the iteration process with some design-defined precision. Each method provides a specific approach to systematically identify the value of free parameters to limit the number of iterations and the computational cost.
The *relaxation methods* approximate differential equations with finite-difference equations by considering a finite number of points within the domain of integration. As a first attempt, we assume a value for the dependent variables $\boldsymbol{y}$ at each point. Such values in general do not satisfy either the boundary conditions (124) and (125) or the differential equations (122). Subsequently, through successive iterations, the value

of $\boldsymbol{y}$ at the mesh points is varied so as to progressively approach the fulfillment of the problem conditions. These methods involve the presence of a large number of variables but, despite this, are convenient in some specific cases. Relaxation is indeed very efficient in case of expressions that are difficult to solve in closed form or when there are particularly stringent boundary conditions. In contrast, they are not recommended when the shape of the solution is oscillatory. In fact, in these cases, it is necessary to consider a large number of variables that slow down the method. In any case, to make the method efficient, it is important to consider an appropriate initial guess.
Numerical methods are effective in finding the optimal solution but have as their major drawback a high computational cost. In the case of real-time applications, especially for the aerospace sector, the computational budget is a major limiting factor that has led to the investigation of alternatives such as direct methods. For the latter, the approach used is to reduce the initial problem to a second one, which does not involve differential equations. This ensures fast and robust convergence that makes these methods very suitable for solving TPBVP applied to real-time trajectory generation. The Terminal Guidance problem constitutes a special case of TPBVP and can be defined as follows:

Given the initial position (x_i, y_i, z_i) *and velocity defined by* $(V_\mathrm{i}, \gamma_\mathrm{i}, \chi_\mathrm{i})$*, fly by the target point* (x_f, y_f, z_f) *with the desired velocity given by* (V_f, γ_f, χ_f).

Where it is recalled that the direction of the final velocity depends on the direction of the wind on the ground as the goal is to land upwind to minimize the touchdown velocity. The main difficulties in the design of a strategy for such a problem, as already mentioned, are related to the unpredictability of wind effects. However, there are other elements to take into account. A glided flight under parafoil without a propulsive system in fact ensures low control authority. The presence of no-fly zones is also an additional element that influences trajectory generation. In the specific case of Space Rider, it is then necessary to keep in mind the absence of an air-data system that can provide real-time data on the actual wind encountered during the descent. Several possible approaches to the problem at hand are proposed in the literature.

Fowlerl and Rogers have proposed in [17] an algorithm that involves the use of one or more cubic Bézier curves for the generation of the optimal trajectory. The parameters characterizing the Bézier curves are optimized so as to minimize the touchdown distance error from the target landing point. Model Predictive Control techniques are then applied to minimize a cost function J that can include additional penalty terms such as distances from curve midpoints to the target and maximum yaw acceleration. The problem definition also requires the maximum yaw rate not to exceed a certain threshold and the trajectory not to intersect any 3D obstacle. The structure of the problem, combined with the presence of nonlinear constraints, makes the optimization problem non-convex. The Bézier curve path planner is characterized by an optimization sequence that involves calculating the trajectory with a gradually increasing number of Bézier curves. A first trajectory with only one curve is computed, and, if a feasible solution is not found, first two curves are employed, then three, and so on. An example with two Bézier curves is shown in Figure 4-3.

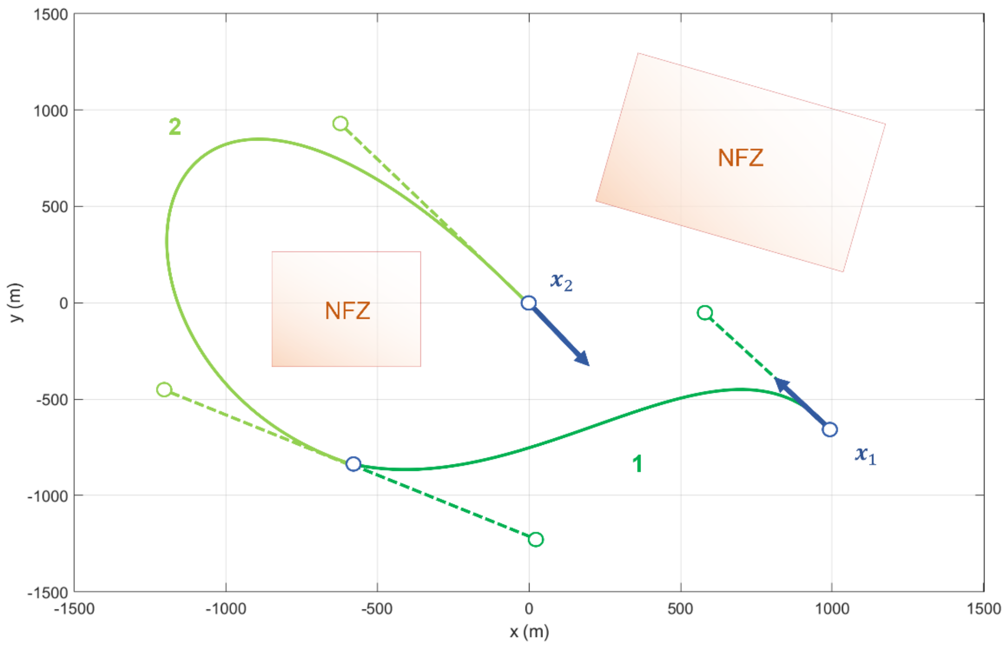

Figure 4-2: Bézier curve path planner

This approach allows a high degree of flexibility in the shape of the solution that is well suited to complex geometry configurations of the problem, especially in the presence of obstacles such as no-fly zones. The main drawback of this particular approach, however, is the high computational burden, which makes it unsuitable for real-time trajectory computation when the computational load budget is rather tight as in the case study. However, application to the Space Rider case cannot be ruled out. In fact, this method could be used for the generation of an offline reference trajectory that can be imposed to be followed using tube-based MPC techniques that effectively counteract the effect of external disturbances. The trajectory can then be periodically updated along the path by making use of a scheduler that distributes the computation of the optimal trajectory over multiple guidance cycles.

Another possible approach for solving the Terminal Guidance problem is the one proposed by Rademacher and Lu in [18] based on Dubins path. In this case, the strategy used for trajectory generation is a hybrid one as it combines the use of modified Dubins paths and minimum-control-energy paths. The type of trajectory computed is a function of the value of the parameter of normalized altitude margin from the optimal Dubins path (minimum time). The problem is solved by an appropriate change of the independent variable that converts the 3D problem to 2D enabling optimization using the minimum principle of Pontryagin. This method is quite efficient but, in contrast to the Bézier curve path planner, it does not guarantee easy collision avoidance.

Other interesting approaches are, for instance, those of Carter or Calise and Preston. Carter proposed in [19] a bandwidth-limited trajectory planner using the simplex search algorithm by Nelder and Mead that guarantees a low computational load. Calise and Preston in [20] developed a swarming control law that allows for obstacle avoidance to be taken into account by calculating the probability of collision.

Another strategy proposed in the literature is that of Slegers and Yakimenko who in [1] discuss a trajectory planner based on direct methods of calculus of variations. This approach provides high flexibility with low computational cost. It represents the method selected for this study and it is described in detail in Section 4.3.

4.3. Direct-Method-Based Approach

This section presents the basic method proposed by Slegers and Yakimenko in [1] for generating optimal trajectories of the Terminal Guidance phase of PADS. Specific modifications will then be applied to this method, which are discussed in Section 4.4. The approach under consideration is based on a direct method of calculus of variations and inverse dynamics in a virtual domain. The main idea is to obtain a method of solving that does not involve the presence of differential equations. The goal is to reduce the problem to a single parameter optimization problem so as to obtain an efficient method for real-time trajectory computation. Indeed, for the special case of the Terminal Guidance problem, a continuous update of the trajectory is required to ensure a robust design in an environment with high wind-to-airspeed ratios.

4.3.1. TPBVP Definition

First, it is required to define the boundary value problem in the form described in Section 4.1. As differential equations governing the system, it is possible to consider in first approximation the simple kinematic equations provided by the 3DOF model described in Section 3.5.1:

$$\begin{bmatrix} V_x \\ V_y \\ V_z \end{bmatrix} = \begin{bmatrix} \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} \approx \begin{bmatrix} V_{ha0}\ cos\psi \\ V_{ha0}\ sin\psi \\ V_{va0} \end{bmatrix} + \begin{bmatrix} W_x \\ W_y \\ W_z \end{bmatrix} \tag{126}$$

where it is recalled that assumptions of negligible sideslip β and constant vertical and horizontal airspeed norms (V_{va0} and V_{ha0}) were considered. An additional important assumption that can be made is that the vertical wind component is negligible:

$$W_z \approx 0 \tag{127}$$

This assumption has a major impact on the configuration of the problem because it allows the transition from the initial three-dimensional problem to a two-dimensional problem since the vertical component of groundspeed coincides with the vertical component of airspeed, which is considered constant. The differential equations to be considered for the TPBVP become:

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix} = \begin{bmatrix} V_{ha0}\ cos\psi + W_x(h) \\ V_{ha0}\ sin\psi + W_y(h) \end{bmatrix} \tag{128}$$

where the down-wind and cross-wind components, W_x and W_y, respectively, may depend on altitude h but are considered known. With this problem formulation, a central parameter for the application of the direct method can be easily calculated, namely, the total duration of the Terminal Guidance maneuver. It will be given by:

$$T_m = \frac{(\Delta h)_{terguid}}{V_{va0}} \tag{129}$$

It is now required to define the boundary conditions at the initial point x_1 and at the final point x_2. Henceforth the initial and final conditions will be referred to as subscripts i and f respectively. In both cases, the position, velocity, and acceleration vectors must be provided. The initial point considers the current state of the system while the end final consists of the target of the terminal guidance phase. The boundary conditions below are given in the physical domain using time t as the independent variable. The position vectors are:

$$\begin{bmatrix} x \\ y \end{bmatrix}_{t=t_i} = \begin{bmatrix} x_i \\ y_i \end{bmatrix} \tag{130}$$

$$\begin{bmatrix} x \\ y \end{bmatrix}_{t=t_f} = \begin{bmatrix} x_f \\ y_f \end{bmatrix} \tag{131}$$

Velocity vectors are obtained from the equations of kinematics (126):

$$\begin{bmatrix} \dot{x} \\ \dot{y} \end{bmatrix}_{t=t_i} = \begin{bmatrix} V_{ha0}\ cos\psi_i + W_x(h_i) \\ V_{ha0}\ sin\psi_i + W_y(h_i) \end{bmatrix} \tag{132}$$

$$\begin{bmatrix} x \\ y \end{bmatrix}_{t=t_f} = \begin{bmatrix} V_{ha0}\ cos\psi_f + W_x(h_f) \\ V_{ha0}\ sin\psi_f + W_y(h_f) \end{bmatrix} \tag{133}$$

Deriving with respect to time yields the boundary conditions in terms of accelerations:

$$\begin{bmatrix} x \\ y \end{bmatrix}_{t=t_i} = \begin{bmatrix} -\ \psi_i V_{ha0}\ sin\psi_i - W_x'\ \ h_i\ \ \ V_{va0} \\ +\ \psi_i V_{ha0}\ cos\psi_i - W_y'\ \ h_i\ \ \ V_{va0} \end{bmatrix} \tag{134}$$

$$\begin{bmatrix}\ddot{x}\\ \ddot{y}\end{bmatrix}_{t=t_f} = \begin{bmatrix}-\ \dot{\psi}_f V_{ha0}\, sin\, \psi_f - W_x'(h_f)V_{va0}\\ +\ \dot{\psi}_f V_{ha0}\, cos\, \psi_f - W_y'(h_f)V_{va0}\end{bmatrix} = \begin{bmatrix}-W_x'(h_f)V_{va0}\\ -W_y'(h_f)V_{va0}\end{bmatrix} \tag{135}$$

considering the following relationships between the derivative with respect to time ($\dot{W}$) and that with respect to altitude (W') of wind components:

$$\dot{W}_x = \frac{dW_x}{dt} = \frac{dW_x}{dh}\frac{dh}{dt} = \ W_x'(h)\ (-V_{va0}) \tag{136}$$

$$\dot{W}_y = \frac{dW_y}{dt} = \frac{dW_y}{dh}\frac{dh}{dt} = \ W_y'(h)\ (-V_{va0}) \tag{137}$$

recalling that the vertical component V_{va0} of airspeed is described in LV which is characterized by a downward z axis. The use of the derivative in altitude is due to the fact that the wind profile is often known as a function of altitude and not of time. For equation (135), it was also imposed that the derivative of the final heading rate be zero:

$$\dot{\psi}_f = 0 \tag{138}$$

This ensures a smoother arrival that results in a straight line in the final point approach phase. In addition, in the special case of constant wind profile, the right-hand side of Equation (135) cancels out and thus the accelerations at the arrival point are zero. Expression (128) together with the boundary conditions defined by equations (130) through (135) define the Two-Point Boundary Value Problem applied to the Terminal Guidance phase of a flight under parafoil. The physical domain of definition of the independent variable that consists of time is as follows:

$$t \in [0, T_m] \tag{139}$$

It is important to note that during the TERGUID phase there is a continuous update of the optimal trajectory so that the initial conditions defined above are also updated at each guidance step when a new trajectory is generated. Similarly, the time domain to be considered varies at each step.

4.3.2. Direct Method Algorithm

This section describes the direct method selected for solving the TPBVP. This method does not involve the use of differential equations since it reduces the resolution to single parameter optimization. Thus, the objective is to minimize a cost function with respect to a single parameter. The parameter under consideration consists of the virtual variable τ. In fact, the method involves describing the trajectory in an appropriate virtual domain for optimization. Once the optimal trajectory in the virtual domain is found, the solution is returned to the physical domain. Finally, knowing the trajectory as a function of time, the control input is derived by inverting the dynamics.

First, it is required to select the form of the analytical solution in the virtual domain. This choice is relevant since a significant restriction is being made to the spectrum of possible solutions to the problem. This is why it is generally referred to as a near-optimal solution. Considering the virtual domain:

$$\tau \in [\tau_i, \tau_f] \tag{140}$$

solutions of the following type are considered:

$$\begin{cases} x(\bar{\tau}) = P_1(\bar{\tau}) = a_0^1 + a_1^1\bar{\tau} + a_2^1\bar{\tau}^2 + a_3^1\bar{\tau}^3 + b_1^1 sin\ (\pi\bar{\tau}) + b_2^1 sin\ (2\pi\bar{\tau}) \\ y(\bar{\tau}) = P_2(\bar{\tau}) = a_0^2 + a_1^2 + a_2^2\bar{\tau}^2 + a_3^2\bar{\tau}^3 + b_1^2 sin\ (\pi\bar{\tau}) + b_2^2 sin\ (2\pi\bar{\tau}) \end{cases} \tag{141}$$

where the normalized virtual variable $\bar{\tau}$ is given by:

$$\bar{\tau} = \frac{\tau}{\tau_f} \in [\bar{\tau}_i, \bar{\tau}_f] = [0,1] \tag{142}$$

The type of solution chosen consists of a polynomial part and a sinusoidal part. This choice guarantees a rather malleable shape of the solution as well as advantages in terms of computation of the derivatives of x and y. For trajectory definition, the values of the coefficients must be identified a_m^η, b_n^η with $\eta = 1,2,$ $m = 0,1,2,3,$ and $n = 1,2.$ For this purpose, the coordinate x and y are derived twice with respect to the nonnormalized virtual variable $(\tau = \bar{\tau}\tau_f)$:

$$x'(\bar{\tau}) = \frac{1}{\tau_f}(a_1^1 + 2a_2^1\bar{\tau} + 3a_3^1\bar{\tau}^2 + \pi\ b_1^1\, cos(\pi\bar{\tau}) + 2\pi\ b_2^1\, cos(2\pi\bar{\tau})) \tag{143}$$

$$x''(\bar{\tau}) = \frac{1}{{\tau_f}^2}(2a_2^1 + 6a_3^1\bar{\tau} - \pi^2 b_1^1\, sin(\pi\bar{\tau}) - 2\pi^2\ b_2^1\, sin(2\pi\bar{\tau})) \tag{144}$$

and

$$y''(\bar{\tau}) = \frac{1}{\tau_f}(a_1^2 + 2a_2^2\bar{\tau} + 3a_3^2\bar{\tau}^2 + \pi\ b_1^2\, cos(\pi\bar{\tau}) + 2\pi\ b_2^2\, cos(2\pi\bar{\tau})) \tag{145}$$

$$y''(\bar{\tau}) = \frac{1}{{\tau_f}^2}(2a_2^2 + 6a_3^2\bar{\tau} - \pi^2 b_1^2\, sin(\pi\bar{\tau}) - 2\pi^2\ b_2^2\, sin(2\pi\bar{\tau})) \tag{146}$$

The values of x, y, and the respective derivatives at the two boundary points are now derived. Considering, for instance, the x-coordinate, for the first extreme:

$$\begin{gathered} x(\bar{\tau}_i) = x(0) = x_i \\ = a_0^1 + a_1^1\bar{\tau} + a_2^1\bar{\tau}^2 + a_3^1\bar{\tau}^3 + b_1^1 sin\ (\pi\bar{\tau}) + b_2^1 sin\ (2\pi\bar{\tau}) = a_0^1 \end{gathered} \tag{147}$$

$$\begin{gathered} x'(\bar{\tau}_i) = x'(0) = x_i' \\ = \frac{1}{\tau_f}(a_1^1 + 2a_2^1\bar{\tau} + 3a_3^1\bar{\tau}^2 + \pi\ b_1^1\, cos(\pi\bar{\tau}) + 2\pi\ b_2^1\, cos(2\pi\bar{\tau})) \\ = \frac{1}{\tau_f}\ (a_1^1 + \pi\ b_1^1 + 2\pi\ b_2^1) \end{gathered} \tag{148}$$

$$\begin{gathered} x''(\bar{\tau}_i) = x''(0) = x_i'' = \\ \frac{1}{{\tau_f}^2}(2a_2^1 + 6a_3^1\bar{\tau} - \pi^2 b_1^1\, sin(\pi\bar{\tau}) - 2\pi^2\ b_2^1\, sin(2\pi\bar{\tau})) = \frac{1}{{\tau_f}^2}(2a_2^1) \end{gathered} \tag{149}$$

with:

$$\bar{\tau}_i = \frac{\tau_i}{\tau_f} = 0 \tag{150}$$

On the other hand, for the second extreme results:

$$\begin{aligned} x(\bar{\tau}_f) &= x(1) = x_f \\ x_f = a_0^1 + a_1^1\bar{\tau} + a_2^1\bar{\tau}^2 &+ a_3^1\bar{\tau}^3 + b_1^1\, sin(\pi\bar{\tau}) + b_2^1\, sin(2\pi\bar{\tau}) \\ &= \ a_0^1 + a_1^1 + a_2^1 + a_3^1 \end{aligned} \tag{151}$$

$$\begin{aligned} x'(\bar{\tau}_f) &= x'(1) = x'_f \\ = \frac{1}{\tau_f}(a_1^1 + 2a_2^1\bar{\tau} + 3a_3^1\bar{\tau}^2 &+ \pi\ b_1^1\, cos(\pi\bar{\tau}) + 2\pi\ b_2^1\, cos(2\pi\bar{\tau})) \\ &= \frac{1}{\tau_f}(a_1^1 + 2a_2^1 + 3a_3^1 - \pi\ b_1^1 + 2\pi\ b_2^1) \end{aligned} \tag{152}$$

$$\begin{aligned} x''(\bar{\tau}_f) &= x''(1) = x''_f \\ = \frac{1}{{\tau_f}^2}(2a_2^1 + 6a_3^1\bar{\tau} &- \pi^2 b_1^1\, sin(\pi\bar{\tau}) - 2\pi^2\ b_2^1\, sin(2\pi\bar{\tau})) \\ &= \frac{1}{{\tau_f}^2}(2a_2^1 + 6a_3^1) \end{aligned} \tag{153}$$

with:

$$\bar{\tau}_f = \frac{\tau_f}{\tau_f} = 1 \tag{154}$$

Therefore, for the x coordinate alone, a system of 6 equations in the 6 unknowns $a_m^1,\, b_n^1$ with $\bar{\tau}_f$ as a parameter is obtained. This system, expressed in matrix form, is as follows:

$$\begin{bmatrix} x_0 \\ x_f \\ x'_0\ \tau_f \\ x'_f\ \tau_f \\ x''_0\ {\tau_f}^2 \\ x''_f\ {\tau_f}^2 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & \pi & 2\pi \\ 0 & 1 & 2 & 3 & -\pi & 2\pi \\ 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 2 & 6 & 0 & 0 \end{bmatrix} \begin{bmatrix} a_0^1 \\ a_1^1 \\ a_2^1 \\ a_3^1 \\ b_1^1 \\ b_2^1 \end{bmatrix} \tag{155}$$

