

POLITECNICO DI TORINO
MASTER's Degree in AEROSPACE ENGINEERING



**Politecnico
di Torino**

MASTER's Degree Thesis

**SOFTWARE TOOL DEVELOPMENT
FOR TRAJECTORY OPTIMIZATION
OF VERTICAL LANDING SPACE
VEHICLES**

Supervisors

Prof. LORENZO CASALINO

Ing. MARTINS SUDARS

Ing. ANDREA MUSACCHIO

Candidate

VALERIO MARTONI

OCTOBER 2025

A Mamma e Papà.
A Francesco.

Acknowledgements

This thesis is the result of work carried out during my internship at Thales Alenia Space Italia in Turin. I would like to thank those who gave me this opportunity, allowing me to enjoy an educational experience enriched by the people with whom I shared it.

Abstract

Since the birth of the space industry, the topic of re-usability has been taken into consideration without ever being able to be truly systematically applied to the development of fully reusable spacecrafts. Maintaining an efficient mass ratio for reusable vehicles is a very hard engineering challenge. In fact, launching relative small payload into orbit requires a big quantity of mass propellant, making it inefficient and expensive: re-usability requirements worsen it by adding further subsystems inert mass. However, today, the space industry is witnessing a renewed interest in the development of fully reusable space systems thanks to the economic advantages and the increase in safety and environmental sustainability. In this context, this thesis aims to contribute to the technological research in this field, investigating the current state of the art of reusable spacecrafts. The subject matter covering reusable vehicles, from their design to their use, is quite broad: this thesis focuses on vertical landing first stage launchers, that fall into the category of low mechanical energy re-entry spacecrafts. The re-entry problem into the atmosphere of a non-reusable spacecraft is already well known and has been widely addressed over the decades and this knowledge can also be applied to the re-entry of reusable spacecraft. On the contrary, the final descent into the lower atmosphere and the vertical landing differ from typical recovery strategies adopted for non-reusable vehicles. In fact, this phase is critical because it requires active, continuous and precise control of the vehicle to achieve specific position and velocity conditions to complete the landing. These trajectory requirements translate directly into performance requirements for the vehicle, which must be equipped with appropriate control systems such as fins, thrust throttling and thrust vector control. For this reasons this thesis work developed a **MATLAB** software tool able to compute offline trajectory optimization for final descent and vertical landing. This tool, given specific set of input design parameters of the vehicle, can optimize trajectories respecting the bounds and the constraints imposed by the mission scenario. The software tool is conceived for preliminary mission analysis applications. By studying the trajectory optimization results, vehicle performance requirements can be analyzed to validate the set of input design parameters allowing to preliminary size a reusable first stage launcher and reach the main objective of this thesis work.

Table of Contents

List of Tables	III
List of Figures	IV
Acronyms	VII
1 The Research Problem	1
1.1 Introduction: The New Space Industry of Reusable Spacecrafts . . .	1
1.2 Mission Context: Recovery and Landing Strategy for a First Stage Vehicle	3
1.3 Objective: Vertical Landing Trajectory Optimization	6
2 Engineering Modeling for Vertical Landing Vehicle	7
2.1 Vehicle Geometry Model	9
2.1.1 Aerodynamics control surfaces: the planar fins	11
2.1.2 Thrust Vector Control system	12
2.2 Vehicle Aerodynamics Model	12
2.3 Equations of Motion	18
3 Trajectory Optimization	24
3.1 The Optimal Control Problem	24
3.2 The Direct Collocation Method	27
3.3 Trajectory Optimization for Vertical Landing First Stage Vehicle . .	33
4 Software Tool Development	39
4.1 Aerodynamics Submodule	40
4.2 Dynamics Submodule	41
4.3 Optimization Submodule	42

5	Software Tool Testing	52
5.1	Case study	52
5.2	Results	57
5.2.1	Solution A	60
5.2.2	Solution B	68
5.2.3	Error analysis	76
6	Conclusion	78
	Bibliography	79

List of Tables

2.1	High-level design variables	9
3.1	Setup for aerodynamic descent trajectory optimization.	36
3.2	Setup for powered landing trajectory optimization.	38
5.1	Design parameters input set for software tool testing	54
5.2	Landing target coordinates in ECEF	54
5.3	Path constraints and constant bounds definition for control systems	57
5.4	System states constant bounds	58
5.5	Solution A	58
5.6	Solution B	59

List of Figures

1.1	Falcon 9 and Starship by Space X, New Shepard by Blue Origin . .	1
1.2	Forecast of Launch Costs by 2040 (\$ per kg). See [1]	2
1.3	Reentry corridor profile examples for high and low energy vehicles. See [2]	3
1.4	Recovery concepts. See [2]	5
2.1	Disciplines correlations in launch vehicle design. See [4]	8
2.2	Design process logic for vertical landing vehicle. See [5]	8
2.3	Geometry model (non-propotional) for the vehicle first stage.	10
2.4	3D model of Falcon 9-class vehicle. See [6]	10
2.5	Elements of Falcon 9-class vehicles. See [6]	11
2.6	Planar fin model. See [9]	12
2.7	Different TVCs strategies. See [10]	13
2.8	Static stability of a descending VLV.	14
2.9	System representation for aerodynamics analysis.	16
2.10	System representation for translational equations of motion.	20
2.11	System representation for rotational equations of motion.	20
3.1	Example of local and global optima of a non-linear function. See [17]	26
3.2	OCPs resolution strategies. See [18]	27
3.3	Example of function approximation using linear spline. See [21] . .	29
3.4	Collocation nodes evaluation process. See [22]	30
3.5	Quadratic spline interpolation for system control and dynamics when using Hermite-Simpson DCM. See [23]	31
3.6	Example of error for system dynamics	33
3.7	Flight phases of VLV. See [25]	34
4.1	SNOPT Call Interface.	43
5.1	Sketches of investigated reusable first stages. See [30]	53
5.2	Cross-range in AD phase	60
5.3	Altitude in AD phase	60

5.4	Horizontal velocity in AD phase	60
5.5	Vertical velocity in AD phase	60
5.6	Pitch angle in AD phase	61
5.7	Pitch rate in AD phase	61
5.8	Angle of attack in AD phase	61
5.9	Flight path angle in AD phase	61
5.10	Single fin deflection for aerodynamic control	62
5.11	Cross-range in PL phase	62
5.12	Altitude in PL phase	62
5.13	Horizontal velocity in PL phase	63
5.14	Vertical velocity in PL phase	63
5.15	Pitch angle in PL phase	63
5.16	Pitch rate in PL phase	63
5.17	Angle of attack in PL phase	63
5.18	Flight path angle in PL phase	63
5.19	Thrust magnitude as % of total thrust	64
5.20	Thrust deflection angle by TVC	64
5.21	Velocity time profile for solution A	64
5.22	Pitch angle and pitch rate time profile for solution A	65
5.23	AoA and FPA time profile for solution A	66
5.24	Total vehicle mass profile for solution A	67
5.25	Trajectory position visualization for solution A	67
5.26	Cross-range in AD phase	68
5.27	Altitude in AD phase	68
5.28	Horizontal velocity in AD phase	68
5.29	Vertical velocity in AD phase	68
5.30	Pitch angle in AD phase	69
5.31	Pitch rate in AD phase	69
5.32	Angle of attack in AD phase	69
5.33	Flight path angle in AD phase	69
5.34	Single fin deflection for aerodynamic control	70
5.35	Cross-range in PL phase	70
5.36	Altitude in PL phase	70
5.37	Horizontal velocity in PL phase	71
5.38	Vertical velocity in PL phase	71
5.39	Pitch angle in PL phase	71
5.40	Pitch rate in PL phase	71
5.41	Angle of attack in PL phase	71
5.42	Flight path angle in PL phase	71
5.43	Thrust magnitude as % of total thrust	72
5.44	Thrust deflection angle by TVC	72

5.45	Velocity time profile for solution B	72
5.46	Pitch angle and pitch rate time profile for solution B	73
5.47	AoA and FPA time profile for solution B	74
5.48	Total vehicle mass profile for solution B	75
5.49	Trajectory position visualization for solution B	75
5.50	Error for horizontal velocity	76
5.51	Error for horizontal acceleration	76
5.52	Error for pitch angular rate	77
5.53	Error for pitch acceleration rate	77

Acronyms

AD

Aerodynamic Descent

AoA

Angle of Attack

BRF

Body Reference Frame

CoG

Centre of Gravity

CP

Center of Pressure

DoF

Deegres of Freedom

DCM

Direct Collocation Method

DR

Down Range

ENU

East North Up

HL

Horizontal Landing

NLP

Non Linear Program

OCP

Optimal Control Problem

PL

Powered Landing

RTLS

Return To Landing Site

TVC

Thrust Vector Control

VL

Vertical Landing

VLV

Vertical Landing Vehicle

WRF

Wind Reference Frame

Chapter 1

The Research Problem

1.1 Introduction: The New Space Industry of Reusable Spacecrafts

Since the birth of the space industry, the topic of re-usability has been taken into consideration without ever being able to be truly systematically applied to the development of fully reusable spacecrafts. The recovery of the re-entry but not reusable Apollo capsule and the missions of Space Shuttle, designed as a partially reusable system, marked the main stages of the reusable space vehicles history. After that, technological research for reusable spacecrafts was mainly abandoned, for reasons primarily due to technological limitations.



Figure 1.1: Falcon 9 and Starship by Space X, New Shepard by Blue Origin

The main challenge in spacecraft design has always been to ensure an efficient payload ratio, which is to say, maximizing the mass of the payload carried relative to the total remaining mass of all other subsystems and propellant. In fact, launching relative small payload into orbit requires a big quantity of mass propellant, making it inefficient and expensive: reusability requirements worsen it by adding further

subsystems inert mass and in some cases ulterior propellant mass.

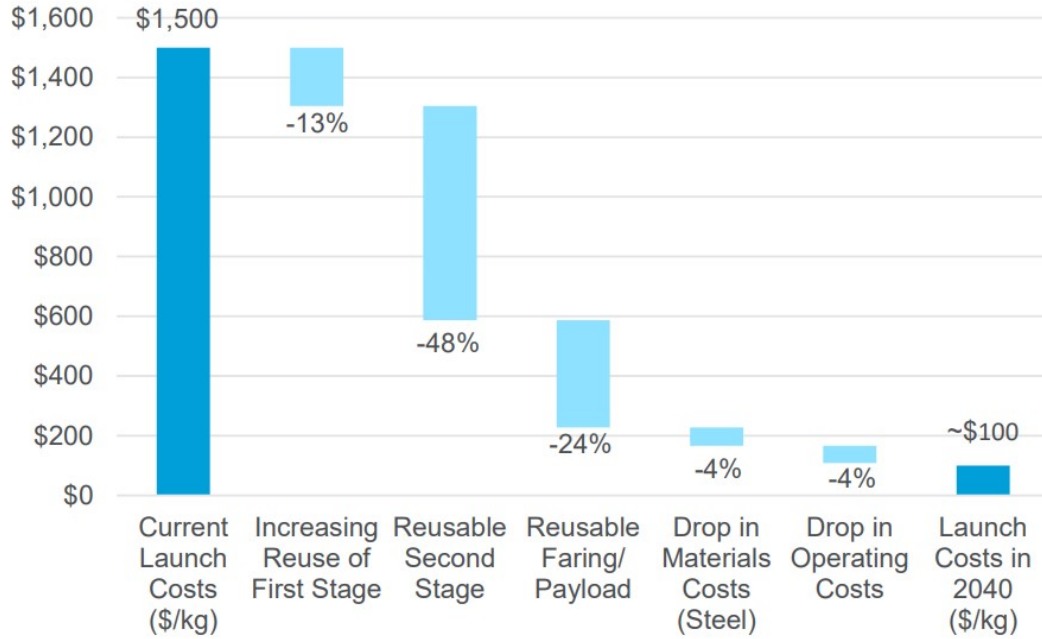


Figure 1.2: Forecast of Launch Costs by 2040 (\$ per kg). See [1]