The coefficients $a_m^1,\, b_n^1$ are then obtained as a function of the single parameter τ_f:

$$a_0^1 = x_0 \tag{156}$$

$$a_1^1 = -(x_0 - x_f) + \frac{1}{6}\tau_f^2(2x_0'' + x_f'') \tag{157}$$

$$a_2^1 = \frac{1}{2}\tau_f^2 x_0'' \tag{158}$$

$$a_3^1 = -\frac{1}{6}\tau_f^2(x_0'' - x_f'') \tag{159}$$

$$b_1^1 = \frac{1}{4\pi}\left(2\tau_f(x_0' - x_f') + \tau_f^2(x_0'' + x_f'')\right) \tag{160}$$

$$b_2^1 = \frac{1}{24\pi}\left(12(x_0 - x_f) + 6\tau_f(x_0' + x_f') + \tau_f^2(x_0'' - x_f'')\right) \tag{161}$$

A quite similar procedure can be followed for the y-coordinate for which 6 more equations for a_m^2, b_n^2 in the parameter τ_f are obtained. It is important now to keep in mind that the derivatives seen so far are taken in the virtual domain. it is possible to shift to the physical domain by considering that:

$$\dot{x} = \frac{dx}{dt} = \frac{dx}{d\tau}\frac{d\tau}{dt} = \frac{dx}{d\tau}\lambda = \lambda x' \tag{162}$$

$$\ddot{x} = \frac{d}{dt}\left(\frac{dx}{dt}\right) = \frac{d(\lambda x')}{dt} = \frac{d(\lambda x')}{d\tau}\frac{d\tau}{dt} = (\lambda' x' + \lambda x'')\lambda \tag{163}$$

with:

$$\lambda = \frac{d\tau}{dt} \tag{164}$$

To switch from virtual domain to physical domain then:

$$x' = \lambda^{-1}\dot{x} \tag{165}$$

$$x'' = \lambda^{-1}(\lambda^{-1}\ddot{x} - \lambda' x') = \lambda^{-1}(\lambda^{-1}\ddot{x} - \lambda'\lambda^{-1}\dot{x}) = \lambda^{-2}(\ddot{x} - \lambda'\dot{x}) \tag{166}$$

It can now be noted that the derivatives are calculated only at the domain extremes when boundary conditions are imposed so it is useful to take into account what is the value of the derivative at the extremes:

$$\lambda_i = \frac{d\tau}{dt}\bigg|_i = 1 \tag{167}$$

$$\lambda_f = \frac{d\tau}{dt}\bigg|_f = 1 \tag{168}$$

$$\lambda_i' = \frac{d}{d\tau}\left(\frac{d\tau}{dt}\right)\bigg|_i = 0 \tag{169}$$

$$\lambda_f' = \frac{d}{d\tau}\left(\frac{d\tau}{dt}\right)\bigg|_f = 0 \tag{170}$$

The derivatives at the extremes then result as follows:

$$x' = \lambda^{-1}\dot{x} = \dot{x} \tag{171}$$

$$x'' = \lambda^{-2}(\ddot{x} - \lambda'\dot{x}) = \ddot{x} \tag{172}$$

At this stage, by fixing the parameter τ_f, the coefficients a_m^η, b_n^η and thus the trajectory in the virtual domain can be easily obtained. Then, the x and y coordinates and respective x' and y' derivatives are computed over a fixed set of N points (or nodes) spaced evenly along the virtual arc $[0, \tau_f]$. The virtual interval under consideration is thus:

$$\Delta\tau = \frac{\tau_f}{N-1} \tag{173}$$

From which the values τ_j to be considered are obtained:

$$\tau_j = \tau_{j-1} + \Delta\tau \quad \textit{for} \quad j = 2, \dots, N \tag{174}$$

with:

$$\tau_1 = 0 \tag{175}$$

Again, considering only the x-coordinate as an example, the following is obtained for each of the N points:

$$x(\tau_j) = x_j = a_0^1 + a_1^1\tau_j + a_2^1{\tau_j}^2 + a_3^1{\tau_j}^3 + b_1^1 sin\ (\pi\tau_j) + b_2^1 sin\ (2\pi\tau_j) \tag{176}$$

$$x'(\tau_j) = x_j' = \frac{1}{\tau_f}(a_1^1 + 2a_2^1\tau_j + 3a_3^1{\tau_j}^2 + \pi\ b_1^1\, cos(\pi\tau_j) + 2\pi\ b_2^1\, cos(2\pi\tau_j)) \tag{177}$$

At this point, it is needed to compute the time interval Δt in the physical domain corresponding to each interval $\Delta\tau$ in the virtual domain. In fact, Δt intervals, in contrast to $\Delta\tau$ intervals, are not all equal:

$$\Delta t_{j-1} = \frac{(\Delta x)_{j-1}}{(V_h)_{j-1}} = \frac{\sqrt{(x_j - x_{j-1})^2 + (y_j - y_{j-1})^2}}{(V_h)_{j-1}} \tag{178}$$

where the horizontal ground speed vector V_h, taking into account the steady-state horizontal airspeed hypothesis and the equations of the kinematics (126), is given by:

$$\begin{aligned}(V_h)_{j-1} &= \sqrt{\dot{x}_{j-1}^2 + \dot{y}_{j-1}^2} \\ &= \sqrt{V_{ha0}^2 + (W_x(h_{j-1}) + W_y(h_{j-1}))^2 + 2V_{ha0}\left(W_x(h_{j-1})cos(\psi_{j-1}) + W_y(h_{j-1})sin(\psi_{j-1})\right)}\end{aligned} \tag{179}$$

Once all the time intervals Δt in the physical domain have been calculated, it is possible to calculate the total time required to cover the entire trajectory:

$$\Delta t_{tot} = \sum_{j=2}^{N} \Delta t_{j-1} \tag{180}$$

It is now possible to build the cost function J to be minimized in the single parameter τ_f considering the target maneuver time T_m (129):

$$J = (\Delta t_{tot} - T_m)^2 \tag{181}$$

Minimizing the function, therefore, means ensuring that the total duration of the maneuver Δt_{tot} matches the desired one T_m. Any optimization function such as the MATLAB functions *fminbnd* or *fminsearch* can be used to solve this problem. As an initial guess for the τ_f parameter, half the value of the circumference with a diameter equal to the distance between the initial and the final point $(\boldsymbol{x}_i, \boldsymbol{x}_f)$ can be employed:

$$\tau_{f0} = \frac{\pi}{2}\sqrt{(x_f - x_i)^2 + (y_f - y_i)^2} \tag{182}$$

The result of minimization is as follows:

$$J_{opt} = \min_{\tau_f} J \tag{183}$$

$$(\tau_f)_{opt} = \arg\min_{\tau_f} J \tag{184}$$

Once the optimal value $(\tau_f)_{opt}$ for the virtual parameter τ_f is found, the trajectory can be easily derived by computing the coefficients thanks to equations (156) through (161) for the x coordinate and 6 other similar equations for the y coordinate. An example of a trajectory generated with this method using $N = 20$ and no wind is shown in the Figure 4-3. In dark blue are shown the boundary conditions of the problem, in green the generated optimal trajectory while in light blue are indicated the velocity vectors along the trajectory. Given the assumption of no wind, the latter are all of the same magnitude, equal to the steady-state horizontal airspeed V_{ha0}. The reported trajectory is in 2D because, as previously discussed, we assume constant vertical ground speed, so the third dimension is not relevant to the problem.

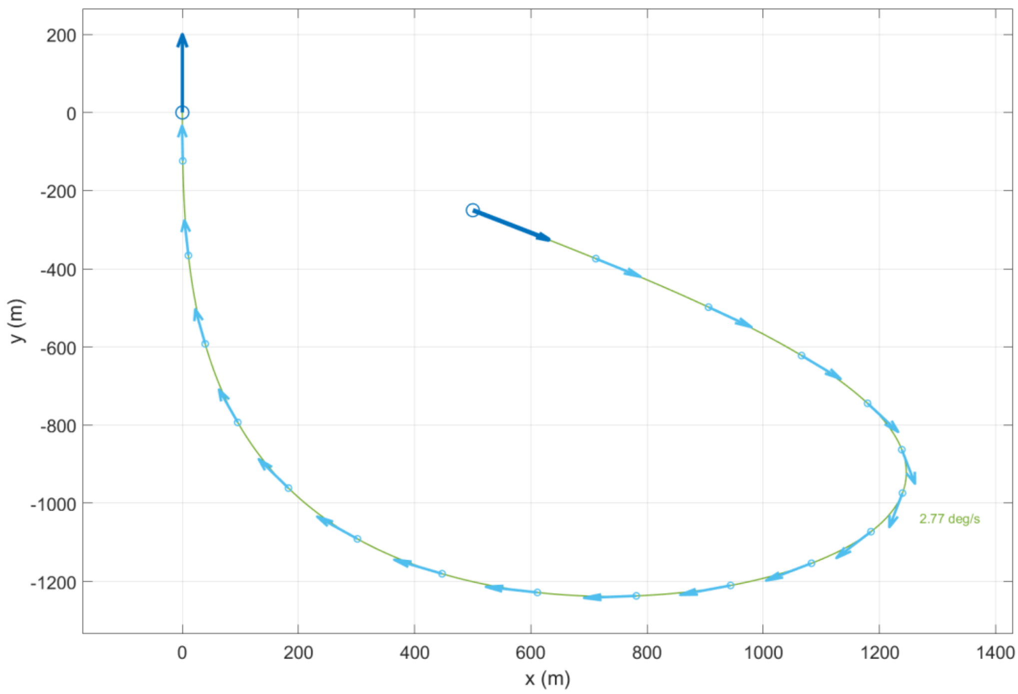

Figure 4-3: Example of optimal trajectory

Once the optimal trajectory is known, it is necessary to derive the heading and heading rate sequence with respect to the ground speed vector where the latter constitutes the control input. Given the assumptions made in Section 3.5.1, it is possible to equivalently consider the yaw angle ψ and yaw rate $\dot{\psi}$ instead of heading angle and heading rate. A simple, purely geometric procedure can be employed to derive the sequence of yaw angle values along the trajectory. From equations (128) it follows:

$$V_{ha0} = \frac{\dot{x} - W_x}{cos\psi} = \frac{\dot{y} - W_y}{sin\psi} \tag{185}$$

therefore:

$$tan\psi = \frac{sin\psi}{cos\psi} = \frac{\dot{y} - W_y}{\dot{x} - W_x} \tag{186}$$

Then, for each node, recalling equation (162), it can be computed:

$$\psi_j = atan\left(\frac{\dot{y}_j - W_y(h_j)}{\dot{x}_j - W_x(h_j)}\right) = atan\left(\frac{\lambda_j y'(\tau_j) - W_y(h_j)}{\lambda_j x'(\tau_j) - W_x(h_j)}\right) \tag{187}$$

where it is possible to approximate:

$$\lambda_j = \frac{\Delta\tau}{\Delta t_{j-1}} \tag{188}$$

Through (187), the sequence of yaw angles at each of the N reference points of the trajectory is determined. To derive the yaw rate sequence to be applied in order to follow the optimal trajectory several possible approaches are described in detail in Chapter 5. Here a simple geometric method based on finite-difference is illustrated. This approach does not take into account the actual dynamics of the vehicle, but it is useful to provide an idea of the order of magnitude of the yaw rate values to be assumed to follow the trajectory:

$$\dot{\psi}_j = \frac{(\psi_j - \psi_{j-1})}{\Delta t_{j-1}} \tag{189}$$

As described in Section 3.2, the maximum heading rate (thus the maximum yaw rate) is one of the main limiting factors for the design of the PGNC. It is therefore critical to make sure that the values assumed by $\dot{\psi}_j$ do not exceed a certain threshold when planning the trajectory. For this reason, it is reasonable to consider adding a term to the cost function J that takes into account the maximum heading rate value $\dot{\psi}_{max}$:

$$J = (\Delta t_{tot} - T_m)^2 + k_{\dot{\psi}}\ \Delta\dot{\psi}_J \tag{190}$$

with:

$$\Delta\dot{\psi}_J = \max_j \left(0, (|\dot{\psi}_j| - \dot{\psi}_{max})\right)^2 \tag{191}$$

where $k_{\dot{\psi}}$ is a weighting coefficient for the penalty term $\Delta\psi_J$.

In conclusion, an efficient TPBVP solving method that is based on single-parameter optimization in τ_f was obtained. This approach is based on a simplified 3DOF model of the dynamics of the payload-parafoil system, so it is subject to all the corresponding assumptions (see Section 3.5.1). However, it can be proven that, for limited turn rates, it provides results with good accuracy. This fits with the limitations of the parafoil dynamics which in any case do not allow high turn rates. In addition, the algorithm described in this section requires a very low computational cost. This allows the trajectory to be updated in real-time very frequently if required. However, as will be discussed in Section 6.3, frequent optimal trajectory updating is not required in practice, which, indeed can be detrimental to driving strategy. It will be shown that an update every 5-15 seconds can be reasonable in many cases.

4.4. Algorithm Modifications

The method outlined in Section 4.3, if applied to the Space Rider case without modification, may present some critical issues that make it unsuitable for meeting mission requirements. This section describes the main modifications carried out in order to adapt Slegers and Yakimenko method to the case at hand. For the design of the final algorithm, a series of tools were developed in the MATLAB environment to test the different solutions described in the following sections. One of the tools developed allows testing of trajectory generation by varying one of the boundary conditions and keeping the others fixed. Figure 4-4, for instance, shows the case where all boundary conditions are fixed except for the direction of the initial airspeed. Fixed boundary conditions consisting of initial and final position and final velocity are shown in blue. Instead, the different trajectories generated as the direction of the initial airspeed changes are indicated with three different colors. There are three categories of solutions that can be obtained through the optimization process:

- Optimal solution (green)
- Sub-optimal solution (yellow)
- Not acceptable solution (red)

For the solution not to be unacceptable first (optimal or suboptimal solution), the following requirement must be met:

$$TR = \frac{|T_m - \Delta t_{tot}|}{T_m} < TR_{max} \tag{192}$$

where the value $TR_{max} = 0.01$ is considered for this study. Therefore, the time ratio TR must be such that the difference between desired maneuver duration T_m and achieved maneuver duration Δt_{tot} is less than 1% of the target maneuver time T_m. This value constitutes a tunable parameter and the value of 0.01 is the result of a series of trade-offs conducted during the testing phase.

If this condition is met, the criterion for assigning the category to the solution depends on the maximum yaw rate $\dot{\psi}_{max}$ obtained along the entire trajectory which is calculated through the simplified method described in Section 4.3.2. Categories are then assigned as follows:

- Optimal solution: $TR < TR_{max}$ and $\dot{\psi}_{max} < \dot{\psi}_{lim1}$
- Sub-optimal solution: $TR < TR_{max}$ and $\dot{\psi}_{lim1} \leq \dot{\psi}_{max} < \dot{\psi}_{lim2}$
- Not acceptable solution: $TR > TR_{max}$ and/or $\dot{\psi}_{max} \geq \dot{\psi}_{lim1}$

The solution is considered optimal only if the maximum yaw rate $\dot{\psi}_{max}$ is less than a specific threshold value $\dot{\psi}_{lim1}$, which for this work is considered to be 7 deg/s. There is then a sub-optimal solution for which the yaw rate can still be considered acceptable under certain circumstances. For the sub-optimal solution, the yaw rate must not exceed a second threshold $\dot{\psi}_{lim2}$ set equal to 10 deg/s. The $\dot{\psi}_{lim1}$, $\dot{\psi}_{lim2}$ values are system parameters that depend on the dynamic properties of the parafoil-payload system. Figure 4-4 also shows the maximum yaw rate value located, for each trajectory, near the point of the path where this value is assumed.

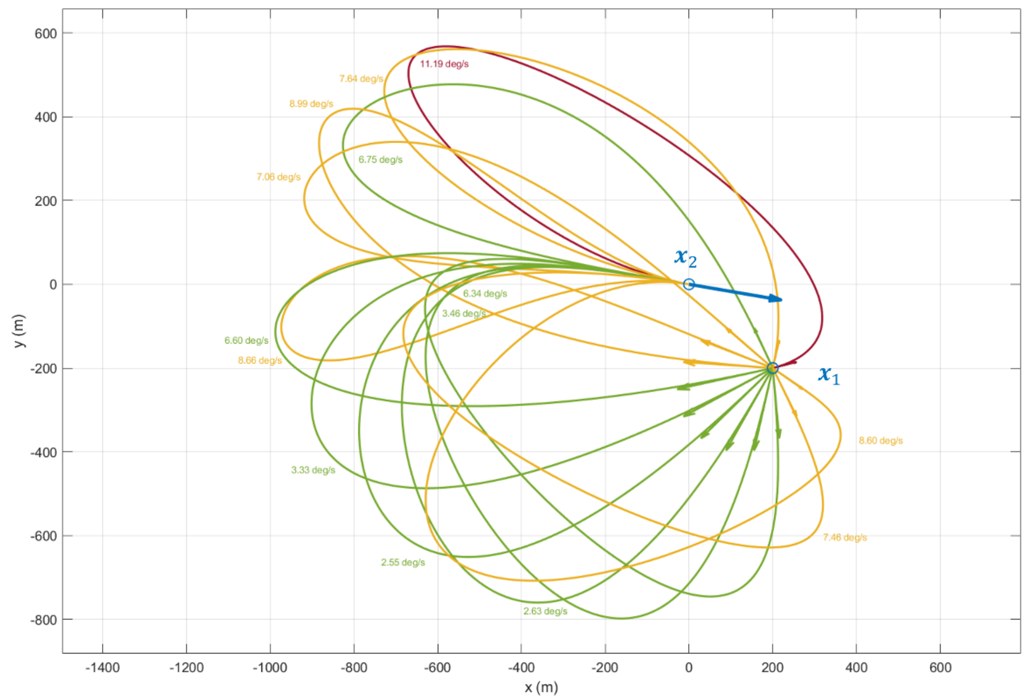

Figure 4-4: Trajectory generation tool

The tool just described allows direct integration of the MATLAB functions developed in this study for the GNC software to test them before implementation. For this purpose, an additional tool was developed to run a Monte Carlo campaign in which all boundary conditions are varied (and not only one) so that the performance of the trajectory generation algorithm can be analyzed. In addition, there are a set of other tools, for instance, for comparing two different algorithms applied to the same problem, analyzing the computation time required, or verifying the yaw rate sequences obtained. These tools will be briefly described in subsequent sections where needed.

4.4.1. Cost Function

First adjustments were made by modifying the cost function J reported in (190). The changes are related to three main causes:

- Unphysical solutions
- Ineffective yaw rate penalty term $\Delta\dot{\psi}_J$
- No-fly zones avoidance

The mathematical formulation of the algorithm described in Section 4.3 always guarantees the presence of a solution, which, however, in some cases, may be considered unacceptable according to the criteria described above. Moreover, there are some further cases where the mathematical solution found turns out to be nonphysical. These are particular cases where the initial/final direction of the trajectory is aligned with the initial/final velocity but has the opposite orientation. An example is shown in Figure 4-5 where both initial and final directions are wrong (remembering that the velocity at the final point is in blue). To solve this problem, it is sufficient to add two additional penalty terms to the cost function. The first is the square of the angular difference between initial trajectory yaw and initial velocity yaw (intended as a boundary condition) and the second is analogous but applied to the final point:

$$J = (\Delta t_{tot} - T_M)^2 + k_{\dot{\psi}}\ \Delta\dot{\psi}_J + k_{\psi}(\Delta\psi_{Ji} + \Delta\psi_{Jf}) \tag{193}$$

with:

$$\Delta\psi_{Ji} = (\psi_i - \psi_{Vi})^2 \tag{194}$$

$$\Delta\psi_{Jf} = (\psi_f - \psi_{Vf})^2 \tag{195}$$

where ψ_i and ψ_f denote respectively the initial and final yaw relative to the optimal trajectory, while ψ_{Vi} and ψ_{Vf} denote the yaw relative to the initial and final velocities provided as boundary conditions. It is important to emphasize that for simplicity of notation the angular difference has been denoted by the simple subtraction symbol. This operation actually requires the development of a special algorithm that takes into account all possible combinations in terms of directions in the 4 quadrants of the plane. Indeed, it is essential that in the case where trajectory direction and velocity are aligned but with opposite orientations, the result of the operation is 180 and not 0 degrees. The parameter k_ψ is the weighting coefficient for the penalty terms just introduced. The value acquired by it must be arbitrarily large so as to avoid in any way an unphysical solution.

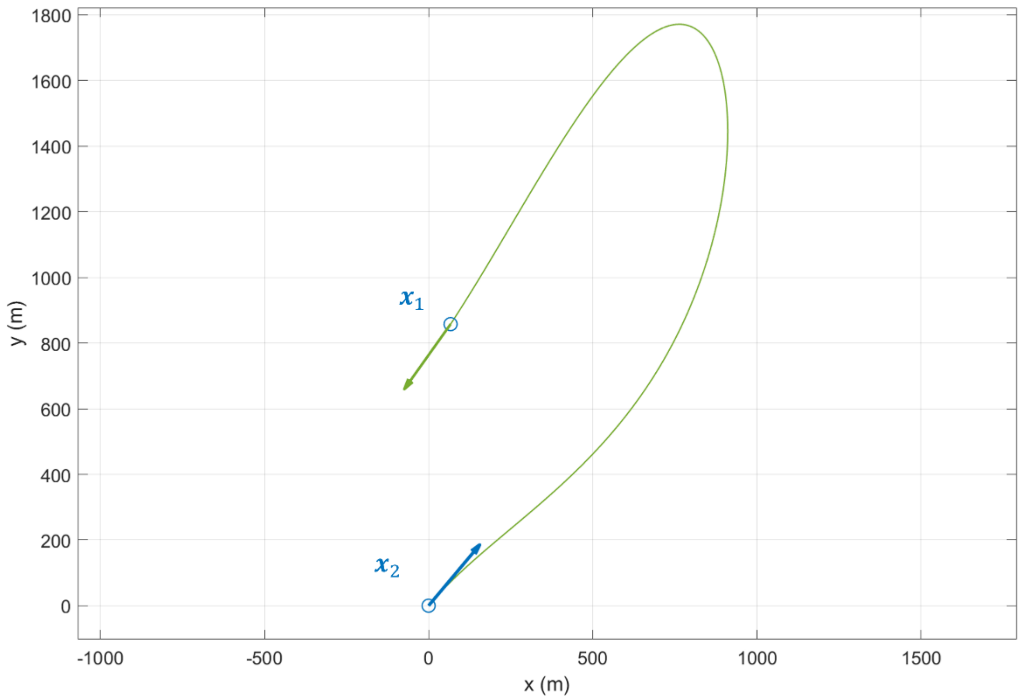

Figure 4-5: Unphysical solution

Figure 4-6 shows an example in which the results obtained with and without additional terms $\Delta\psi_{Ji}$ and $\Delta\psi_{Jf}$ in the cost function are compared. On the left, it can be seen that in some cases, for certain geometric conditions of the problem, unphysical solutions are obtained if no specific measures are taken. On the other hand, on the right, the terms $\Delta\psi_{Ji}$ and $\Delta\psi_{Jf}$ are introduced within the cost function J and the nonphysical cases no longer appear.

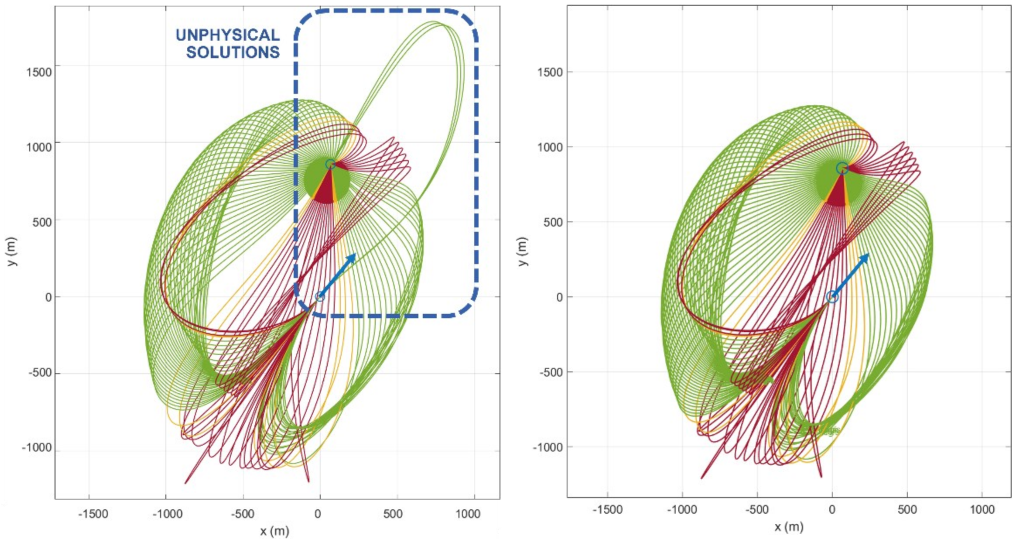

Figure 4-6: Unphysical solutions correction

A second improvement that can be performed on the cost function concerns the maximum yaw rate term $\Delta\dot{\psi}_J$ defined in the (191). The definitions considered so far only use the single maximum yaw rate value encountered along the entire trajectory. This approach can be made more efficient by considering an integral along the trajectory of all yaw rate values that exceed the threshold $\dot{\psi}_{lim2}$. Taking into account that the yaw rate is computed only over a finite number of nodes, it is possible to redefine the penalty term for the yaw rate as follows:

$$\Delta\dot{\psi}'_J = \sum \dot{\psi}_j \quad \text{such that} \quad \dot{\psi}_j \geq \dot{\psi}_{lim2} \tag{196}$$

The cost function then becomes:

$$J = (\Delta t_{tot} - T_M)^2 + k_{\dot{\psi}}\, \Delta\dot{\psi}'_J + k_{\psi}(\Delta\psi_{Ji} + \Delta\psi_{Jf}) \tag{197}$$

A specific test campaign carried out in this regard has shown that with this new formulation the algorithm can more easily detect and avoid cases where the yaw rate limit is exceeded.
A final change made on the cost function concerns the possibility of avoiding intersections with no-fly zones. For this purpose, an algorithm was developed in order to detect possible intersections by simply checking whether any of the nodes of the trajectory is contained in the geometric area of one of the no-fly zones. A new term ΔN_J is then introduced into the cost function, which results in zero if there are no intersections and one otherwise as shown in Figure 4-7. The cost function becomes:

$$J = (\Delta t_{tot} - T_M)^2 + k_{\dot{\psi}}\ \Delta \dot{\psi}'_J + k_{\psi}(\Delta\psi_{Ji} + \Delta\psi_{Jf}) + k_N\ \Delta N_J \tag{198}$$

where k_N is a weighting coefficient that, as in the case of k_{ψ}, must be large enough to ensure that there are always no intersections with no-fly zones. This simple expedient effectively avoids no-fly zones in all cases where they do not impose too restrictive conditions by requiring complex path geometry.

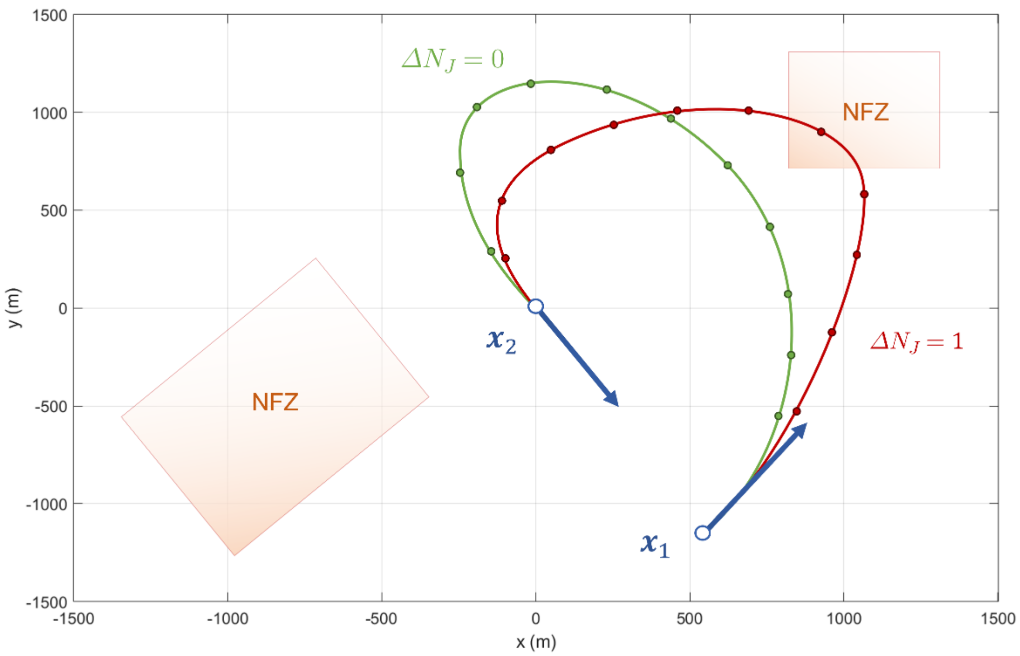

Figure 4-7: No-fly zones avoidance

4.4.2. Wind Drift and Wind-Fixed Frame

The wind is a crucial element of the Terminal Guidance problem, and predicting its effects is a key factor for mission success. For this reason, the GNC subsystem is provided with a wind table that is sent from the ground station and updated with a certain frequency. This table provides a wind profile over the altitude based on the data collected by dedicated instrumentation. By means of such a table, it is possible to estimate what the effects of wind may be on the next phase of flight. It is common practice in these cases to introduce a new reference system called Wind-fixed Frame (WF) that considers a virtual position of the vehicle given by the sum of the actual position with the wind drift vector. Given the assumptions of no vertical wind, this corresponds to a shift in the horizontal plane. The position of the vehicle in such a reference system is generally denoted by air position $\boldsymbol{X}_a$. The wind drift vector is defined as follows:

$$\Delta\boldsymbol{X}_W = \begin{bmatrix}\Delta x_W \\ \Delta y_W\end{bmatrix} = \int_{t_i}^{t_f} \begin{bmatrix}W_x(t) \\ W_y(t)\end{bmatrix} dt \tag{199}$$

Considering that the wind profile given by the table is a function of altitude h, expression (199) can be rewritten as follows:

$$\Delta\boldsymbol{X}_W = \int_{t_i}^{t_f} \begin{bmatrix}W_x(t) \\ W_y(t)\end{bmatrix} dt = \int_{h_0}^{h_f} \begin{bmatrix}W_x(h) \\ W_y(h)\end{bmatrix} \frac{dt}{dh} dh \approx -\int_{h_0}^{h_f} \begin{bmatrix}W_x(h) \\ W_y(h)\end{bmatrix} \frac{dh}{V_{va0}} \tag{200}$$

The air coordinates in the WF system are given by:

$$\boldsymbol{X}_a = \boldsymbol{X} + \Delta\boldsymbol{X}_W = \begin{bmatrix}x + \Delta x_W \\ y + \Delta y_W\end{bmatrix} \tag{201}$$

The strategy adopted in this study is to remove any kind of reference to wind speed within the direct method algorithm by always using air coordinates and airspeed instead of ground speed. The effect of wind is taken into account through the drift contribution alone. As an example, the equations of kinematics (128) become:

$$\begin{bmatrix}\dot{x}_a \\ \dot{y}_a\end{bmatrix} = \begin{bmatrix}V_{xa} \\ V_{ya}\end{bmatrix} = \begin{bmatrix}V_{ha0}\ cos\psi \\ V_{ha0}\ sin\psi)\end{bmatrix} \tag{202}$$

4.4.3. Two-Parameter Optimization

The algorithm described so far proves to be very effective for specific geometric conditions. For example, in the case of a maneuver with a yaw variation of 180 degrees, the algorithm always provides an optimal solution. This is clearly true if the distance between the initial and the final point is not too large taking into account the horizontal velocity and the difference in altitude to be traveled (under the assumptions of constant vertical velocity). If the delta of altitude is not sufficient to guarantee the time needed to travel the horizontal distance between the initial and the final point, there is no physical solution to the problem. Figure 4-8 shows an example of a problem geometry (with $\Delta\psi = 180°$) for which the tests performed demonstrated the complete reliability of the algorithm. A Monte Carlo campaign was conducted by varying all the boundary conditions but keeping the delta yaw $\Delta\psi$ around 180 degrees. In 100% of the cases with an available physical solution, an optimal trajectory was found (meeting all requirements in terms of maneuver time and maximum yaw rate).

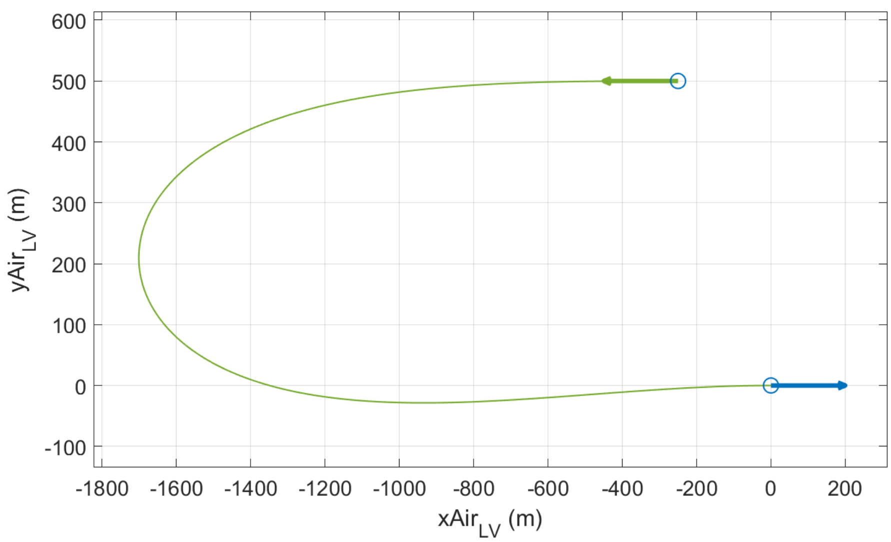

Figure 4-8: Problem geometry with $\Delta\psi = 180°$

However, running a more extensive Monte Carlo campaign which also varied the delta yaw $\Delta\psi$ between initial and final directions, it is observed that there are several geometric conditions for which the developed algorithm does not encounter an optimal solution.

These conditions are a combination of the relative position of the initial and final point and the relative direction between initial and final velocity, so there is no standard case where the algorithm does not find a solution. Certainly, one of the most problematic is the one in which $\Delta\psi$ is about 360 degrees with fairly close initial and final positions. An example is illustrated in Figure 4-9. In this case, because of the geometry of the problem, it is very complex to find a solution that meets the maximum turn rate requirements, which for parafoil are very stringent.

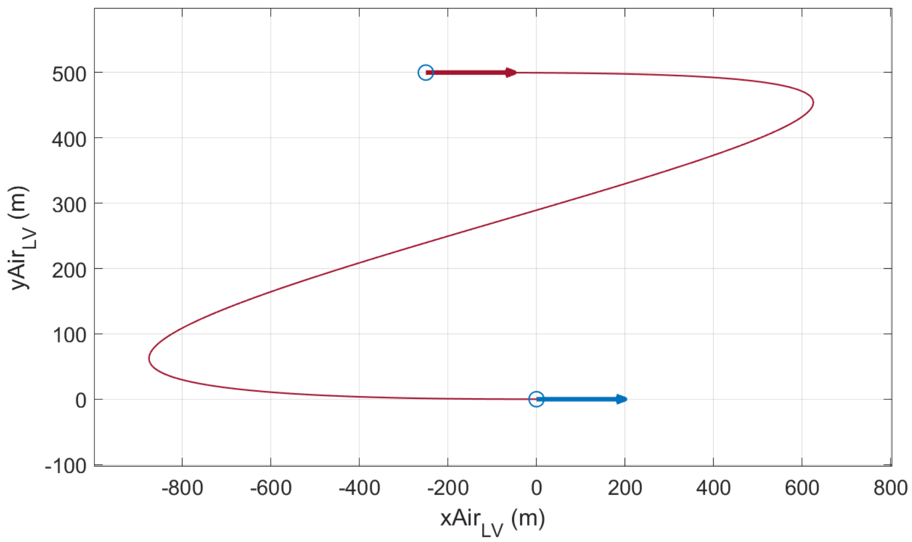

Figure 4-9: Problem geometry with $\Delta\psi = 360°$

Although there are particular geometric conditions in which an optimal solution cannot be guaranteed in any case, some measures can be taken in order to improve the performance of the algorithm. One of them is to relax the boundary conditions to broaden the range of possible acceptable solutions. In particular, in Section 4.3.1, the yaw rate $\dot{\psi}_f$ at the final point was imposed to be zero to make the arrival smoother. It is possible to relax that condition by allowing a non-zero final yaw rate but it still maintains reasonably low values. Nevertheless, the final yaw rate $\dot{\psi}_f$ is needed to define the boundary conditions so it cannot be considered as a generic free parameter but must have a specific value for the complete problem definition. One possible solution is to consider the final yaw rate $\dot{\psi}_f$ as a second parameter in the optimization process. Two-parameter optimization (TPO) certainly guarantees better results in terms of finding the optimal trajectory but with a higher computational cost that results in more time required for optimization. TPO was implemented in this work similarly to single-parameter optimization (SPO) by employing the MATLAB function *fminsearch*. The two parameters used for the optimization are:

- Virtual variable τ_f
- Final yaw rate $\dot{\psi}_f$

The optimization problem then becomes:

$$J_{opt} = \min_{\tau_f, \dot{\psi}_f} J \tag{203}$$

A comparison of the performance obtained with single-parameter and two-parameter optimization is now presented. Figure 4-10 and Figure 4-11 present a comparative example in which the two types of optimization are applied to the same problem. In particular, the initial and final positions and the final velocity are kept fixed. In contrast, the direction of the initial airspeed varies. It is noted that, for this example, groundspeed is still employed so, since the wind at the initial point is considered constant in direction and magnitude, the change in the direction of the initial airspeed causes a change in the magnitude of the initial groundspeed. It appears evident that with a TPO, the algorithm is able to find an optimal solution for a much wider range of combinations of boundary conditions. The results related to the aforementioned figures are shown in Table 6. It is observed that only 3 out of a total of 100 cases are not acceptable for two-parameter optimization while with a single parameter there are as many as 47.

Type of solution	**Single parameter number of cases**	**Two-parameter number of cases**
Optimal (green)	40/100	85/100
Sub-optimal (yellow)	13/100	12/100
Not acceptable (red)	47/100	3/100

Table 6: SPO/TPO single case comparison

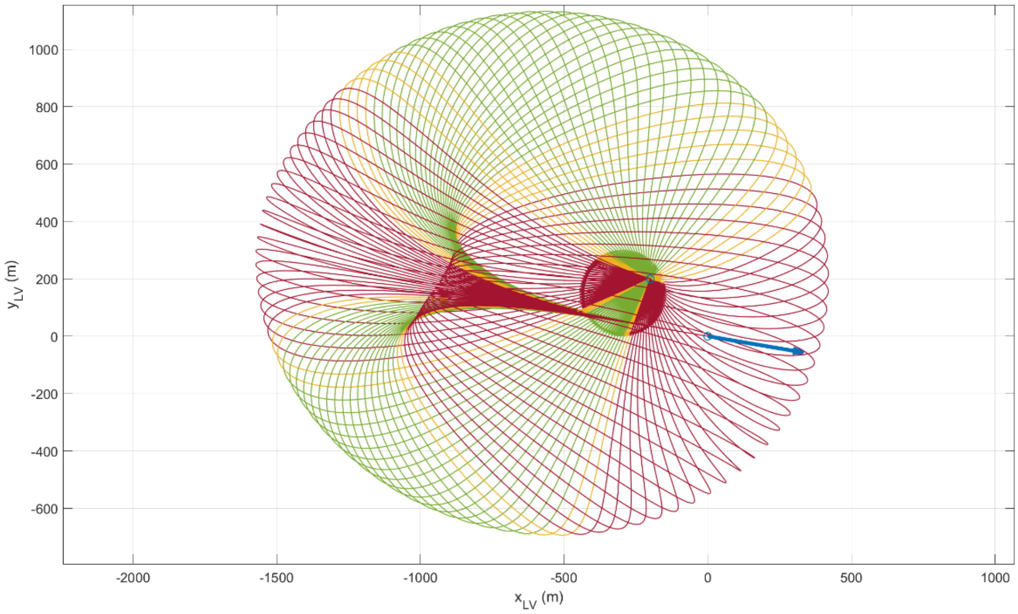

Figure 4-10: SPO trajectories

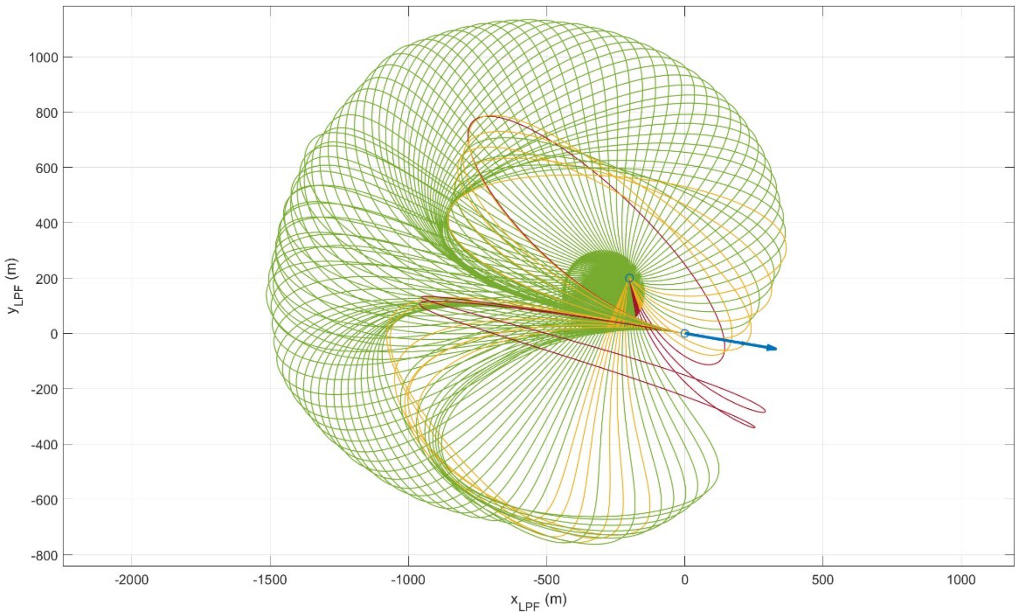

Figure 4-11: TWO trajectories

The figures below show the performance for the comparison under consideration. Figure 4-12 shows the trend of the maximum yaw rate (in logarithmic scale) as the direction of the initial velocity changes. In the case of single-parameter optimization, it is observed that there are two specific directions for which the maximum yaw rate tends to infinity. Two-parameter optimization tends to reduce the maximum yaw rate corresponding to these critical directions to bring it below the acceptable threshold. For this purpose, the TPO relaxes the final yaw rate conditions. This effect is evident in Figure 4-14 where it is observed that the final yaw rate increases precisely at the critical directions. Despite this, the final yaw rate values always remain below 1.5-2 deg/s. As discussed in Section 2.1, the Terminal Guidance phase is followed by a short corrective phase before flare activation. During that phase, it is possible to correct the yaw rate by bringing it to zero. For this reason, the relaxation of boundary conditions does not affect the performance of the PGNC. Another interesting figure is the missing time MT defined as the difference between desired maneuver time and total travel time of the trajectory given by the optimization algorithm:

$$MT = \Delta t_{tot} - T_m \tag{204}$$

In Figure 4-13, the missing time for SPO and TPO is shown. Considering that the trajectories under investigation have a target time T_m of about 300 seconds, it is observed that the missing time always remains limited and there is no particular difference between the two optimization methods. To further understand the difference in performance between the two types of optimization, a Monte Carlo campaign was carried out based on the comparison just examined but varying all boundary conditions using Gaussian distributions. The results obtained are shown in Table 7.