However, today, the space industry is witnessing a renewed interest in the development of fully reusable space systems thanks to the technological advancements made in space engineering. In particular, innovations in material science, propulsion systems and guidance and control techniques have already fully demonstrated the possibility of building partially but systematically reusable rockets, aiming at fully reusable spacecrafts in the near future. The technical advancements in reusable rockets include vertical landing systems and grid fins that enhance precision during descent allowing rockets to land safely on designated platforms or ships, ensuring they can be reused multiple times. These technologies have improved reliability and efficiency, making space missions more sustainable. Thus, the main reasons for this re-discovered trend is based on two main considerations [2]: reusability can provide an economic advantage lowering the launch cost up to 30 % [3], with a continuously projected decline as shown in Figure 1.2, depending on the launch rate, the recovery and refurbishment cost and the extension of reusability to the entire spacecraft; reuse of stages can avoid fallout in sea or lands, with more or less important consequences depending on parameters like the population density in fallout zones and the potential pollution of stages components in these areas. Among the pioneers of reusable launch vehicles, the private companies Space X and New Glenn stand out, which, in addition to marking a decisive advancement

in the space industry, have also revolutionized the space economy. In fact, the affordability due to cost reduction is democratizing space exploration, allowing private smaller companies and research institutions to participate in LEO missions that were previously cost-prohibitive and, furthermore, it has been possible to consider new frontiers for human research and exploration in space, toward the Moon and beyond.

1.2 Mission Context: Recovery and Landing Strategy for a First Stage Vehicle

Reusable vehicles cover a wide range of re-entry mission scenarios mainly according to the energy associated to them when re-entry mission starts, i.e. the corresponding sum of kinetics and potential energies. Two main categories can be identified.

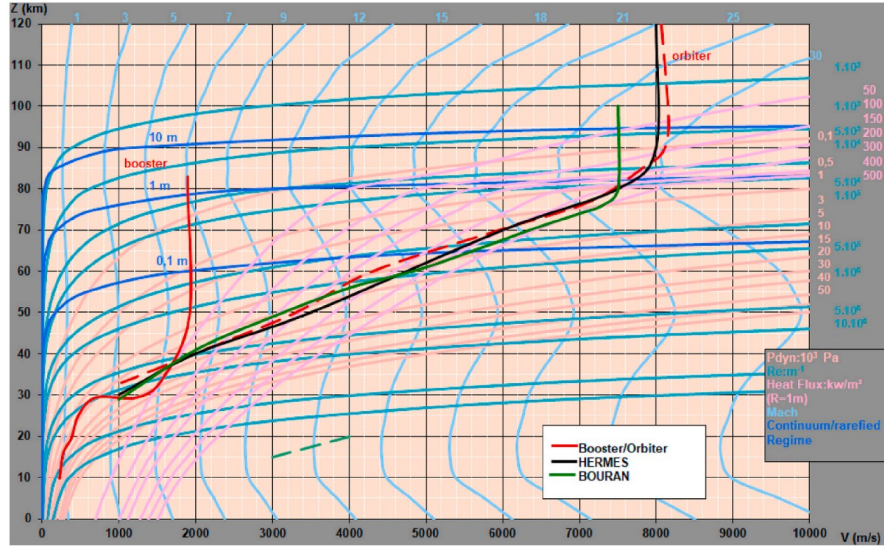


Figure 1.3: Reentry corridor profile examples for high and low energy vehicles. See [2]

The first one refers to the high-energy vehicles which are the orbital vehicles such as launcher second stages or capsules. Instead, the second one refers to the low-energy vehicles which are the suborbital vehicles such as launcher first stages. Reusable vehicles of both categories have to be designed to face the re-entry mission which involves entering in the Earth atmosphere and slowing down the vehicle using the brake action provided by drag atmosphere in order to reduce their mechanical energy before reaching the ground in safety conditions and be recovered or, in the case of reusable vehicles, perform a soft landing. Re-entry

missions are highly demanding in terms of system resilience across various aspects, with thermo-structural integrity being the foremost. Figure 1.3 shows an example of re-entry corridor for three different space vehicles in the altitude-velocity plane. The re-entry corridor is the narrow, three-dimensional flight domain defined by the minimum and maximum values allowed for specific parameters that allow a safety recovery of the vehicle. The critical area in the reentry corridor is identified in relation to the maximum heat flux and dynamic pressure. As can be expected, a low-energy vehicle re-entry corridor is subjected to less stringent conditions because of the lower Mach.

Instead, what particularly characterizes the specific case of the return of a launch vehicle's first stage is the multitude of scenarios and recovery strategies that have been hypothesized to achieve a soft landing that guarantees the reuse of the vehicle. In fact, by definition, the same reusable vehicle is used several times in different launch missions, so each mission is a specific case in which MECO (Main Engine Cut-Off), that marks the start of the re-entry mission for the first stage, can occur under different Mach and altitude conditions. This is directly linked to the first distinction in recovery strategies between **RTLS** (Return To Landing Site) strategy and **DR** (Down Range) strategy. The second distinction is linked to the geometry of the vehicle and how it is controlled in the atmosphere, and consists of winged vehicles or no-winged vehicles. Finally, there is the distinction between **VL** (Vertical Landing) and **HL** (Horizontal Landing). Figure 1.4 shows different re-entry and recovery trajectories primarily designed for the first stage of a launch vehicle but applicable to any type of vehicle, even high-energy orbital vehicles, after they have been appropriately re-entered into the atmosphere. Green and purple lines are the ascending trajectories, the blue and orange ones are the descending trajectories and the red segments identify the trajectory segments where engine are re-ignited.

- **Concepts A, C.** These are both RTLS concepts. in concept A a non-winged vehicle performs VL while in concept C a winged vehicle performs HL. The main difference is that the dashed ascending trajectory for concept A must be verticalized in order to not increase the cross range distance, while concept C allow a certain cross range distance which is canceled thanks to a final gliding descent. Both concepts start the re-entry mission with a boostback burn: in concept A, the main objective of the boostback burn is to change the direction of the velocity and insert the vehicle on a ballistic trajectory to correctly target the desired landing site, while in concept C it is performed to cancel the horizontal trajectory in order to have vertical free fall of the vehicle to the lower atmosphere. Concept A requires a second burn before entering the atmosphere, called re-entry burn, to decrease the velocity of the vehicle making use of the propulsion system and thus maintain the aero-thermo-mechanical loads under control during the following aerodynamic phase. The objective

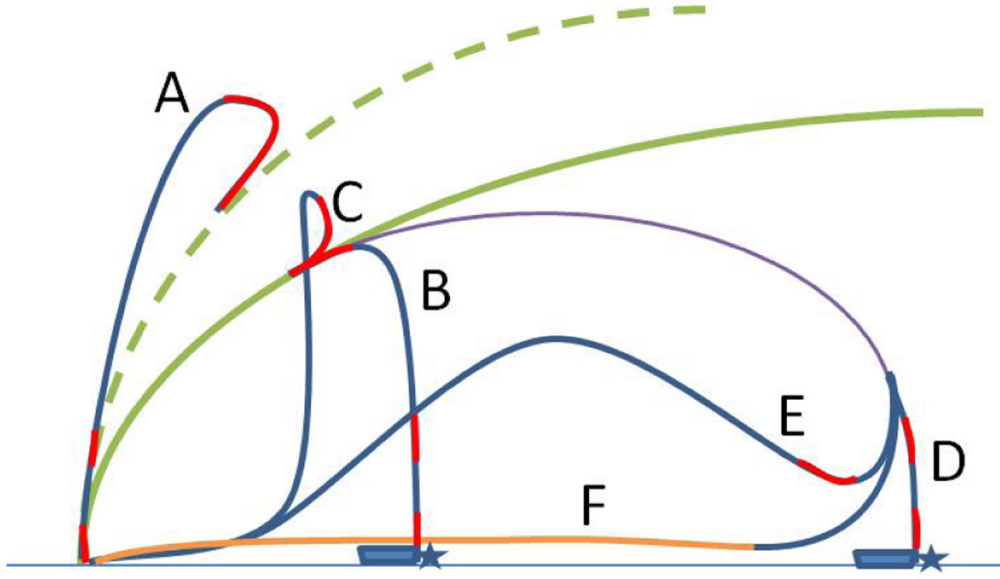


Figure 1.4: Recovery concepts. See [2]

of the aerodynamic entry phase is to successfully slow the vehicle down from hypersonic to subsonic Mach. Finally, a third burn, the landing burn, is performed during landing phase to successfully land the vehicle reaching zero velocity at touchdown.

- **Concepts B, D** These are both DR concepts similar to concept A but with low propellant consumed to perform re-entry. In fact, concept B still requires three burns but the first boostback is less expensive because it is not necessary to invert velocity direction. Concept D requires only two burns because the velocity at MECO conditions is directly used to insert the vehicle into an un-constrained ballistic trajectory.
- **Concepts E,F** These are both RTLS concepts for winged vehicles performing HL. After an un-constrained ballistic trajectory is performed as in concept D, the vehicle uses wings to perform an aerodynamic U-turn. In concept E a boostback burn is performed to target the landing site and then a gliding phase completes the reentry trajectory in atmosphere. Concept F does not require boostback burn but after an aerodynamic U-turn to face the landing site, the trajectory is completed by cruising up to the landing site using secondary propulsion.

1.3 Objective: Vertical Landing Trajectory Optimization

In the context of renewed interest in the design and development of spacecraft presented above, and in light of the new technological and competitive challenges that this entails, this thesis aims to address the subject of reusable spacecraft in order to contribute to technological research in this field. As already described, the subject of reusable vehicles is very broad, both in terms of vehicle type and in terms of the design of the vehicle itself, which must be capable of handling a reentry and landing mission. Considering that this thesis represents an ex-novo approach to this topic, it had to be scaled down to cover a specific class of vehicles among those presented, namely the vertical landing first stage of a reusable launcher. This choice was dictated by the current state of the art, where the only example of a systematically reusable vehicle in use is Space X's Falcon 9 booster. Therefore, this type of vehicle is the subject of more extensive literature and more frequent research, facilitating the retrieval of information useful for the development of this work. The re-entry problem into the atmosphere of a non-reusable spacecraft is already well known and has been widely addressed over the decades and this knowledge can also be applied to manage the less demanding re-entry of reusable spacecraft. On the contrary, the final subsonic phase of the re-entry mission of a reusable vehicle comprehends the final descent into the lower atmosphere and the soft landing that differ from typical recovery strategy of non re-usable vehicles: this phase is critical because it requires active, continuous and precise control of the vehicle to achieve specific position and speed conditions to complete the soft landing. These trajectory requirements translate directly into performance requirements for the vehicle, which must be equipped with appropriate control systems such as fins, thrust throttling and thrust vector control. For this reasons the main objective of this thesis is developing a MATLAB software tool able to compute offline trajectory optimization for final subsonic descent and vertical landing. This tool, given specific high-level input design parameter of the vehicle, can optimize trajectories respecting the bounds and the constraints imposed by the mission scenario allowing to analyze the control performances required and to validate the input design parameters. The software tool is conceived as a design tool for mission analysis to compute nominal reference trajectory.