Type of solution	Single parameter number of cases	Two-parameter number of cases
Optimal (green)	43.16%	85.35%
Sub-optimal (yellow)	15.71%	9.38%
Not acceptable (red)	41.13%	5.27%

Table 7: SPO/TPO MC comparison

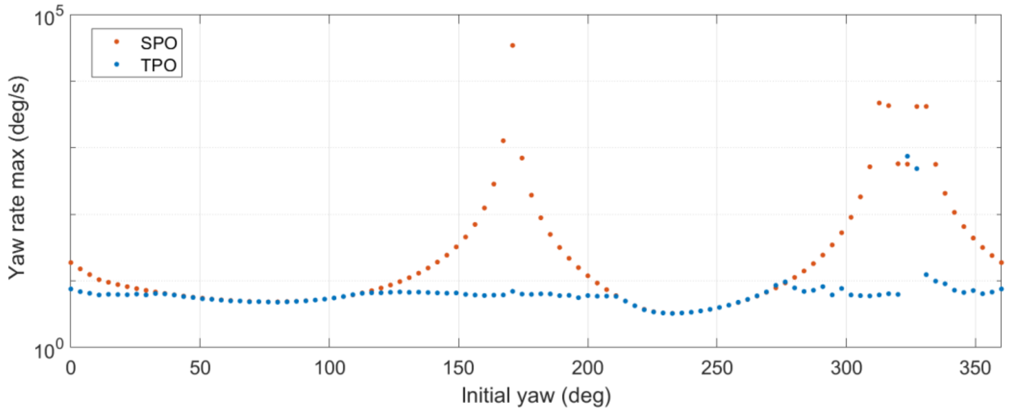

Figure 4-12: SPO/TPO max yaw rate comparison

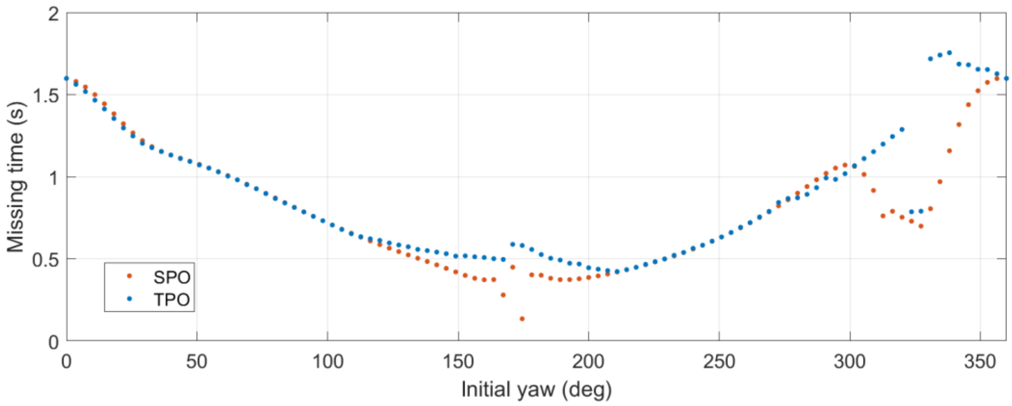

Figure 4-13: SPO/TPO missing time comparison

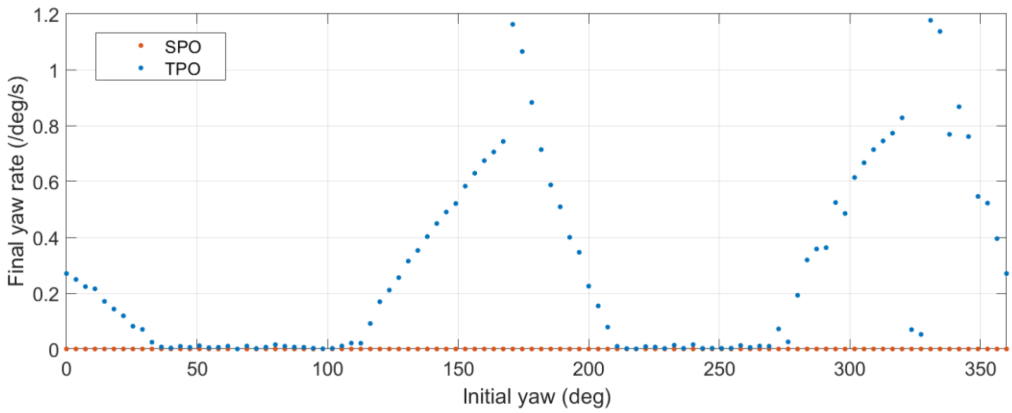

Figure 4-14: SPO/TPO final yaw rate comparison

It is emphasized that the performance so far illustrated is the result of a laborious process of tuning all parameters involved to achieve the desired output. So far, the advantage gained by using TPO appears evident. It provides a more robust approach to the variation of the problem geometry in terms of boundary conditions. However, as previously mentioned, the main drawback of TPO is related to computational cost. Figure 4-15 shows the performance in terms of time required for optimization in the MATLAB environment. The values given are relative to the performance of the machine used for the simulation and are not intended to indicate an absolute figure but to compare the different scenarios: SPO and TPO with the maximum number of iterations for the function *fminsearch* set to 50 and 100. It is observed that varying the maximum number of iterations from 50 to 100 has no noticeable improvement in terms of the maximum yaw rate. On the contrary, there are significant differences in terms of the time required for optimization. With 50 iterations the computational time always remains below 20 ms while for 100 iterations it can exceed 30 ms. In any case, the time required for SPO is significantly lower, never exceeding 10 ms. Therefore, a trade-off is required taking into account computational cost and efficiency in finding an optimal trajectory. A possible strategy is the one proposed in Chapter 8 for which SPO is used as a first approach and, in case the solution found is not acceptable, TPO with a maximum iteration limit of 50 is used.

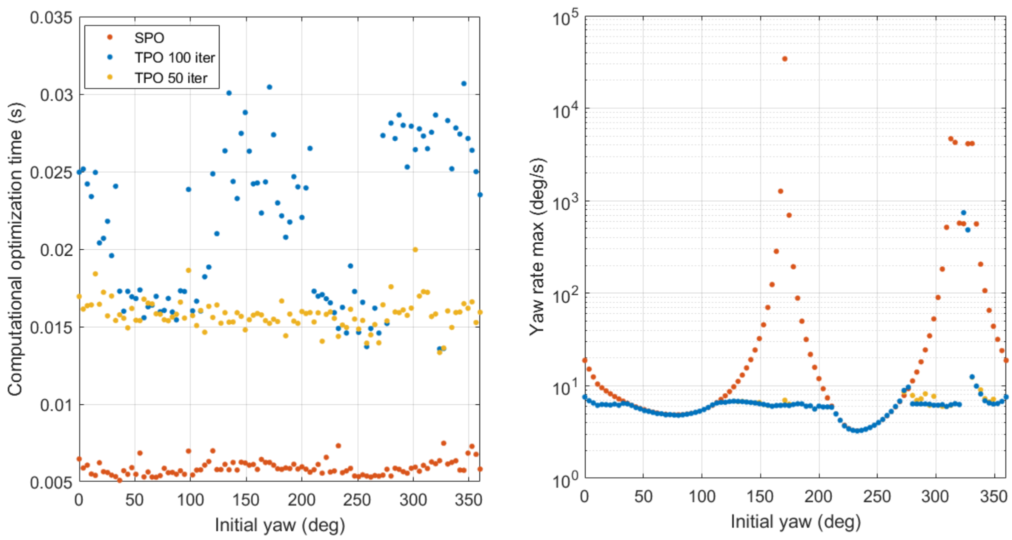

Figure 4-15: SPO/TPO computational cost

5. Trajectory Tracking Design

Chapter 4 is devoted to describing the guidance algorithm used to design the trajectory for the case study. The output of this algorithm is the path to be followed in terms of coordinates in the Wind-fixed Frame, as discussed in Section 4.4.2. The velocity considered is no longer the ground speed but the airspeed since the effect of the wind is taken into account exclusively through the wind drift. The sequence of angles that is obtained through the geometric procedure described in Section 4.3.2 (equation (187)) is therefore no longer related to the yaw angle ψ but to the heading angle of the airspeed (χ_a or HdgAir). The input required by the control is indeed the air heading rate ($\dot{\chi}_a$ or HdrAir). The heading angle of the airspeed and the air heading rate will henceforth be denoted simply by HDG and HDR, respectively. It should be recalled that, for the Terminal Guidance phase, a fixed value of FPA is considered, i.e., a constant symmetric command. Figure 5-1 illustrates the typical scenario under investigation. This chapter describes some suitable approaches to derive the HDR command from the HDG sequence provided by the guidance through different types of SISO controllers.

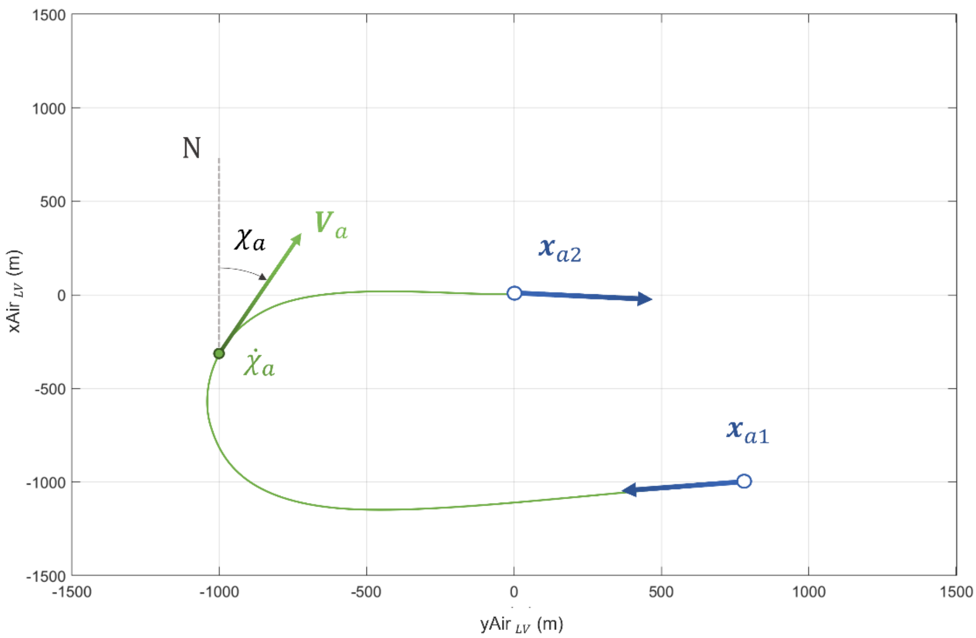

Figure 5-1: Tracking problem configuration

5.1. PID with Finite Differences

The simplest strategy consists of deriving the sequence of HDR from the sequence of HDG using finite differences (equation (189)). The reference HDR values $r\ t$ are sent to a proportional-integral-derivative (PID) controller that provides the control input $u\ t$. The control function must then process the signal obtained from the PID to obtain a command to be sent to the actuators in the form of asymmetric or left/right lines deflections (δ_r, δ_l).

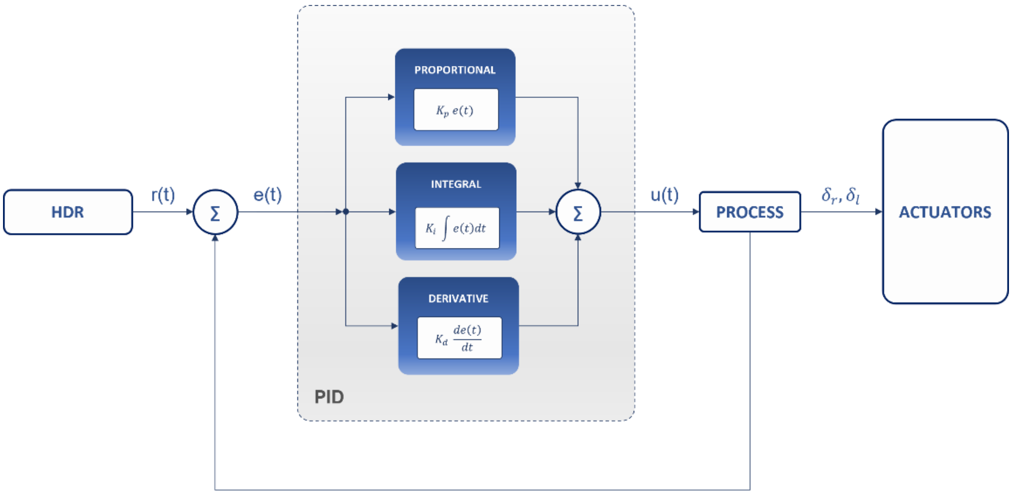

Figure 5-2: PID controller

This method has a negligible computational cost but does not directly take into account the dynamics of the payload-parafoil system. Nevertheless, the feedback employed in the control mechanism ensures that the response of the system is taken into account allowing for indirect consideration of the dynamics of the vehicle. Despite the close-loop control, the accuracy provided in trajectory tracking is poor. However, this approach can be used for an initial test of the operation of the guidance algorithm. The results obtained, as discussed later in Chapter 7, with proper tuning of the PID, can be considered acceptable to a first approximation. For more accurate results, it is required to employ more sophisticated techniques such as Model Predictive Control (MPC).

5.2. Model Predictive Control

A possible improvement in accuracy over PID is the Model Predictive Control (MPC). MPC techniques involve the use of a dynamic model of the system to predict the evolution of the system state over a finite time horizon, the so-called prediction horizon H_p. This method takes into account the dynamics of the system to predict the response to control actions in order to identify the most effective command to achieve the desired state. The MPC, at each iteration, receives the system state from the navigation, solves the Constrained Finite-Time Optimal Control (CFTOC) problem, and identifies the most suitable sequence of commands to obtain the desired output. At each iteration, only the first value of the identified sequence of commands is used. The MPC can handle more constraints on both input and output and adjust the results through an appropriate construction of the cost function. Compared to PID, the MPC requires a significantly higher computational cost that depends on the prediction horizon chosen and the dynamic model of the system. Despite the ability to handle nonlinear models, it is generally convenient to use linear models that guarantee lower computational cost and the possibility of stability, feasibility, and performance analysis. Some linearized models that can be employed in this context are described in Section 3.6. To use such models with MPC, it is necessary to convert them from continuous-time to discrete-time. Within the MATLAB environment, the dedicated *c2d* function can be utilized for this purpose. The discrete-time SISO system obtained has the following form:

$$\begin{cases} \boldsymbol{x}_{k+1} = A_d\, \boldsymbol{x}_k + B_d\, u_k \\ y_k = C_d\, \boldsymbol{x}_k \end{cases} \tag{205}$$

where A_d, B_d, and C_d are the discrete-time state-space matrices and $\boldsymbol{x}_k,\ u_k$, and y_k are respectively the state vector, the control input, and the output at the $k\text{-}th$ sampling time. The system (205) is a Linear Time-Invariant (LTI) system. The left-hand side of the state equation is no longer the derivative of the state vector but the state vector itself at the next sampling time.

According to the classical formulation of the MPC problem, given the prediction horizon H_p, it must be identified the optimal sequence $\boldsymbol{U}^*$ of control inputs over H_p:

$$\boldsymbol{U}^* = \left[u_k^*, u_{k+1}^*, \dots, u_{k+H_p-1}^*\right]^T \tag{206}$$

that minimizes the quadratic cost function:

$$J = (\boldsymbol{Y^d} - \tilde{\boldsymbol{Y}})^T Q (\boldsymbol{Y^d} - \tilde{\boldsymbol{Y}}) + \boldsymbol{U}^T R \boldsymbol{U} \tag{207}$$

such that:

$$\begin{cases} \boldsymbol{x}_{k+1} = A_d\ \boldsymbol{x}_k + B_d\ u_k \\ \quad\quad y_k = C_d\ \boldsymbol{x}_k \end{cases} \tag{208}$$

$$\boldsymbol{x}_0 = \boldsymbol{x}(0) \tag{209}$$

$$u_{min} \leq\ u_k \leq\ u_{max} \quad \forall k \tag{210}$$

$$|\ u_{k+1} -\ u_k| \leq\ \Delta u_{max} \quad \forall k \tag{211}$$

with:

$$\boldsymbol{Y^d} = \left[y_{k+1}^d, y_{k+2}^d, \dots, y_{k+H_p}^d\right]^T \tag{212}$$

$$\tilde{\boldsymbol{Y}} = \left[\tilde{y}_{k+1}, \tilde{y}_{k+2}, \dots, \tilde{y}_{k+H_p}\right]^T \tag{213}$$

$$\boldsymbol{U} = \left[u_k, u_{k+1}, \dots, u_{k+H_p-1}\right]^T \tag{214}$$

The cost function involves the reference output sequence vector $\boldsymbol{Y^d}$, the desired output sequence vector $\tilde{\boldsymbol{Y}}$, and the control input sequence vector $\boldsymbol{U}$. For the SISO case at hand, the output y is the heading while the control input u is the asymmetric line deflection:

$$y = \chi_a \tag{215}$$

$$u = \delta_a \tag{216}$$

The entity of the state vector $\boldsymbol{x}$, on the other hand, depends on the linearized model chosen.

For cost function minimization, the two positive semidefinite symmetric matrices Q and R of size H_pxH_p are introduced. Matrices Q and R allow for weighting the tracking error and control action, respectively. The MPC problem is subject to a set of conditions and constraints. The system in (208) is the LTI system that simulates plant dynamics. The expression (209) constitutes the initial condition i.e., the starting state of the system. The (210) and (211) are instead the constraints imposed on the control input. The first concerns the values that can be assumed by the control input which is the asymmetric deflection δ_a. The second instead imposes a constraint on the difference between consecutive control actions, which is equivalent to a limit on the derivative in continuous-time. It is now observed that the MPC algorithm can replace, with proper modifications, the control function already included in the GNC software bypassing it during the Terminal Guidance phase. However, this approach involves several complications in the SW design so, for this work, as shown in Figure 5-3, it was preferred to convert the control input from the MPC (δ_a) into the input signal required by the original control (HDR) without bypassing it. For this purpose, the dynamic state-space model can be used to derive the heading rate knowing the current state of the system (which includes the HDG) and the optimal control variable provided by the MPC. The formulation of the linear MPC is thus completely defined in its basic compact, flexible, and intuitive form. The advantages in the application of MPC techniques are numerous, but to obtain accurate results, a reliable validated dynamic model must be employed. In addition, the state vector from navigation must also prove to be sufficiently reliable. This is especially critical in the case of indirect measurements of state variables, measurement disturbances, and uncertainties. When using MPC it is then essential to keep in mind that the computational cost can be rather high for real-time application considering also that the problem under examination is generally non-convex. To cope with this, besides properly varying the problem formulation, it is possible to operate on the value of the prediction horizon H_p, making a trade-off between performance and computational cost. There are numerous MPC optimization algorithms available, and an entire toolbox dedicated to the purpose is available within the MATLAB environment.