Chapter 2

Engineering Modeling for Vertical Landing Vehicle

This chapter describes the assumptions, choices, and methods used to model the first stage of a vertical landing space launcher and its position and attitude dynamics. It should be noted that the engineering modeling of the vehicle system was not the main objective of this thesis but was conducted with the primary aim of obtaining a sufficiently accurate preliminary model, to be used for testing the trajectory optimization software tool in order to address mission analysis and preliminary sizing of a VLV, with reference to the final aerodynamic descent and powered landing scenario as explained in chapter 1.

Therefore, this part of the thesis work was conducted by making assumptions, presented below, which simplified an activity that would otherwise have been very time-consuming and resource-intensive. The design of a complex system such as a VLV requires an iterative design process in which mission requirements influence system requirements and vice versa, resulting in complex multidisciplinary work as shown in 2.1. In fact, to approach the modeling of the vehicle system to be used in trajectory optimization, reference was made to the basic methodology used to develop a generic space system. The approach used in these cases is to build an iterative process. Figure 2.2 shows a possible logical flow of the iterative process. The set of design variables constitute the starting point as well as the final objective of the design process, as their values will only be fixed once the iterations have been interrupted upon satisfaction of the mission and system requirements. The various vehicle subsystems are modeled according to the design variables, attempting to follow a “cascade” logic in which the correlations and interdependencies between the various subsystems are exploited to use the model of a more "external" subsystem in the modeling of a more internal "subsystem". It is important to note the red

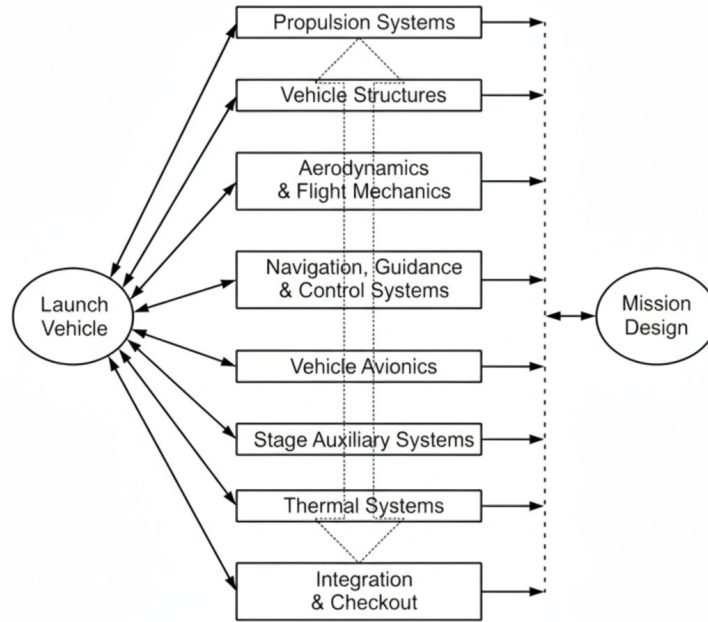


Figure 2.1: Disciplines correlations in launch vehicle design. See [4]

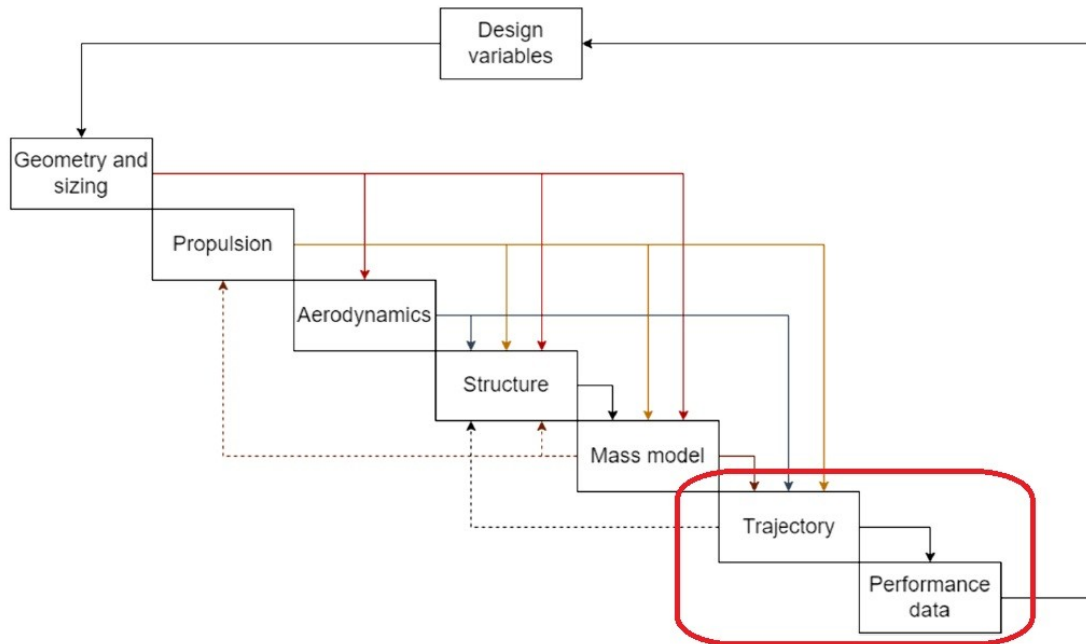


Figure 2.2: Design process logic for vertical landing vehicle. See [5]

box, which is the design phase in which the software tool is applied to perform the mission analysis optimizing the nominal trajectory in order to study the vehicle's performance and then verify the entire iteration, making the necessary changes to the set of design variables to start a new iteration aiming at obtaining convergence. As regards this thesis, this design logic has been simplified by overriding some systems modeling and directly using high-level design variables as input variables. Thus, the software tool at its initial development state can be tested against an input set of design parameters that can be easily estimated or found by consulting the literature, shown in table 2.1; alternatively, if future developments of software tool will be used for complete iterative design processes, more precise and detailed models for evaluating design parameters can be integrated upstream of the software tool in an iterative process.