5.3. Linear Quadratic Regulator

Optimization algorithms used for MPC are nowadays very efficient, but the computational cost still cannot fall below a certain threshold because of the way the problem is formulated. In some cases, it is possible to drastically reduce the computational cost by using a Linear Quadratic Regulator (LQR), which is nothing more than a special case of unconstrained MPC. The formulation of the problem is the same as that seen in Section 5.2 but without the constraints of equations (210) and (211). In this case, an analytical solution to the problem can be found without resorting to specific optimization algorithms. The resolution method for LQR is briefly discussed below.
Denoting by $\boldsymbol{x}(k)$ and $u(k)$ (instead of $\boldsymbol{x}_k$ and $\boldsymbol{u}_k$) the discrete-time state vector and control input at the k-th sampling time, using the first equation of the (205), it results:

$$\begin{aligned}
&\boldsymbol{x}(k+1) = A_d\boldsymbol{x}(k) + B_d u(k) \\
&\boldsymbol{x}(k+2) = A_d\boldsymbol{x}(k+1) + B_d u(k+1) = A_d(A_d\boldsymbol{x}(k) + B_d u(k)) + B_d u(k+1) \\
&\qquad\qquad\quad = {A_d}^2\boldsymbol{x}(k) + A_d B_d u(k) + B_d u(k+1) \\
&\boldsymbol{x}(k+3) = A_d\boldsymbol{x}(k+2) + B_d u(k+2) = \cdots \\
&\qquad\qquad\quad = {A_d}^3\boldsymbol{x}(k) + {A_d}^2 B_d u(k) + A_d B_d u(k+1) + B_d u(k+2) \\
&\boldsymbol{x}(k+4) = \cdots
\end{aligned} \tag{217}$$

It is noted that the state $\boldsymbol{x}$ always depends solely on the initial state $\boldsymbol{x}(k)$ and the control input sequence applied to the previous steps. The time evolution of the state vector can therefore be predicted as follows:

$$\widetilde{\boldsymbol{X}} = \begin{bmatrix} \tilde{\boldsymbol{x}}(k+1) \\ \tilde{\boldsymbol{x}}(k+2) \\ \dots . \\ \tilde{\boldsymbol{x}}(k+H_p) \end{bmatrix} = \begin{bmatrix} \tilde{\boldsymbol{x}}_{k+1} \\ \tilde{\boldsymbol{x}}_{k+2} \\ \dots . \\ \tilde{\boldsymbol{x}}_{k+H_p} \end{bmatrix} = \mathcal{A}\boldsymbol{x}_k + \mathcal{B}\boldsymbol{U}$$

$$= \begin{bmatrix} A_d \\ {A_d}^2 \\ \vdots \\ {A_d}^{H_p} \end{bmatrix} \boldsymbol{x}_k + \begin{bmatrix} B_d & 0 & 0 & \cdots & 0 \\ A_d B_d & B_d & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ {A_d}^{H_p-2} B_d & {A_d}^{H_p-3} B_d & A_d B_d & \cdots & 0 \\ {A_d}^{H_p-1} B_d & {A_d}^{H_p-2} B_d & {A_d}^{H_p-3} B_d & \cdots & B_d \end{bmatrix} \begin{bmatrix} u_k \\ u_{k+1} \\ \cdots \\ u_{k+H_p-1} \end{bmatrix} \tag{218}$$

where the tilde mark denotes the estimated quantities. Now, taking into account that the second equation of (205) gives:

$$y_k = C_d \; \boldsymbol{x}_k \tag{219}$$

and following a procedure similar to the one just seen yields:

$$\begin{gathered}\tilde{\boldsymbol{Y}} = \begin{bmatrix} \tilde{y}(k+1) \\ \tilde{y}(k+2) \\ \cdots . \\ \tilde{y}(k+H_p) \end{bmatrix} = \begin{bmatrix} \tilde{y}_{k+1} \\ \tilde{y}_{k+2} \\ \cdots . \\ \tilde{y}_{k+H_p} \end{bmatrix} = \bar{\mathcal{A}}\boldsymbol{x}_k + \overline{\mathcal{B}}\boldsymbol{U} \\ = C_d \begin{bmatrix} A_d \\ {A_d}^2 \\ \vdots \\ {A_d}^{H_p} \end{bmatrix} \boldsymbol{x}_k + C_d \begin{bmatrix} B_d & 0 & 0 & \cdots & 0 \\ A_d B_d & B_d & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ {A_d}^{H_p-2} B_d & {A_d}^{H_p-3} B_d & A_d B_d & \cdots & 0 \\ {A_d}^{H_p-1} B_d & {A_d}^{H_p-2} B_d & {A_d}^{H_p-3} B_d & \cdots & B_d \end{bmatrix} \begin{bmatrix} u_k \\ u_{k+1} \\ \cdots \\ u_{k+H_p-1} \end{bmatrix}\end{gathered} \tag{220}$$

The formulation of the LQR problem then requires finding the optimal control input sequence $\boldsymbol{U}^*$ over the entire prediction horizon H_p that minimizes the cost function:

$$\begin{aligned} J &= (\boldsymbol{Y^d} - \tilde{\boldsymbol{Y}})^T Q (\boldsymbol{Y^d} - \tilde{\boldsymbol{Y}}) + \boldsymbol{U}^T R \boldsymbol{U} \\ &= (\boldsymbol{Y^d} - \bar{\mathcal{A}}\boldsymbol{x}_k + \overline{\mathcal{B}}\boldsymbol{U}\ )^T Q (\boldsymbol{Y^d} - \bar{\mathcal{A}}\boldsymbol{x}_k + \overline{\mathcal{B}}\boldsymbol{U}\ ) + \boldsymbol{U}^T R \boldsymbol{U} \end{aligned} \tag{221}$$

that is:

$$\boldsymbol{U}^* = arg \min_{\boldsymbol{U}} J(\boldsymbol{x}_k, \boldsymbol{U}) \tag{222}$$

with:

$$J^* = \min_{\boldsymbol{U}} J(\boldsymbol{x}_k, \boldsymbol{U}) \tag{223}$$

In the case of the Linear Quadratic Regulator, it can be shown that the vector $\boldsymbol{U}^*$ can be derived analytically as follows:

$$\boldsymbol{U}^* \ = \left(\left(\overline{\mathcal{B}}^T Q \overline{\mathcal{B}} + R \right)^{-1} \ \overline{\mathcal{B}}^T Q \right) (\boldsymbol{Y^d} - \bar{\mathcal{A}}\boldsymbol{x}_k) = \mathcal{F} (\boldsymbol{Y^d} - \bar{\mathcal{A}}\boldsymbol{x}_k) \tag{224}$$

Once the optimal control vector has been derived, at each control step, the first element can be extracted and used as the control input value. Following the diagram in Figure 5-3, this value can be processed and sent directly to the actuators as a command in the form of left/right line deflection. This procedure bypasses the original PGNC Control (PCON) whose functions are replaced in their totality by the LQR and the associated processing functions. In contrast, the approach used for this study involves processing the output of the LQR to obtain the HDR signal required by the original PCON, which is therefore not bypassed. The HDR value is derived by introducing the current state $\boldsymbol{x}$ of the system and the output δ_a of the LQR within the linearized continuous-time state-space system from which the discrete-time system was derived. An alternative but less accurate procedure might be to use a lookup table (LUT) to derive the HDR. By means of a series of open-loop simulations, it is indeed possible to analyze the dynamic response in terms of HDR to a δ_a command and construct a table that, based on the state of the system, univocally provides the value of HDR. The table generated must take into account a set of main parameters such as the mass of the system and the density of the air. Figure 3-3 shows visually a portion of the structure of the tables under consideration. This figure should be extended to more dimensions to consider the selected system parameters. This method is quite common in GNC design but provides less accurate results than using the dynamic model because it is not possible to account for all the parameters involved.

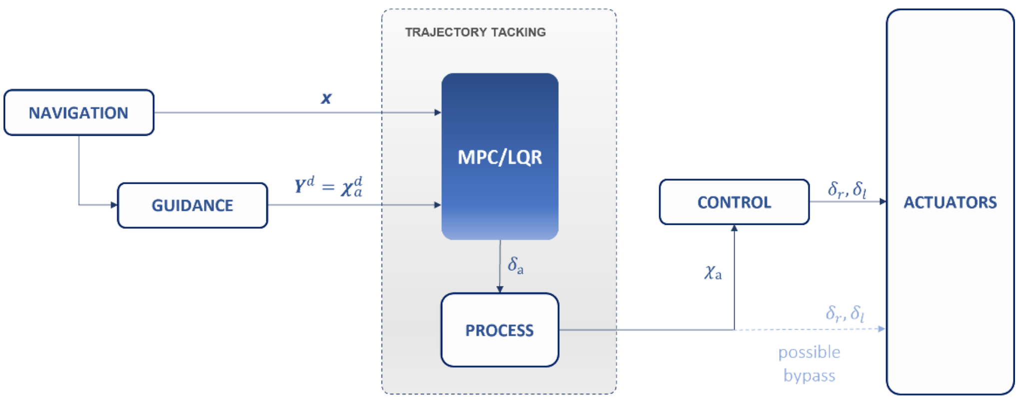

Figure 5-3: LQR/MPC implementation

For this study, LQR was selected as the trajectory tracking method. This choice is attributable to two main reasons. The first is related to the low computational cost which is a crucial element for the problem under investigation. The second, on the other hand, partly justifies the use of the LQR instead of the MPC. The heading sequence coming from the driving algorithm that is used by the LQR already takes into account the constraint in terms of the values that can be assumed by the command and the maximum turn rate. The imposition of a new constraint is therefore redundant. Despite this, it is always good practice to provide a saturator for the commands, associated, in the case of PID controllers, with an anti-windup system to avoid possible issues with the integral term in case of saturation of the command.
To analyze the performance of the LQR, a simple 3DOF model such as the one described in Section 3.5.1 was developed in the Simulink environment. The purpose of this analysis is mainly to investigate the evolution of the output of the trajectory tracking algorithm without focusing on the result in terms of the path followed. Figure 5-4 shows the comparison of the HDR sequences obtained with finite differences and LQR. In particular, for LQR, the two different methods seen to derive HDR from the asymmetric deflection δ_a were considered.

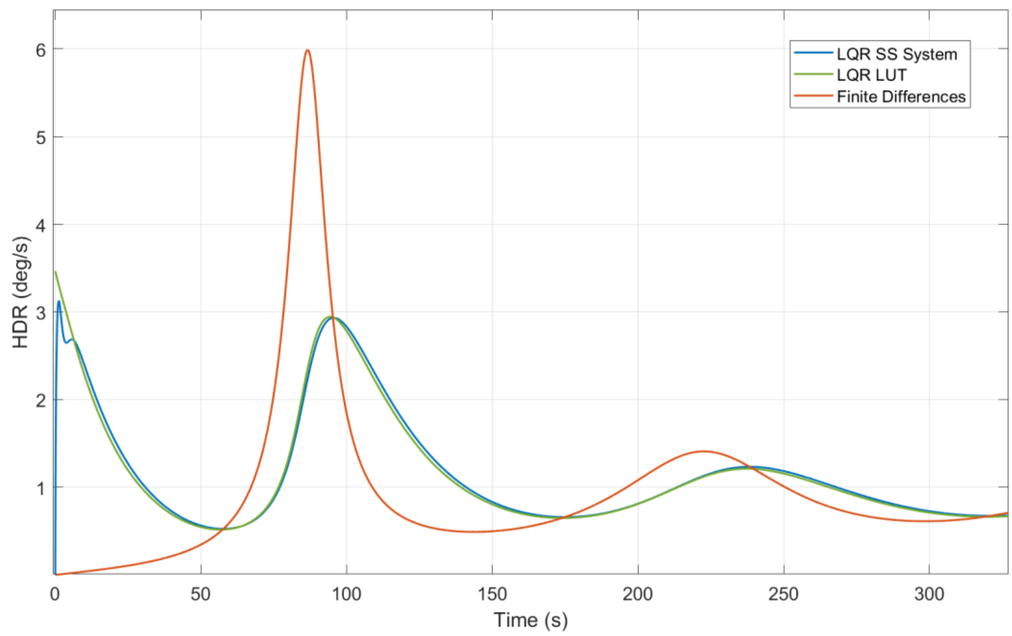

Figure 5-4: HDR sequence with LQR and Finite Differences

The first evident result obtained is that the HDR estimation performed by the finite difference method is much greater than that by LQR. This element is an additional point in favor of using LQR instead of MPC. In fact, it indicates that the criteria used for the limitation in terms of maximum HDR within the guidance algorithm are actually more stringent than expected ensuring a certain margin of safety. It is also observed that the LQR trend exhibits an initial peak that makes the whole sequence appear very different from the finite difference one. Comparing the two methods used to derive the HDR for the LQR shows that the two sequences are quite similar except for the initial stage of the sequence. The use of the space-state model involves considering the dynamic evolution of the system, which therefore starts (in this case) from a null HDR command. On the other hand, the LUT method immediately associates a certain value of δ_a with the corresponding value of HDR without considering the current state of the system. A further analysis that can be performed concerns the selection of the prediction horizon. In general, the choice of a longer horizon allows the evolution of the system to be considered over a longer period of time but with a higher computational cost. In addition, a larger H_p clashes with the assumptions made for the linearization providing less accurate results. A trade-off was carried out to find the most suitable value of H_p, which was set to 30. Figure 5-5 shows the HDR sequences related to LQR with LUT as H_p varies.

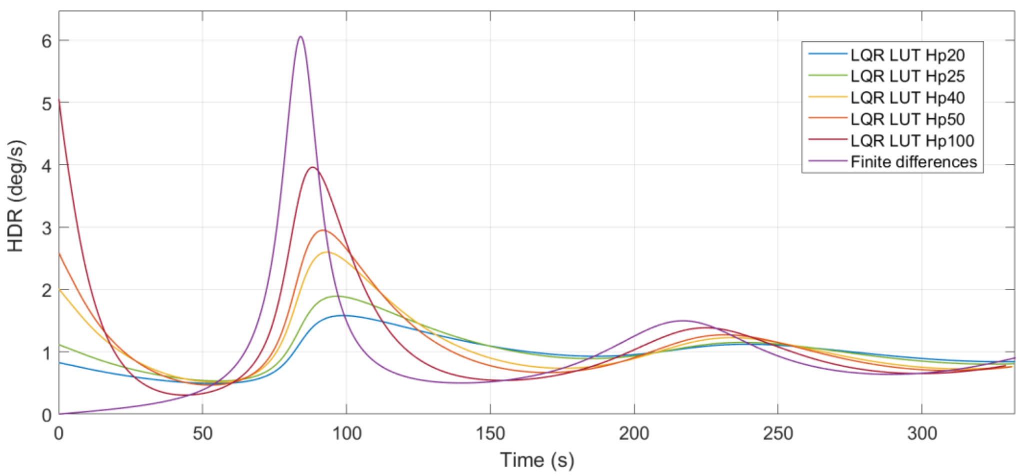

Figure 5-5: HDR sequences varying the prediction horizon

6. TERGUID Functional Architecture

So far, the algorithms for the generation of the optimal trajectory and its tracking have been developed. These algorithms must be properly incorporated into the complete PGNC code to be able to carry out full simulations with the 6DOF simulator provided by SENER Aeroespacial. For this purpose, a complete logic for the entire Terminal Guidance phase was first designed, then implemented in a set of MATLAB functions that were incorporated into the complete PGNC code. The architecture of the code for implementing the developed algorithms is a rather delicate task because it has to take into account all the interfaces with the rest of the code, but also possible issues related to the operation of the algorithms. Therefore, it is essential to possess a complete overview of the architecture and operation of the entire GNC code besides having a clear understanding of the mission profile and the challenges associated with it. Two different logics have been developed for this study. One employs finite differences for HDR sequence computation and the other employs LQR. The following sections only concern the second case, describing both the logic and its implementation strategy.
Figure 6-1 shows the general architecture adopted for the PGNC functions developed to manage the Terminal Guidance phase. As seen in Section 2.3, the parafoil GNC (PGNC) algorithms are part of the more general framework of the complete GNC software of the vehicle. The PGNC can be considered as a submode of the GNC activated in correspondence with parafoil deployment and composed in turn of specific submodes. The PGNC is composed of the following main functions:

- Parafoil Navigation (PNAV)
- Parafoil Mode Manager (PMNG)
- Parafoil Guidance (PGUI)
- Parafoil Control (PCON)

For the purpose of this study, the PNAV function already implemented in the design developed by SENER Aeroespacial was not modified. In contrast, the code of PMNG, PGUI, and PCON related to the TERGUID phase was properly edited to implement the algorithms seen in the previous chapters. In particular, a set of functions has been developed for generating the trajectory and its tracking (section 6.1), for managing the triggering and exit from the TERGUID phase (section 6.2), and for handling the general operation of the TERGUID through the logic designed for this phase. It should also be recalled that all PGNC functions are called with a frequency of 2.5 Hz except for the guidance functions, which are characterized by a frequency of 25 Hz. The PGUI is therefore activated every 10 cycles of complete PGNC.

Figure 6-1: PGNC TERGUID architecture

6.1. Trajectory Generation & Tracking Implementation

This section describes the implementation of the guidance and trajectory tracking algorithms seen in the previous two chapters (4 and 5). In addition to them, to make the adopted strategy sufficiently robust, a further algorithm was included i.e., a backup solution for trajectory generation.

6.1.1. Optimal Solution Search

The real-time optimal trajectory search is based on four low-level functions that are called sequentially:

a. *GetOptSolution*
b. *GetOptTrajectory*
c. *GetOptHdrSequenceFD*
d. *CheckOptSolution*

The sequence begins with the function *GetOptSolution*, which aims to minimize the cost function J using the MATLAB function *fminsearch*. For this purpose, a specific function was developed to implement the cost function J described in Section 4.4.1 that must be properly called within the function *fminsearch*. The output of the GetOptSolution function is constituted by the optimal values of the two optimization parameters selected (virtual variable τ_f and final yaw rate $\dot{\psi}_f$). These values are then used by the *GetOptTrajectory* function to define the optimal trajectory in terms of coordinates in WF. This function allows the coefficients a_m^η and b_n^η of equation (141) to be identified so that an analytical expression of the solution can be defined. From this, the HDG sequence needed to obtain the HDR sequence is derived. This last step is performed by the *GetOptHdrSequenceFD* function, which derives HDR values by making use of the finite difference method. The obtained sequence is not used as a control input (given by the LQR) but only to check the status of the solution using the *CheckOptSolution* function according to the criteria presented in Section 4. This choice is due to the lower computational cost of the finite difference method compared to LQR taking into account that the former is also more conservative.

6.1.2. Backup Solution

The proposed guidance algorithm succeeds in ensuring an optimal solution very efficiently where a physical solution to the problem is guaranteed to exist. There are, however, some cases where it does not exist. This can occur because of the initial conditions for the Terminal Guidance phase, due to issues related to previous phases, or as a result of wind drift during the TERGUID phase itself. Figure 6-2 helps to understand the basic conditions for which the problem does not possess physical solution. Considering constant horizontal and vertical airspeed norms, it is possible to define the following time parameters, named vertical and horizontal time respectively:

$$T_{vertical} = \frac{d_{vertical}}{V_{va}} \tag{225}$$

$$T_{horizontal} = \frac{d_{horizontal}}{V_{ha}} \tag{226}$$

where $d_{vertical}$ and $d_{horizontal}$ are the vertical and horizontal distance of the vehicle from the landing point in the Wind-fixed Frame (WF), assuming that the final point used for the guidance algorithm is the LP.