Table 2.1: High-level design variables

High-level design variables legend with ref. to Figure 2.3	
M_0	First stage dry mass + fuel mass estimated for powered landing
L	First stage length
d	First stage diameter
c_{fin}	Fin aerodynamic mean chord
b_{fin}	Fin span
$X_{LE_{fin}}$	Fin leading edge longitudinal position w.r.t the base
CoG	Centre of gravity longitudinal position w.r.t. the base
A_e	Exit nozzle area
I_{sp}	Single engine specific impulse in vacuum
T_{sl}	Single engine thrust at sea level

2.1 Vehicle Geometry Model

The geometric model for the first stage of a launch vehicle was created using SpaceX's Falcon 9 vehicle as a reference, as it is currently the only space launch vehicle regularly used to launch payloads into orbit and return to Earth with a powered vertical landing. Consequently, it was possible to find information and data in the literature with which to estimate and model the vehicle. From the material found in the literature, Figure 2.4 shows a 3D model of a Falcon 9-type vehicle from which the simplified 2D model adopted in this thesis problem was derived, shown in Figure 2.3. The planar 2D model is used to study the DoFs-reduced system dynamics described below. It is constituted by a cylinder that represents the main body of the vehicle, that is shell of the booster, and only two aerodynamic surfaces,

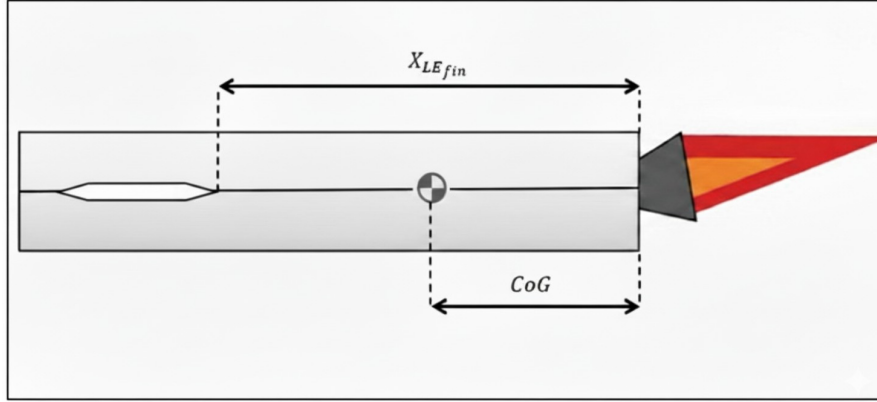


Figure 2.3: Geometry model (non-proportional) for the vehicle first stage.

the fins, mounted with their rotation axis perpendicular to the plane. They are symmetrically positioned and actuated together with the same deflection angle, in order to not generate roll moments, but only pitch moments. Furthermore, the absence of a second couple of fins rotated by 90° with respect to the first ones does not allow to generate yaw moments. As already shown in Table 2.1, fins position



Figure 2.4: 3D model of Falcon 9-class vehicle. See [6]

and center of mass calculation are fixed input design parameter for the simplified vehicle model. In particular, the latter assumption for CoG ignores that during powered landing CoG shifts due to propellant mass expulsion. This choice has been

made considering that the propellant mass used to perform powered landing is very small compared to the dry mass of the booster. Specifically, it depends on burning time and mission scenarios but it has been demonstrated that for typical powered landing scenarios propellant mass used is around 20% of total mass M_0 [7], [8] thus the change of CoG position due propellant mass expulsion is small. Concerning the estimation of these two parameters, the fins are required to be located in the higher part of the booster opposite to the engines due to aerodynamic control reasons explained in the next section, while CoG is expected to be located in the lower part of the booster. In fact, as shown in Figure 2.5 the main elements that make up the first stage vehicle mass are the set of engines, the landing legs, the shell, the two tanks for fuel and oxidizer, the propellant mass and the fins. For the initial configuration of Falcon 9-class vehicle at lift-off, CoG is located at 29.3 m from the base for a total length of the first stage of 70 m [6]. This consideration, united to the fact that, when AD phase and PL phase start, the vehicle consists only in its first stage and in a little percentage of the initial total propellant mass, allows to say that mass is concentrated in the lower part of the booster. Thus, it is reasonable to assume that for the landing configuration the CoG is located around the lower quarter of the booster total length.



Figure 2.5: Elements of Falcon 9-class vehicles. See [6]

2.1.1 Aerodynamics control surfaces: the planar fins

Fins are one of the two control systems modeled to maneuver the vehicle. Controlling their deflection angle, and therefore their angle of attack, they are used during AD phase to modify the aerodynamic resultant force and pitch moment on the vehicle in order to control its translational and attitude dynamics. During PL phase, fins are modeled to be deployed and fixed perpendicular to the longitudinal direction of the vehicle in order to maximize the surface exposed to air flow and use air friction to produce drag and slow down the vehicle. Two main fins design exist: planar fins and grid fins. The first one is a simpler design that makes easier to model the aerodynamics of the fins as shown in next section. Planar fins design must be allow to use them, not only during flight in lower atmosphere, but also during the first phase of aerodynamic descent at higher altitude at hypersonic speed. Thus, classical wing profile can not be used, instead a valid solution can be a rectangular

shaped fin with an hexagon profile [9], as shown in Figure 5.10. Design parameters b_{fin} and c_{fin} define the rectangular surface dimensions.

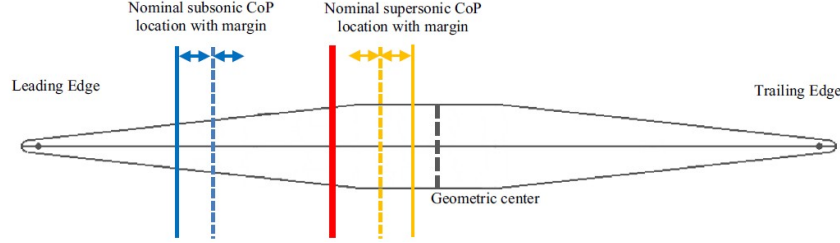


Figure 2.6: Planar fin model. See [9]

2.1.2 Thrust Vector Control system

The second control system used to maneuver the vehicle is the thrust control vector (**TVC**). This system allows to control the thrust vector deflection angle with respect to the longitudinal direction of the vehicle and, therefore, it can control the thrust vector components in the body reference frame creating a resulting moment to modify the attitude; it can also deviate the thrust vector in order to steer the vehicle. TVC system is used together with the throttling control system that allows to modulate the magnitude of thrust vector disbursed by the engines that is evaluated as a percentage of the maximum deliverable thrust in vacuum. This control method is used during PL phase when only one of the engines is reignited. As already mentioned for the fins, the only attitude control required for the DoFs-reduced system dynamics is the one on the pitch moment. This means that TVC is used to only control the deflection angle of thrust in the 2D plane of vehicle as shown in Figure 2.7. The TVC is modeled in this thesis following the gimballed thrust strategy.

2.2 Vehicle Aerodynamics Model

The problem of optimizing the trajectory for the VLV in the lower atmosphere performing a powered landing made it necessary to have a sufficiently accurate model of the aircraft's aerodynamic characteristics, as this was closely linked to the modeling of the control of the system through the fins. In particular, this required the aerodynamics of the vehicle to be represented as a function of the instantaneous state variables of the vehicle in order to ensure active control at every point of the trajectory evolution. This problem could have been easily solved by disposing of an aerodynamic database but given the preliminary approach adopted in this thesis

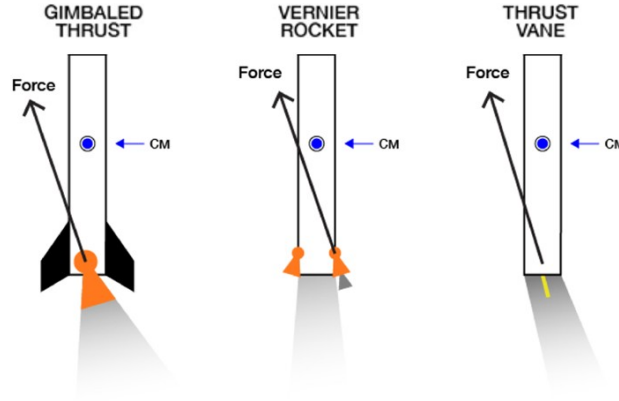


Figure 2.7: Different TVCs strategies. See [10]

and the fact that the vehicle model has been developed ex-novo, a more empirical and fast method has been used to reach the final objective of having relations to estimate aerodynamic coefficients.

In order to better understand how to develop the aerodynamics model, statics stability of a generic space launcher booster during descent was explored. When flying in atmosphere, the vehicle is in air flow and its interaction with it generates aerodynamic forces applied on a specific point of the body called center of pressure **CP**. If the air flow is parallel to the longitudinal direction of the vehicle, only axial aerodynamic force is produced, that is aerodynamic drag that slows down the vehicle, but if an angle of attack exists the resultant aerodynamic force has both an axial component and a normal component with respect to the longitudinal body axis. The second one generates a moment with respect to the center of mass when an off-set between CoG and center of pressure exists, that is the moment arm. The vehicle is said to be statically stable if the aerodynamic moment that is generated by the angle of attack tends to rotate the vehicle to cancel the angle of attack itself, restoring the initial longitudinal air flow conditions. This is fundamental to assure that if an external perturbation disturbs the vehicle flying in a generic trim condition, the vehicle itself tends to restore and keep its initial equilibrium instead of moving away from it increasing the perturbation until losing control. The static stability or instability of a vehicle depends on the static margin defined as the off-set between the CoG and the CP. As shown in Figure 2.8, for a VLV that is descending in the atmosphere, the static stability exists only if the CP is located behind the CoG with respect to the base of the vehicle. This condition is intrinsic in the cylindrical body of the booster due to the fact that it has a low CoG so it always tends to zero angle of attack. Furthermore, depending on where the fins are located, the CP position can be shifted forwards or backwards: in order to increase the static margin and making the complete vehicle more static stable, the fins are

located in the higher part of the vehicle, opposite to the engines. It is important to observe that for nominal orientation of the fins, i.e. longitudinally aligned with the booster body, the only trimmed flight condition is the one at zero angle of attack (AoA). Instead, when flying at a non-zero AoA, movable fins can be deflected in order to produce an aerodynamic resultant force with the desired normal component needed to provide an equal and opposite moment to the restoring one generated by the body of the vehicle, thus the resulting moment of the vehicle is zero and AoA is not nullified. Once that it was clarified how the basic aerodynamics of the