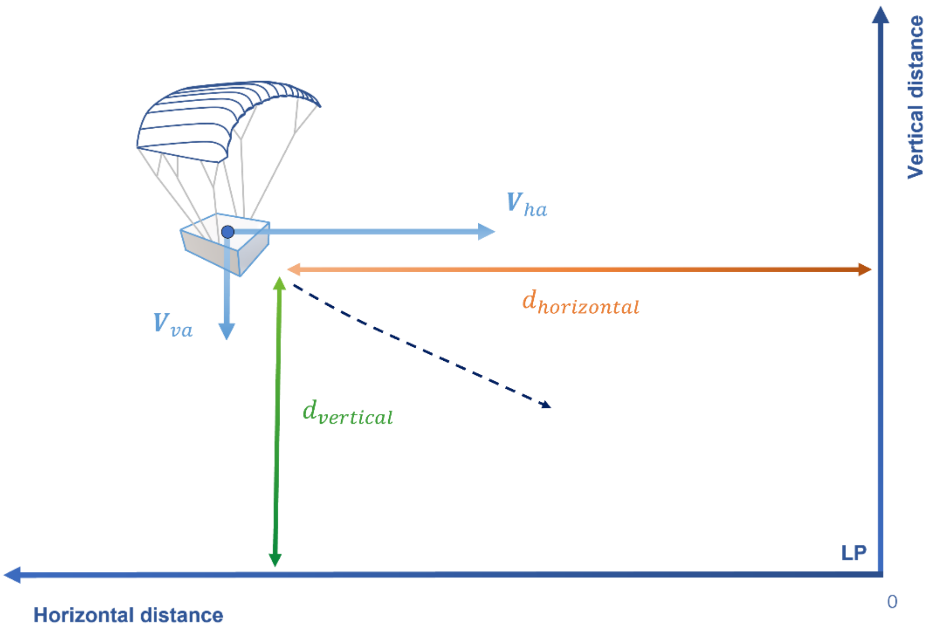

Figure 6-2: Vertical and horizontal time

The necessary but not sufficient condition for a physical solution of the Terminal Guidance problem to exist is the following:

$$T_{vertical} \geq T_{horizontal} \tag{227}$$

This condition is not sufficient since for an optimal solution to exist, a certain margin of time difference ΔT between $T_{vertical}$ and $T_{horizontal}$ is also necessary to plan a trajectory that meets the maximum turn rate requirements. The value of ΔT depends on the boundary conditions of the problem so it is possible that (227) is met but no solution to the problem exists. The standard trend of vertical and horizontal times is shown in Figure 6-3 where it is always observed the presence of some margin ΔT which tends to zero as the final point of the trajectory is approached.

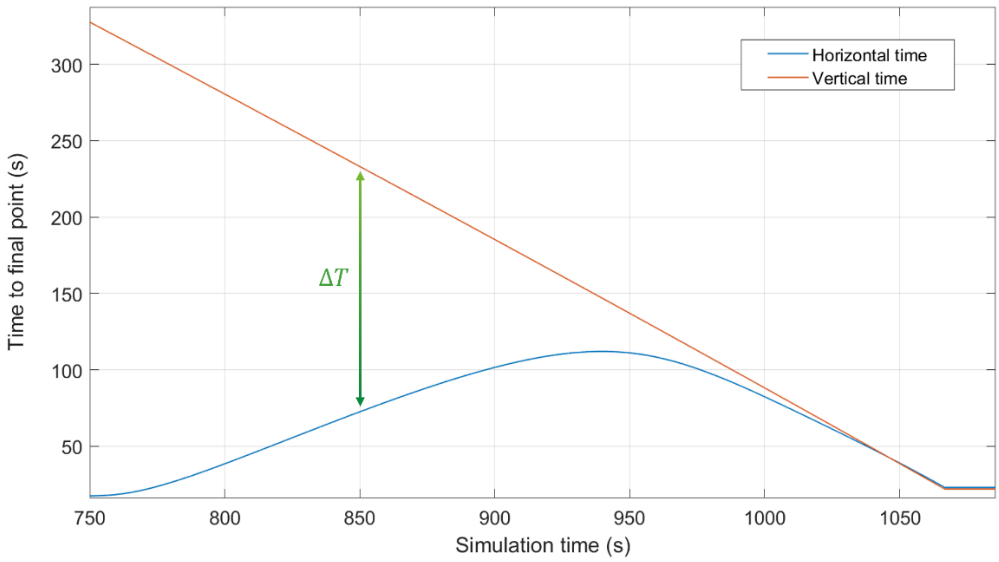

Figure 6-3: Vertical and horizontal time trends

Given the observations just made, it appears clear the need to provide a backup approach in case an optimal solution is not available. The objective of this approach is to limit the distance to the LP at touchdown as much as possible in order to avoid possible hazardous conditions in terms of mission safety requirements. The maneuver that is generally adopted as a backup solution is what is usually referred to as Point to Target. The trajectory generated in the horizontal plane is a simple straight line joining the current position of the vehicle in WF and the LP. Thus, neither the direction of the vehicle's current speed nor the desired final speed is taken into account in planning the backup trajectory. This element has a severe impact on touchdown speed requirements that cannot generally be guaranteed. This solution is therefore critical to the fulfillment of the requirements and should be adopted only if it is actually needed. It is crucial to design a robust logic for activating the backup solution when required (see Section 6.3). Figure 5 compares 3 different solutions for the Terminal Guidance trajectory as the initial horizontal position changes (with the same initial altitude). For cases 1 and 2, it is observed that the horizontal distance is still within the limits that guarantee the presence of an optimal trajectory. In the third case, on the other hand, the horizontal distance is too large and a Point to Target solution is used. The trajectory depicted is the one calculated by the guidance and not the one traveled by the vehicle. Given the absence of a physical solution, the touchdown point will be located along it and not exactly at the LP.

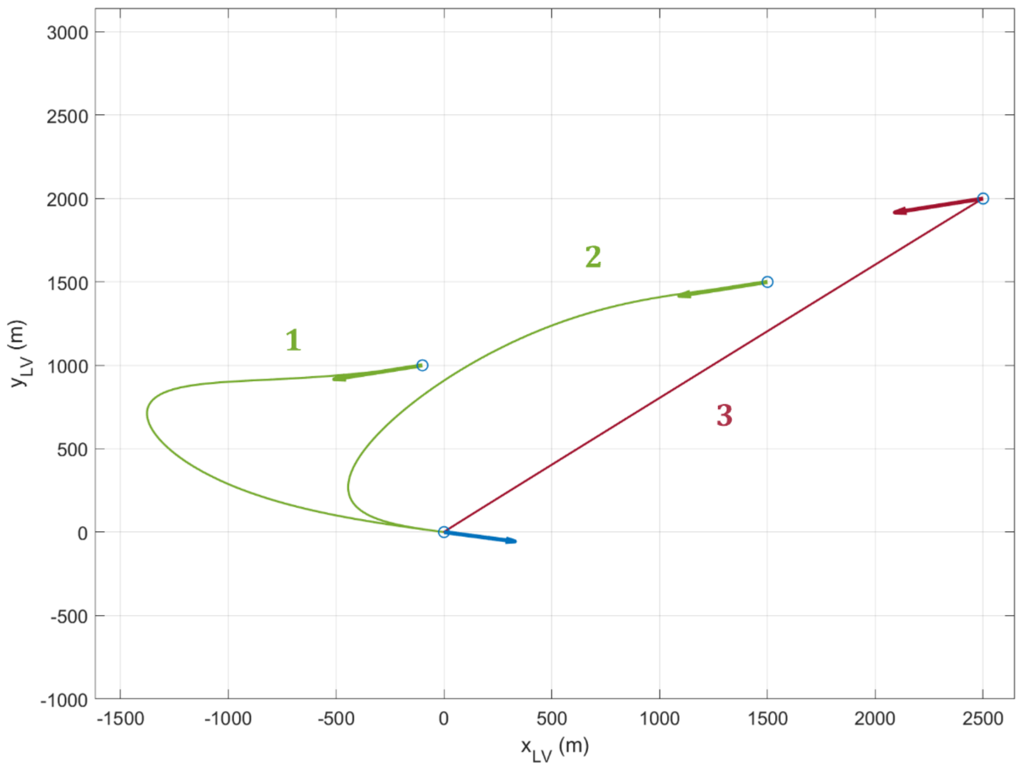

Figure 6-4: Optimal and backup solutions

6.1.3. Trajectory Tracking

Once the trajectory to be followed has been calculated, it is necessary to implement a control function to track it. As seen in Section 5, there are several possible approaches. The use of Model Predictive Control techniques with constraints on the output value is certainly preferable in case there are no special limitations on the computational cost. However, for the case under consideration, the computational cost is limited and, in addition, constraints are imposed downstream during the trajectory generation. For this reason, a Linear Quadratic Regulator was selected for trajectory tracking. The LQR solves an unconstrained minimization problem over a finite time horizon.
The algorithm described in Section 5.3 was implemented in a function named *GetOptAsymDeflectionLQR* that receives as input the state vector from the navigation and the HDG sequence to follow from the guidance and provides as output the optimal asymmetric deflection value. The obtained value δ_a must then be converted into the heading rate value HDR of the airspeed vector, which consists of the input required by the original PCON. The *GetOptHdrLQR* function performs this conversion using the linearized dynamic system. The use of LQR is scheduled for each PGNC cycle in which an optimal or sub-optimal trajectory is available. In the event of backup solution activation instead, LQR no longer becomes needed. In fact, for a Point to Target-type trajectory, it is in general sufficient to use a simple Proportional–Integral–Derivative controller characterized by low computational cost and sufficiently accurate results. For the backup solution, the trajectory tracking simply requires following a straight path which results in constant HDG. Even in the case of PID use, it is necessary to obtain as a final output the value of HDR to be sent to the original PCON functions. LQR and PID are activated one at a time based on whether or not the backup solution has been triggered. Downstream of both the LQR and the PID, it is then essential to ensure the presence of a saturator in order to guarantee that the command value does not exceed the design limits. In the case of this work, this element is already included within the PCON function.

6.2. Submode Transitions

An additional important function to be implemented for the correct operation of the code is the triggering of the different submodes. In this case, a first function was developed to exit the Energy Management (ENEMNG) submode and activate the terminal guidance (TERGUID) submode, and a second one to exit the TERGUID submode and enter the final corrections submode. The ENEMNG exit function is active during all and only the ENEMNG phase. For the exit from this phase, the following condition must first be met:

$$h \leq h_{ThrsE2T} \tag{228}$$

where $h_{ThrsE2T}$ is the altitude threshold required for Terminal Guidance activation. For the case study, TERGUID cannot be activated above 1500 meters. The other required condition is the availability of an optimal solution. The high-level function for exiting ENEMNG, at each PGNC cycle, executes the sequence of Section 6.1.1 to search for an optimal solution. The last function of the sequence (*CheckOptSolution*) returns a flag indicating the type of solution found. If the solution is optimal (see Section 4.4 for the criteria to be met) and the altitude limit is respected, the PMNG activates the TERGUID submode. In summary, the conditions to be met simultaneously for TERGUID activation are as follows:

- Current phase: ENEMNG
- Current altitude $< h_{ThrsE2T} = 1500\ m$
- Optimal TERGUID solution available

It is important then to take into account the eventuality in which the conditions just presented are never met during the ENEMNG phase. In this case, it is necessary to introduce an altitude threshold after which the TERGUID submode is forced even without the required conditions being met. The Terminal Guidance submode is then automatically activated when:

$$h \leq h_{LimE2T} \tag{229}$$

where the altitude limit h_{LimE2T} is set at 800 meters. In case of forced triggering of TERGUID submode, there is an initial absence of optimal trajectory, so the backup solution is activated. It is possible, however, that under particular conditions, for example, due to wind drift, the optimal solution is recovered during the Terminal Guidance phase. Therefore, the search for the optimal solution remains active anyway. Once the TERGUID submode is activated, the PMNG begins to execute the TERGUID exit function. The conditions imposed for exiting the TERGUID and starting the final corrections phase are as follows:

$$h \leq h_{ThrsT2FC} \tag{230}$$

$$\Delta HDG \leq \Delta HDG_{ThrsT2FC} \tag{231}$$

where $\Delta HDG_{ThrsT2FC}$ is the angular difference between the current heading and the target final heading. The value chosen for this parameter is 10 degrees. This is because the Final Corrections submode can easily correct heading differences of this order of magnitude. The altitude threshold $h_{ThrsT2FC}$ is 400 m. If the conditions are not met, the Final Corrections submode is not activated and there is a direct transition from TERGUID to flare (always activated).

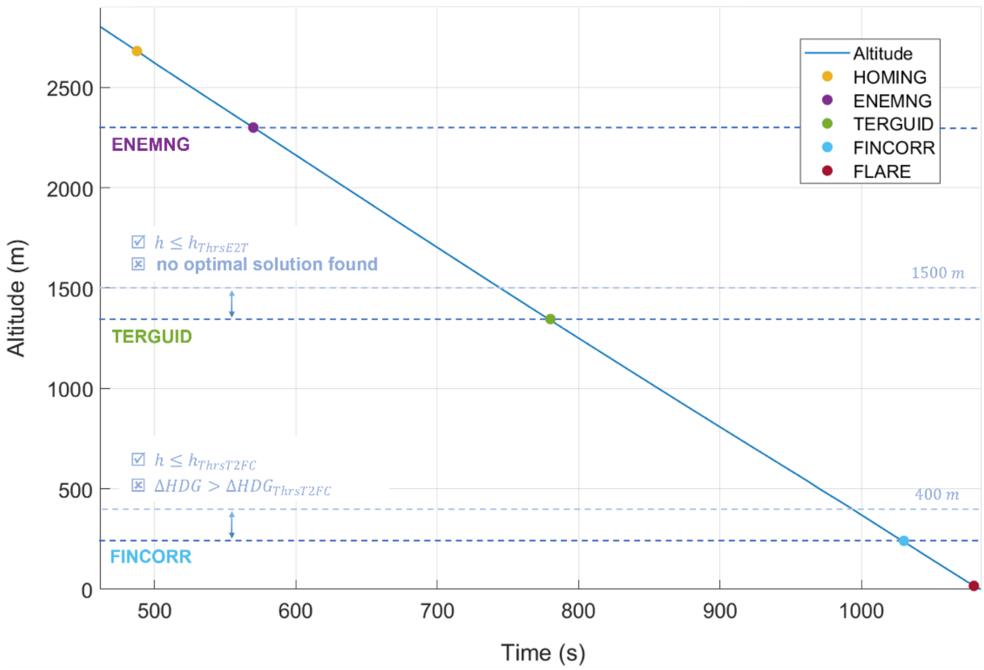

Figure 6-5: Submode triggering examples

6.3. TERGUID Logic

When the TERGUID submode is active, it is needed to properly manage the different functions taking into account several factors, such as the different frequencies, the computational cost, and the mission profile. The PMNG works at the main frequency of the PGNC, which is 25 Hz. During the Terminal Guidance phase, the TERGUID exit function is performed at each PGNC cycle to check whether the output conditions are verified. The LQR and PID functions introduced downstream of the original PCON must also operate at the frequency of the PGNC because at each cycle the control must send a command signal to the actuators. In contrast, the PGUI works at a frequency ten times lower (2.5 Hz). Figure 6-6 shows the logic designed for the PGUI functions used during the TERGUID phase. A fundamental concept of the logic design is that the computation of the optimal trajectory does not occur at every PGUI cycle. The reason for this choice is not only related to the CPU load budget. Indeed, tests with the 6dof simulator have shown that too frequent updating of the trajectory to be followed can be counterproductive in many cases. In fact, the dynamics of a large parafoil is characterized by a rather slow response that causes the vehicle to be unable to follow frequent changes in trajectory. This problem is worsened by the fact that using a two-parameter optimization the shape of the solution can vary significantly even with a slight change in boundary conditions. In particular, keeping the boundary conditions fixed except for small changes in the direction of the initial velocity, in some cases, a kind of discontinuity in the shape of the solution can be observed. This phenomenon is referred to as branching and can cause several issues in guidance strategy so it must be handled carefully. Figure 6-7 clearly illustrates this phenomenon. A sudden change in the shape of the solution is observed between the two groups of trajectories (1 and 2) due to a slight change in the direction of the initial velocity. This phenomenon is not observed in the case of single-parameter optimization which, however, as seen previously, is much less efficient in the search for an optimal solution.

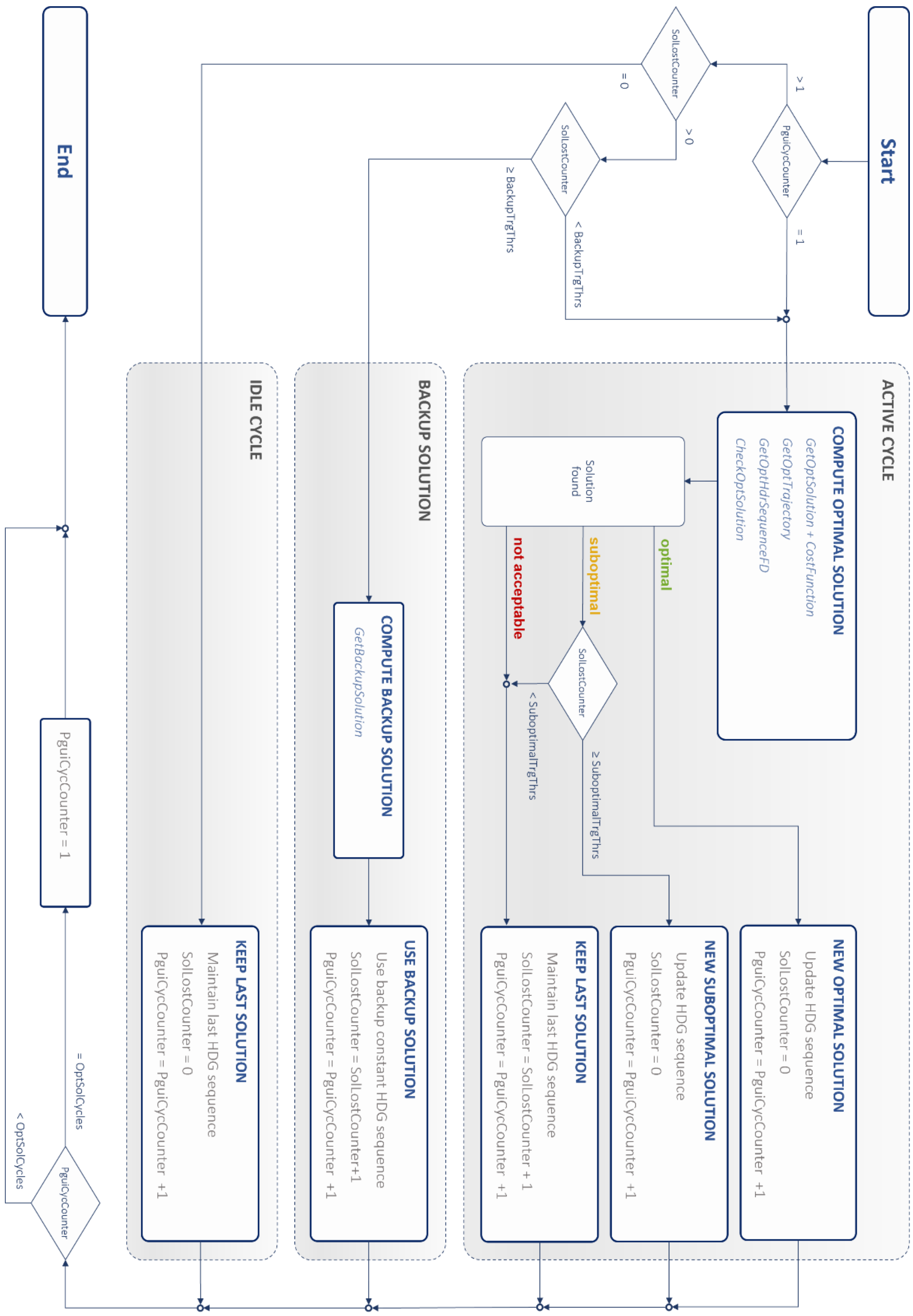

Figure 6-6: TERGUID PGUI logic

During a simulation, for every PGUI cycle, in addition to a change in direction of the initial velocity, there is also a change in initial position so the shape of the trajectory can change much more easily. For this reason, the most convenient approach for trajectory calculation is to perform it at a lower frequency than PGUI. A baseline computation frequency of 1/6 Hz was chosen for the case study. This means that the trajectory is calculated every 15 cycles of PGUI (every 6 seconds). To do this, a counter called *PguiCycCounter* is used. It increases by one unit at each PGUI cycle and resets to the value 1 when it reaches the *OptSolCycles* value, which is equal to 15.

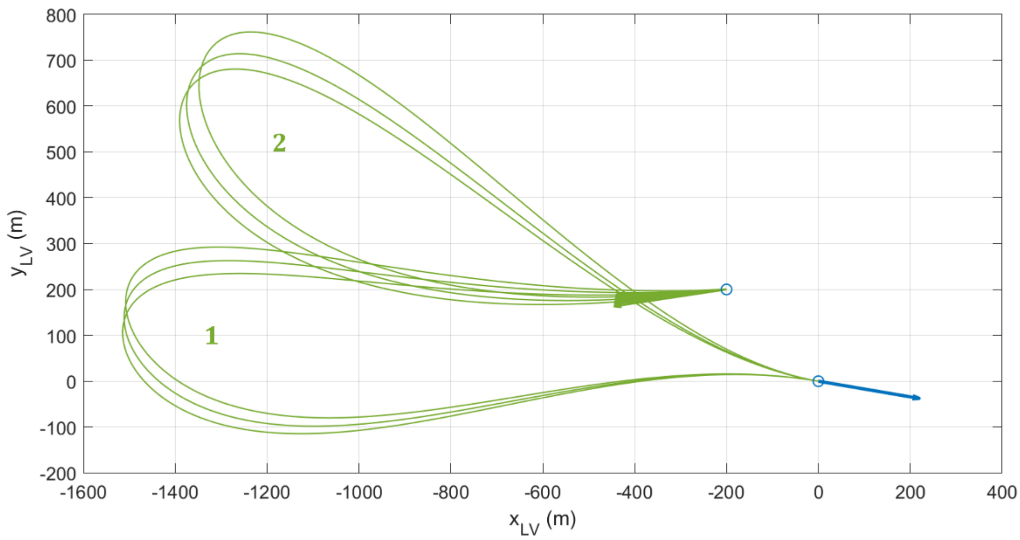

Figure 6-7: Trajectory generation discontinuities

When the value of *PguiCycCounter* is equal to 1 the sequence illustrated in Section 6.1.1 for trajectory generation is activated. This case is referred to as an "active cycle". The *CheckOptSolution* function provides a flag with the category of the solution found, which can be:

- Optimal solution (green)
- Sub-optimal solution (yellow)
- Not acceptable solution (red)

The structure of the logic considering the different cases is now described according to the diagram in Figure 6-6.

In the case where an optimal trajectory has been found, the HDG sequence identified is stored in the memory. This sequence will be used for all subsequent PGNC cycles until the trajectory is updated again. The LQR works at the frequency of PGNC (25 Hz) so, taking into account that the trajectory is calculated every 15 cycles of PGUI (which works at 2.5 Hz), the saved HDG sequence will be used for 150 cycles of PCON. Such cycles are referred to as "idle" because they do not involve the computation of a new trajectory. It is important to properly manage the HDG sequence during idle cycles so that the LQR receives as input the HDG vector relative to the correct prediction horizon H_p.
In case the solution found is not acceptable, the sequence of HDG found in the previous active cycle is maintained. Here, a new counter, called *SolLostCounter*, comes into play, which is used to count the number of cycles that have occurred since the last solution update (not including idle cycles). If at the previous cycle the *SolLostCounter* assumes a non-zero value, the current cycle will surely be an active cycle, regardless of the value of *PguiCycCounter*. This means that until a new feasible trajectory is identified, the PGNC follows the last available trajectory while searching for a new one at each PGUI cycle.
The third case is when the solution found is sub-optimal. The solution found can be considered acceptable, but the HDR values associated with it can be very close to the limit ones. These are therefore feasible trajectories that are preferable to be used if strictly necessary also taking into account that the method employed to categorize the solution (based on HDR) is finite differences, which is rather inaccurate. The action performed by the PGUI, in this case, depends on the value taken by *SolLostCounter*. If the solution has not been updated for a number of cycles greater than or equal to the design value *SuboptimalTrgThrs*, then the HDG sequence is updated with the suboptimal solution found. Otherwise, the previous solution is kept, as in the case of unacceptable solution (increasing the value of *SolLostCounter* by one unit). The value of *SuboptimalTrgThrs* selected is 5 then a suboptimal solution is used only after 4 cycles of PGUI with an unacceptable/suboptimal solution (resetting the *SolLostCounter*).
In addition, there is a further case where the solution found is unacceptable for a number of PGUI cycles greater than or equal to the *BackupTrgThrs* threshold value (which is set to 30). In this case, the

backup solution consisting of Point to Target is activated. Considering the frequency of PGUI, this means that if the solution is lost for more than 12 seconds the backup is triggered. When the backup is active, the SolLostCounter is never reset, so at each PGUI cycle, the PGNC still attempts to recover an optimal solution. This assumption is not to be discarded, considering that, for instance, the strong unpredictable impact of wind on the trajectory may lead to a condition where an optimal solution is available. Please note that the output of the guidance, in this case, consists of a constant HDG and it is sent to the PID instead of the LQR. This strategy allows all possible situations to be taken into account providing a robust approach to the proposed guidance problem. The trajectory update every 6 seconds is the result of a trade-off that mainly takes into account the velocity of the system's dynamic response and the impact of the wind. Figure 6-8 illustrates how the sequence of optimal HDR varies markedly over time as a result of disturbances and uncertainties effects. Shown in light blue is the first calculated HDR sequence while in green is the HDR sequence updated according to the logic described above.

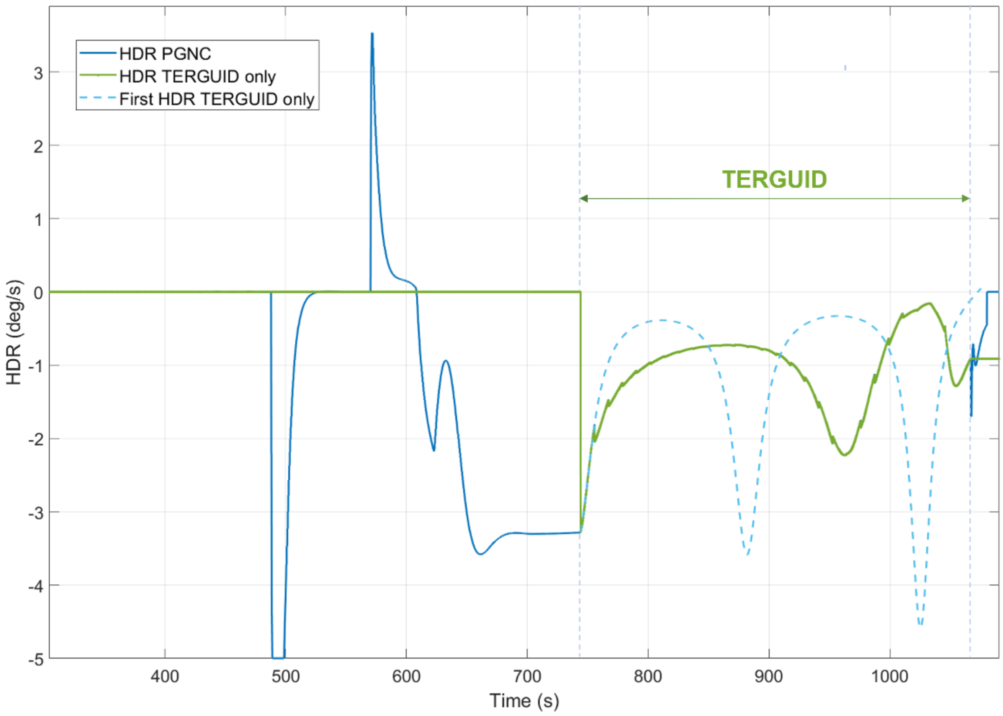

Figure 6-8: HDR sequence simulation

Figure 6-9 and Figure 6-10 show an example of activation of the backup solution. For this example, the trajectory is updated every 12 seconds (30 cycles of PGUI) and the value of *BackupTrgThrs* is 25. A standard first phase is observed in which a new optimal solution (FULLOPT) is calculated every 30 cycles of PGUI and the counter *PguiCycCounter* is then reset. Thus, idle cycles and active cycles alternate regularly. Around second 1057 of the simulation, however, it is noticed that an active cycle gives in output an unacceptable path (NULLOPT). At each subsequent PGUI step the computation of the optimal trajectory is then retried but without any success. The counter *SolLostCounter* thus increases its value until it reaches the threshold of 25 at which the backup solution is activated. The PGUI keeps searching for an optimal solution with a frequency of 2.5 Hz but does not find it and thus remains in backup mode until the end of the simulation.

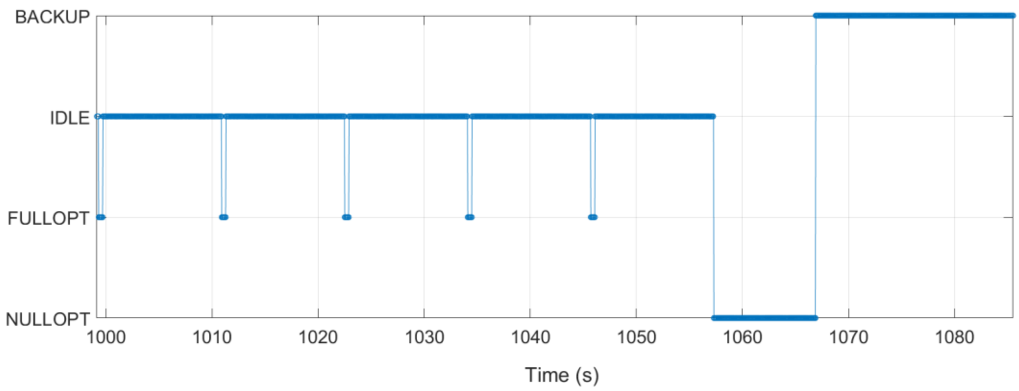

Figure 6-9: TERGUID flags

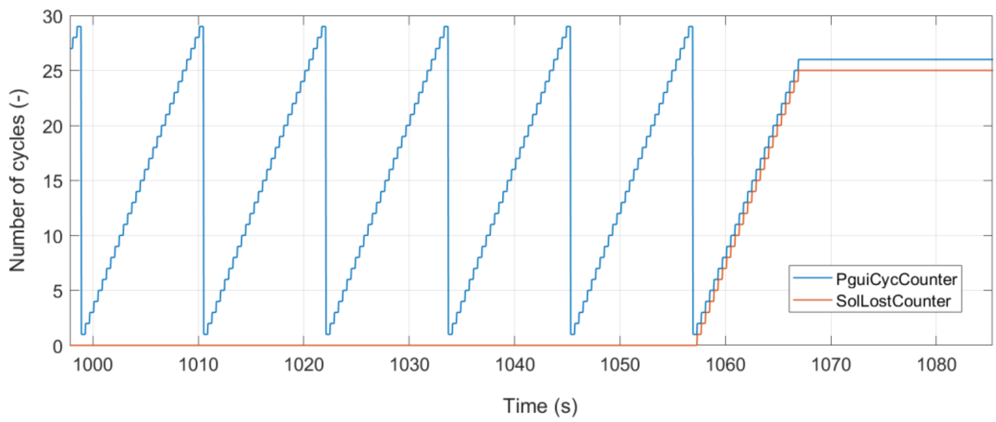

Figure 6-10: TERGUID counter

7. Simulations Results

This section presents the main results obtained using the 6dof simulator developed by SENER Aeroespacial. The simulator allows to perform single and Monte Carlo simulations selecting the parameters to be varied for each simulation. The main parameters that are considered for this study are as follows:

- Initial conditions in terms of position, velocity, and attitude
- Vehicle mass properties
- Sensor uncertainties and errors
- Uncertainties on the aerodynamic parameters of the parafoil

By changing the initial conditions, the boundary conditions of the problem are varied. Mass properties are not known during the design phase because they depend on the payload selected for the RM (TBD), which affects parameters such as the total mass of the system, position of the center of gravity, and moments of inertia. Therefore, for performance assessment, it is necessary to apply a certain dispersion to these parameters to consider different possible configurations. In addition, uncertainties associated with the accuracy of the navigation sensors and knowledge of the aerodynamic parameters of the parafoil must be taken into account. The latter play a key role in characterizing the dynamics of the system. They are generally provided as a result of flight tests and depend on the specific prototype used for testing. For this reason, some level of uncertainty in their knowledge should always be assumed. Moreover, the DKE of the simulator employs a different wind profile for each shoot of a Monte Carlo simulation in order to vary the wind effects. It is also possible to disable the presence of wind to compare the results obtained with and without wind. The overall parafoil mission performance does not depend solely on the TERGUID code implemented for this work, but on the general operation of the GNC software developed by SENER Aeroespacial and already implemented in the simulator. However, in this section, only the performance related to landing accuracy will be analyzed, which mostly depends on the TERGUID algorithms. It is emphasized that the performance figures shown here are not

representative of the ones achieved by the GNC software developed by SENER Aeroespacial. They rather aim to offer a general overview of the performance achievable with the strategy proposed in this study. The campaign of simulations carried out is not to be considered as a formal validation campaign of the implemented algorithms but only as a preliminary assessment of the TERGUID performance. However, the case history available for performance evaluation is sufficiently large to draw some basic conclusions about the operation of the approach used in this work. In fact, in addition to the simulations whose results will be illustrated in this chapter, there is a set of simulations carried out during the design phase. The latter was required for the gradual verification of the functions implemented in the code.
Regarding the mission requirements considered for the present study, there are two main elements to be taken into account. The first is the landing accuracy requirement of 150 m with 3σ standard deviation. It is important to point out that this requirement, in accordance with the observations made in [10] based on the results of flight tests with different types of parafoils (one among all X-38), is rather restrictive for large parafoils such as the one used by Space Rider. Compliance with this requirement depends mainly on the operation of the TERGUID submode. Another requirement to be met concerns the landing direction, which must be parallel and opposite to that of the wind on the ground so as to reduce the speed to touchdown (TD) as much as possible. This is equivalent to imposing horizontal ground speed at touchdown in the positive x-axis direction of the Landing Point Frame (see Section 3.3.1). The first requirement ensures that the RM lands at the intended landing site avoiding mission safety issues. The second requirement, on the other hand, ensures a soft touchdown that prevents any damage to the RM with a view to its reuse. To summarize, the requirements considered are:

- Landing accuracy of 150 m 3σ
- Groundspeed parallel and opposite to the wind direction at TD

7.1. Logic Operation

For performance verification, several post-processing tools were developed in order to analyze the outputs of the simulations. A first tool allows visualizing the trend of the optimal trajectory computed along the different points of the TERGUID path. This provides insight into how the trajectory calculation adapts to external disturbances and allows to analyze any cases where the optimal solution is lost. Two examples of the tool output obtained for single simulations in the presence of wind are shown in Figure 7-1 and Figure 7-2. The blue dots indicate the trajectory followed during the TERGUID phase while the solid lines in green are a sample of the optimal trajectories computed by the guidance algorithm along the path. On the other hand, the light blue solid line is the first computed optimal trajectory, which is the one that triggers the Terminal Guidance submode. In the total absence of disturbances and uncertainties, the vehicle should follow the light blue trajectory for the entire duration of the TERGUID phase. It is observed, however, that this is not the case. In fact, the guidance function works properly by calculating an optimal trajectory that gradually adapts to the varying problem conditions. Analyzing several cases such as the ones under consideration, one common element is noted in most of them. During the initial phase of TERGUID, the computed optimal trajectory varies markedly and then tends to a single solution in the final part of TERGUID. This trend is independent of the wind profile encountered and is mainly related to the fact that the wind drift has less and less effect as the altitude decreases (the drift is an integral in altitude). In the final stages of TERGUID, the calculated trajectory is progressively less affected by the wind (the main disturbance factor) and thus undergoes less variation in its shape. In both cases, the backup solution is not activated, and the TERGUID phase ends in favor of the Final Corrections phase since the conditions for exiting the TERGUID are met. Indeed, a final delta heading of less than 10 degrees is observed.

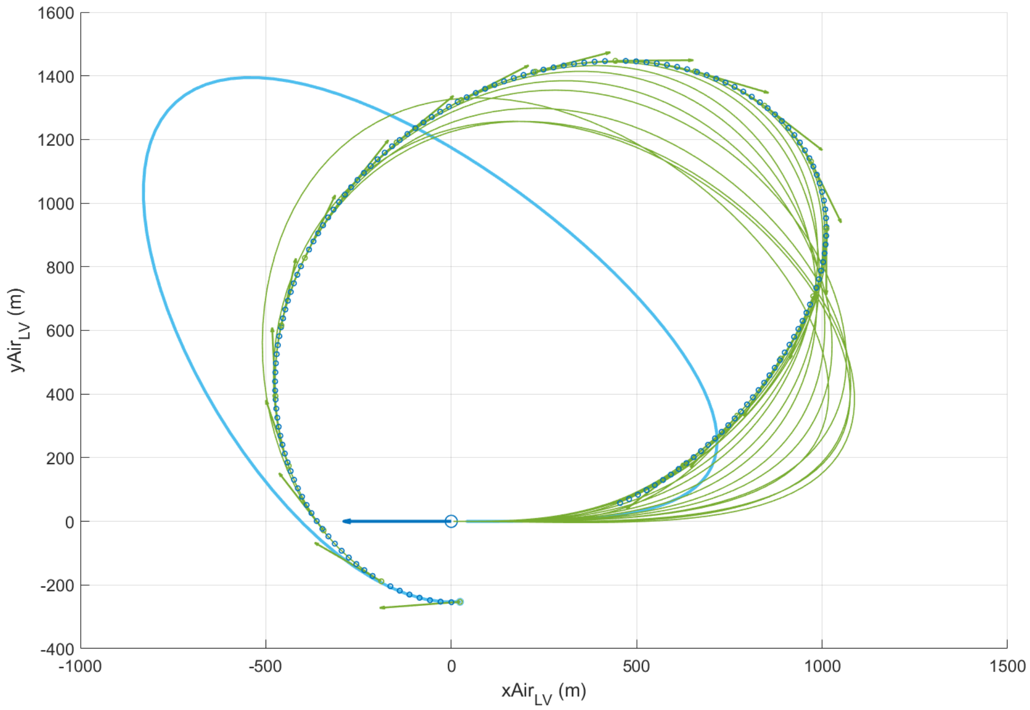

Figure 7-1: First example of optimal trajectory generation evolution

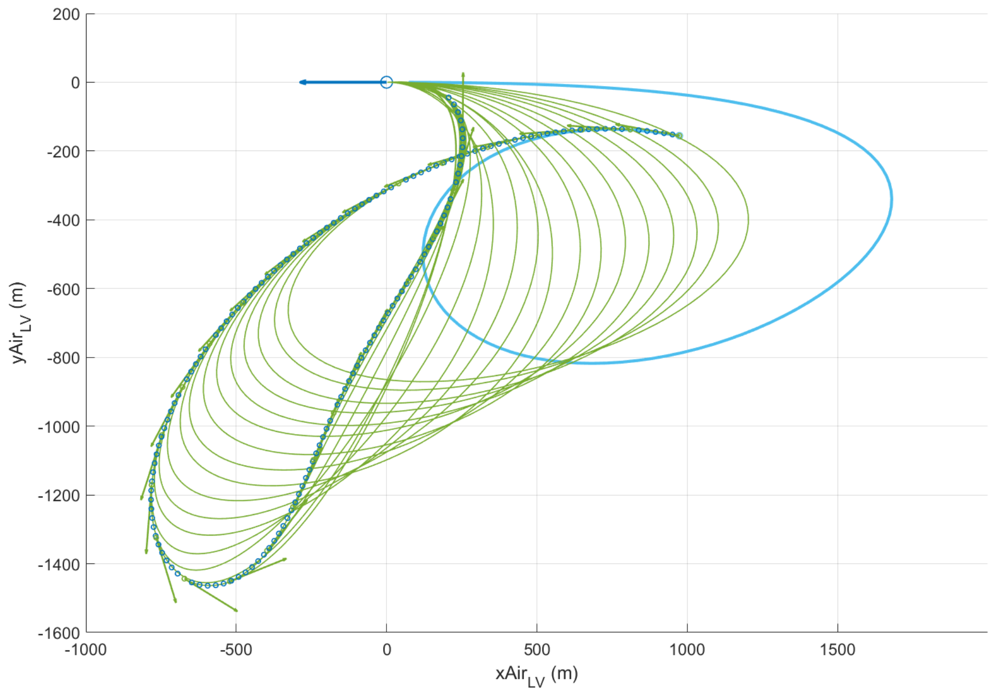

Figure 7-2: Second example of optimal trajectory generation evolution

It is also important to note that the variation of the optimal trajectory along the path of the TERGUID phase depends not only on the effects of disturbances and uncertainties but also on the fact that several assumptions were made in generating the trajectory. These assumptions provide a good approximation of the dynamic behavior of the system but are not exact. An example is the assumption of constant horizontal and vertical airspeed norms. Figure 7-3 shows the norm trends of the two airspeed components. It can be seen that the two components are not constant over time. This has a severe impact on the trajectory generation algorithm, which, for the optimization, relies on the time required to perform the entire maneuver in WF and thus on horizontal and vertical airspeed values. The figure also shows that the airspeed employed by the guidance does not exactly match the one encountered (DKE) because its value is not measured directly but estimated on board by the navigation.

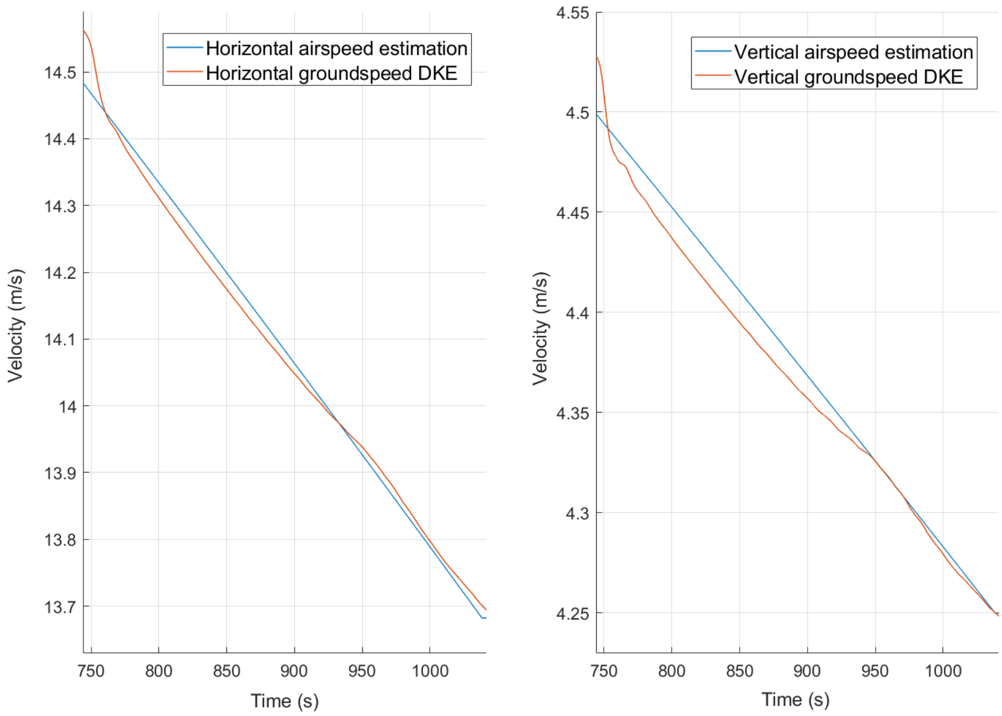

Figure 7-3: Horizontal and vertical airspeed trend

7.2. Landing accuracy

This section presents the performance of the implemented algorithms in terms of landing accuracy. As mentioned, it depends mainly on the TERGUID phase. However, once the TERGUID phase is completed there are additional small corrections for the residual longitudinal and lateral errors through the Final Corrections submode that precedes the Flare. This submode is not investigated for this work therefore, for the landing accuracy performance analysis, it was turned off. The trajectory is thus guided by the TERGUID submode until the instant the Flare is activated. It is important to keep in mind that performance in terms of longitudinal distance (along the x-axis of LPF) and velocity direction at touchdown, can be improved by reactivating that submode. Figure 7-4 shows the result obtained for a 100-case Monte Carlo simulation without wind using a PID controller for trajectory tracking (see section 5.1).

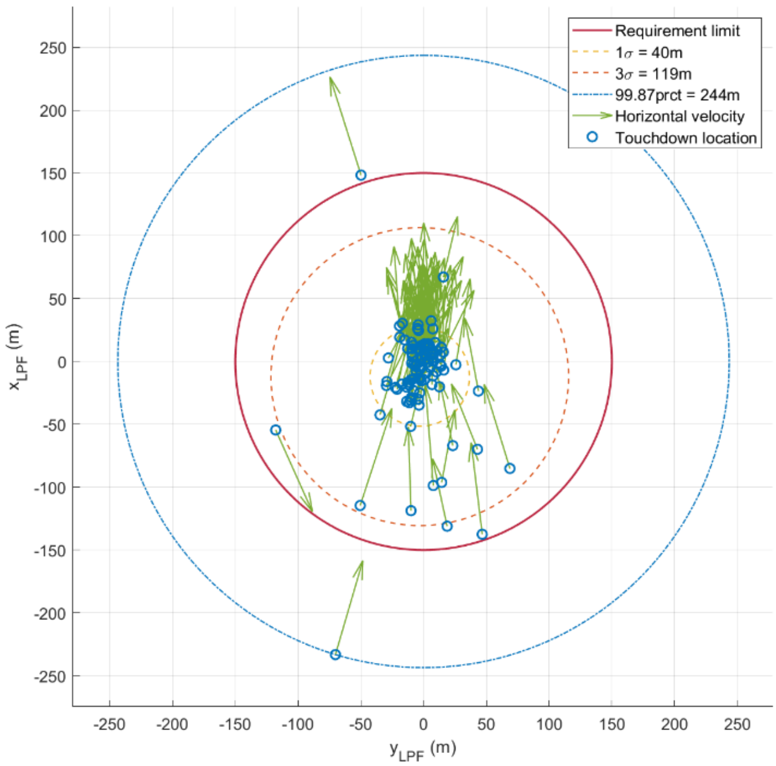

Figure 7-4: MC simulation for PID controller without wind (100 shots)

Touchdown points, in blue, are represented in Landing Point Frame (LPF). The horizontal velocity at touchdown, in green, is reported for an understanding of its orientation without specifics regarding its magnitude. The mission requirement for landing accuracy is shown in red and consists of a circumference with a radius of 150 m centered in the Landing Point (LP). The two dashed-line circumferences in yellow and orange indicate 1σ and 3σ landing accuracy, respectively (the latter being the one to be considered for mission requirements). These circumferences are not centered in the LP but at the average touchdown point. Then the 99.87 percentile is also reported, which, however, in the case of only 100 simulations, corresponds to the maximum value of the distance from the landing site. For the case with PID, there is some dispersion at landing even in the absence of wind, with a 3σ value of 119m. Some outliers do not meet the requirements in terms of landing accuracy or horizontal velocity direction at touchdown. For the validation of the algorithms, an in-depth study of such cases is required in order to identify the issues that arise for each of them. These issues, in general, may be outside the TERGUID's competence and depend on the previous phases of flight. Such analysis, however, is beyond the scope of this section, which is to illustrate the general performance of the implemented algorithms. Figure 7-5 shows a Monte Carlo simulation similar to that in Figure 1 but implementing LQR instead of PID. It is observed that the performance improves compared to the case with PID, with a 3σ value of 56m. The effect of turning off the Final Corrections submode appears evident here, resulting in a distribution of touchdown points along the x-axis due to the absence of a final longitudinal correction. The direction of horizontal velocity at touchdown appears to be well aligned with the desired direction. There is a group of simulations with a direction not exactly aligned with the x-axis. Initial analysis indicated no specific correlation with incorrect operation of the TERGUID algorithms. Moreover, the magnitude of the misalignment is such that it can be easily corrected by activating the Final Corrections submode.

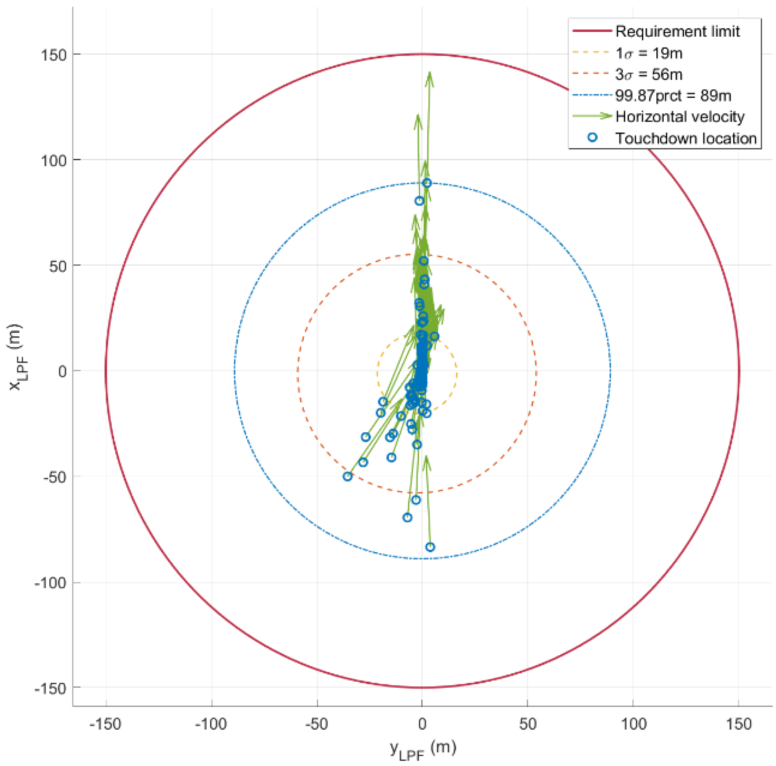

Figure 7-5: MC simulation for LQR without wind (100 shots)

The two simulations analyzed above are useful to compare the results obtained with PID and LQR highlighting the advantages obtainable by using LQR for trajectory tracking. However, these simulations do not take into account the central element for the design of TERGUID algorithms, which is wind. Figure 7-6 shows the results obtained with LQR in the presence of the wind. The wind profiles considered have the following characteristics:

- Maximum intensity: of 12 m/s
- Maximum knowledge error: 5 m/s

The knowledge error refers to the difference between encountered wind (DKE) and the wind predicted by the table on-board. This difference causes an error in the calculation of wind drift (see Sections 4.4.2) and thus in the generation of the optimal trajectory. Adding the wind effect, It is immediately evident that the landing points are significantly more dispersed. The 3σ value is 139 meters. However, it remains within the limits of mission requirements. The wind also has a noticeable effect on the direction of the horizontal velocity at touchdown, which, nevertheless, can be partly compensated for by activating the Final Corrections submode.

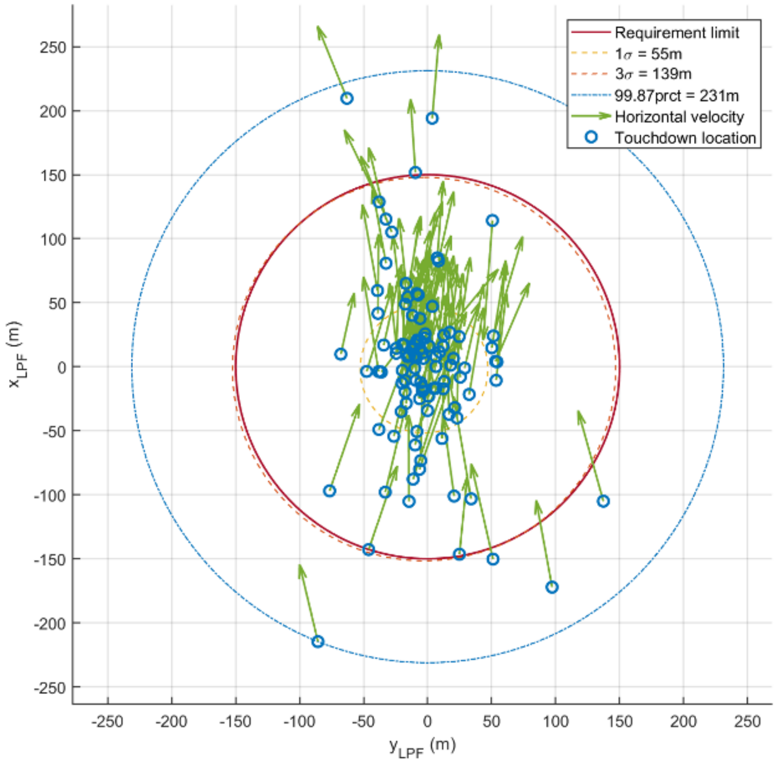

Figure 7-6: MC simulation for LQR with wind (100 shots)

The results shown above concern Monte Carlo simulations of only 100 shots that help to get an idea of the general performance but are not sufficient to provide reasonably reliable figures of merit. Therefore, a larger Monte Carlo simulation campaign was carried out considering 1000 shots for each of the cases seen above. The results are shown in Table 8. They are in line with those seen above and can be considered fully satisfactory for the purposes of this study.

Case study	Landing accuracy 1σ	Landing accuracy 3σ
PID without wind	43.1 m	132.4 m
LQR without wind	25.6 m	47.8 m
LQR with wind	62.3 m	146.0 m

Table 8: 1000-shots MC simulation results

8. Conclusions and Future Work

In this thesis, a possible approach has been proposed for managing the Terminal Guidance phase in the framework of the reentry mission of a payload-parafoil system such as the one constituted by Space Rider. A theoretical discussion of the proposed method has been presented first and then a possible practical implementation in order to be able to test the algorithms in a 6dof simulator. The challenges faced in the design and implementation of the method are many. Foremost among them was being able to adapt the method proposed by Yakimenko and Slegers [1] to the problem at hand. Indeed, early tests with the simulator demonstrated low compatibility of the original guidance algorithm with the Space Rider case bringing to light several issues that were addressed by the modifications proposed in Section 4.4. Among them, the low efficiency in finding a solution of the algorithm with Single-Parameter Optimization (SPO) stands out. This issue does not greatly affect the triggering of the TERGUID phase because the spiral trajectory of the ENEMNG always guarantees a combination of boundary conditions for which the SPO has an optimal solution. During the TERGUID phase, in contrast, as illustrated in Figure 7-2, the effects of disturbances and uncertainties can lead to conditions that deviate greatly from those predicted with the first computed trajectory (when the TERGUID is triggered). This can result in the SPO no longer finding an optimal solution. One approach that can be used is to appropriately employ both Single Parameter Optimization (SPO) and Two-Parameter Optimization (TPO). SPO is characterized by a significantly low computational cost that makes it preferable for equal performance in finding the solution. It is then possible to design the logic in such a way that at each active cycle of PGUI an SPO is used first, and only if the solution found is not optimal, the TPO is employed. TPO, at a higher computational cost, is capable of providing an optimal solution for a much wider range of problem boundary conditions. This type of strategy is often used when applying optimization techniques.

Some other improvements can be applied to the guidance algorithm. For example, it is possible to change the shape of the solution of the direct method by acting on expression (141). Indeed, it may be

interesting to investigate how the algorithm behaves replacing the polynomial or sinusoidal part with another type of expression by checking whether more effective solutions are obtained.
In addition, some modifications can be considered regarding the optimization process. The first interesting analysis can be regarding the second parameter chosen for Two-Parameter Optimization (TPO). In fact, the final HDR $\dot{\psi}_f$ was chosen following a series of comparative tests with several possible parameters. It is possible, however, that some unexplored solutions guarantee better results. For instance, it might be considered to use a scaling factor that multiplies a set of the terms in expression (141) and is used as the second parameter. Another aspect to be studied may be related to the fact that the function to be minimized turns out to be nonconvex, which generally makes the search for the optimal solution slower. This problem is partly compensated for by the fact that the initial point adopted for the optimization (expression (182) and $\dot{\psi}_f = 0$) is generally quite close to the optimal one. In any case, to further accelerate the search for the optimal solution, especially in the case of Two-Parameter Optimization, it is worth trying to make the problem convex to facilitate the search for the minimum point of the cost function.
For trajectory tracking instead, a possible alternative option is to implement the Model Predictive Control seen in Section 5.2 instead of a Linear Quadratic Regulator. MPC, despite a higher computational cost, provides a more robust approach, especially in relation to overshooting. To determine which approach is more cost-effective, a comparative campaign can be performed to highlight performance differences in terms of landing accuracy and computational cost required.
Another major design challenge was the development of the logic for managing the TERGUID phase. This task requires a deep understanding of the complete GNC code and the various interfaces between GNC functions. In the initial theoretical design of the logic, it is indeed necessary to already have in mind its practical implementation within the code. It is then important to be aware of all possible issues that may arise related to both the mission profile and the code itself to build an effective and robust approach. To this end, however, validation of the code through an extensive testing campaign is undoubtedly

required. For the present study, a series of Monte Carlo simulations were performed to test the performance of the various algorithms. For validation, it is necessary to design a larger campaign to verify whether the mission requirements are always fulfilled. For the present study, a series of Monte Carlo simulations were performed to test the performance of the various algorithms. For validation, it is necessary to design a larger campaign so that it can be verified whether the mission requirements are always fulfilled. A larger case study implies a higher probability of detecting special cases with nonstandard behavior, which must then be analyzed in detail to identify possible failures in the strategy used.

Despite the various possible improvements proposed above the results obtained with the 6dof simulator (shown in Chapter 7) can be considered fully satisfactory. The approach used proved to be sufficiently robust in handling the unpredictable effects of disturbances and uncertainties that constitute the main challenge for the proposed guidance problem. The mission requirements in terms of landing accuracy can be considered met despite the presence of some outliers that need further analysis.

References

[1] N. Slegers and O. Yakimenko, "Optimal Control for Terminal Guidance of Autonomous Parafoils," in *20th AIAA Aerodynamic Decelerator Systems Technology Conference and Seminar*, Seattle, Washington, 2009.

[2] B. Etkin, Dynamics of Atmospheric Flight, New York: New York, 1972.

[3] ESA, "Space Rider overview," [Online]. Available: https://www.esa.int/Enabling_Support/Space_Transportation/Space_Rider_overview.

[4] ESA, "Space Rider Enabling & Support," [Online]. Available: https://www.esa.int/ESA_Multimedia/Images/2019/05/Space_Rider#.YrnRyIKYOhc.link.

[5] J. Huart, "ESA," [Online]. Available: https://www.esa.int/ESA_Multimedia/Images/2020/11/Space_Rider3#.YrCrzDE6JLw.link.

[6] S. Gallucci, R. Mancini, E. Scardecchia and F. Spaziani, "The AVUM Orbital Module for the Space Rider System," in *8th European Conference for Aeronautics and Space Sciences (EUCASS)*, Madrid, 2019.

[7] Thalesgroup, "Thales Alenia Space and Avio Sign with European Space Agency the Space Rider Development Contract," [Online]. Available: https://www.thalesgroup.com/en/worldwide/space/press-release/thales-alenia-space-and-avio-sign-european-space-agency-space-rider.

[8] F. Cacciatore, R. Haya Ramos, L. Tarabini Castellani and A. Figueroa, "The Design of the GNC of the Re-entry Module of Space," in *8th European Conference for Aeronautics and Space Sciences (eucass)*, Madrid, Spain, 2019.

[9] I. Kaminer and O. Yakimenko, "Development of Control Algorithm for the Autonomous Gliding Delivery System," in *17th AIAA Aerodynamic Decelerator Systems Conference*, Monterey, CA, 2003.

[10] O. Yakimenko, Precision Aerial Delivery Systems: Modeling, Dynamics, and Control, Reston, VA: AIAA American Institute of Aeronautics and Astronautics, 2015.

[11] P. Lissaman and G. Brown, "Apparent mass effects on parafoil dynamics," *AIAA-93-1236,* 1993.

[12] G. Kowaleczko, "Apparent masses and inertia moments of the parafoil," *Journal of Theoretical and Applied Mechanics 52,* 2014.

[13] T. Jann, "Aerodynamic Model Identification and GNC Design for the Parafoil-Load System Alex," in *AIAA-2001-2015*, 2001.

[14] N. Slegers, "Effects of Canopy-Payload Relative Motion on Control of Autonomous Parafoils," *Journal of Guidance, Control, and Dynamics,* vol. 33, no. 1, p. 116–125, 2010.

[15] E. Mooij, Q. Wijnands and B. Schat, "9 dof Parafoil/Payload Simulator Development and," in *AIAA-03-5459 Simulation and Modelling Technologies Conference*, Austin, Texas, 2003.

[16] N. Slegers and M. Costello, "Model Predictive Control of a Parafoil and Payload System," *Journal of Guidance, Control, and Dynamics,* vol. 28, no. 4, p. 816–821, 2005.

[17] L. Fowler and J. Rogers, "Bézier Curve Path Planning for Parafoil Terminal Guidance," *Journal of Aerospace Information Systems,* vol. 11, no. 5, 2014.

[18] B. Rademacher, P. Lu, A. Strahan and C. Cerimele, "In-Flight Trajectory Planning and Guidance for Autonomous Parafoil," *Journal of Guidance, Control, And Dynamics,* vol. 32, no. 6, 2009.

[19] D. Carter , L. Singh, L. Wholey, L. Rasmussen and Et Al., "Band-Limited Guidance and Control of Large," in *20th Aerodynamic Decelerator Systems Technology Conference and Seminar*, Seattle, Washington, 2009.

[20] A. Calise and D. Preston, "Swarming/Flocking and Collision Avoidance for Mass Airdrop of Autonomous Guided Parafoils," *Journal of Guidance, Control, and Dynamics,* vol. 31, no. 4, pp. 1123-1132, 2008.

[21] ESA, "Space Rider," [Online]. Available: https://www.esa.int/Enabling_Support/Space_Transportation/Space_Rider.

[22] L. Fowler and J. Rogers, "Bézier Curve Path Planning for Parafoil Terminal Guidance," in *AIAA Aerodynamic Decelerator Systems (ADS) Conference AIAA 2013-1325*, Daytona Beach, Florida, 2013.