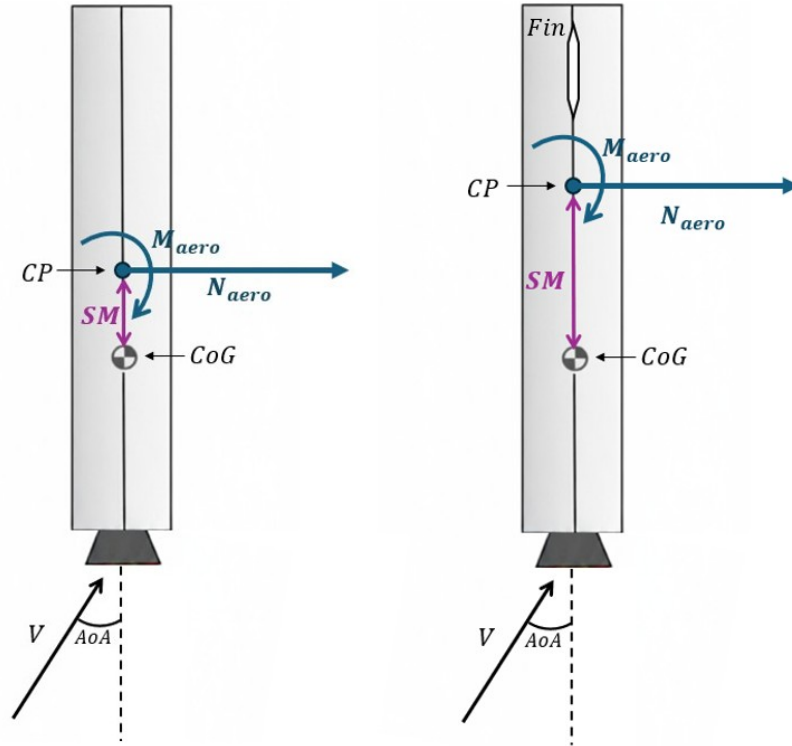


Figure 2.8: Static stability of a descending VLV.

vehicle works, relations to evaluate aerodynamics characteristics were explored. Considering that the trajectory optimization problem of this thesis deals with the final subsonic flight of the vehicle in lower atmosphere, reference was made to the sounding rocket aerodynamics. Sounding rockets are rockets used to collect scientific data and conduct experiments in a microgravity environment or in the upper atmosphere or near space, so their flight profile is similar to the one of first stage launchers when performing final descent and landing. In the sounding rockets aerodynamics field, Barrowman equations represent a reference point. Barrowman has applied the potential flow theory to the common design of model rockets, developing a fast and immediate method able to predict the position of the center

of pressure from the knowledge of the geometry of the aerodynamics only [11]. Equation 2.1 gives the longitudinal position coordinate of fin center of pressure with respect to the base of the cylindrical body:

$$X_{CP_{fin}} = X_{LE_{fin}} + \frac{X_R}{3} \frac{(C_R + 2C_T)}{(C_R + C_T)} + \frac{1}{6} \left[(C_R + C_T) - \frac{(C_R C_T)}{(C_R + C_T)} \right] \quad (2.1)$$

where $X_R = 0$ for rectangular shaped fin because it is the distance between fin root leading edge and fin tip leading edge parallel to body and $C_R = C_T = c_{fin}$ is the fin aerodynamic mean chord, so it results that

$$X_{CP_{fin}} = X_{LE_{fin}} + \frac{1}{4}c$$

i.e. fin center of pressure is located on the front quarter of the fin. In his method Barrowman neglected the normal force developed by the cylindrical body when flying at an AoA because it seems negligible with respect to the normal components developed by the other aerodynamic parts of the rocket, such as fins. Anyway, further research works stated that the normal force is proportional to the AoA and considering that the rocket is oscillating at small range of AoA during the final descent, it is not clear when a component can be neglect with respect to another, thus a corrective term in the Barrowman equations is needed to account for the presence of the body normal force [12]. This force is applied in the center of the planform area, i.e. the surface area of the body cut along its vertical axis that for a 2D cylindrical body is the center of the rectangular surface so the longitudinal position coordinate center of pressure for the cylindrical body is assumed to be:

$$X_{CP_{cyl}} = L/2 \quad (2.2)$$

Note that these simplified relations for center of pressures assume that their position does not change with AoA. Before proceeding with the aerodynamic coefficients evaluation, the definitions of reference systems and notations used in Figure 2.9 are described. The assumptions made are listed here:

- The flight happens in calm atmosphere with no wind so relative air velocity is equal to vehicle inertial velocity;
- The vehicle is flipped i.e. it flies with its engines towards the relative air flow;
- Body reference frame (**BRF**) is the total body fixed frame centered on the vehicle CoG, with X_B parallel to the longitudinal direction of the vehicle pointing towards the engines, Z_B perpendicular to X_B pointing towards up and Y_B completing the right-handed system;

- Wind reference frame (**WRF**) is the reference frame centered on the vehicle CoG, with X_W fixed on the relative air velocity vector, Z_W perpendicular to X_W pointing towards down and Y_W completing the right-handed system;
- Angle of attack is positive when the relative air velocity has a positive Z component with respect to the local body frames fixed on fins or on cylindrical body (the second one coincides with the total BRF). AoA α is the angle of attack for the booster aerodynamics referred to the longitudinal direction of the cylindrical body and computed as $\alpha = \theta - FPA$ where θ is the pitch angle and FPA is the flight path angle. AoA α_{fin} is the angle of attack for the fins aerodynamics;
- Fin deflection angle δ is positive when fin is pitched up.

The aerodynamic coefficients are evaluated with respect to the WRF, so they are referred to as lift and drag coefficients, respectively C_L and C_D , with lift being the aerodynamic force along Z_w and drag being the one along X_W . Note that lift is not intended in the classical way as the force used to sustain the weight but as a steering force to control the vehicle. Furthermore, the problem is 2D and velocity is always in the longitudinal body plane, so there is not a sideslip angle and lateral-directional aerodynamics is not considered but only longitudinal aerodynamic exists.

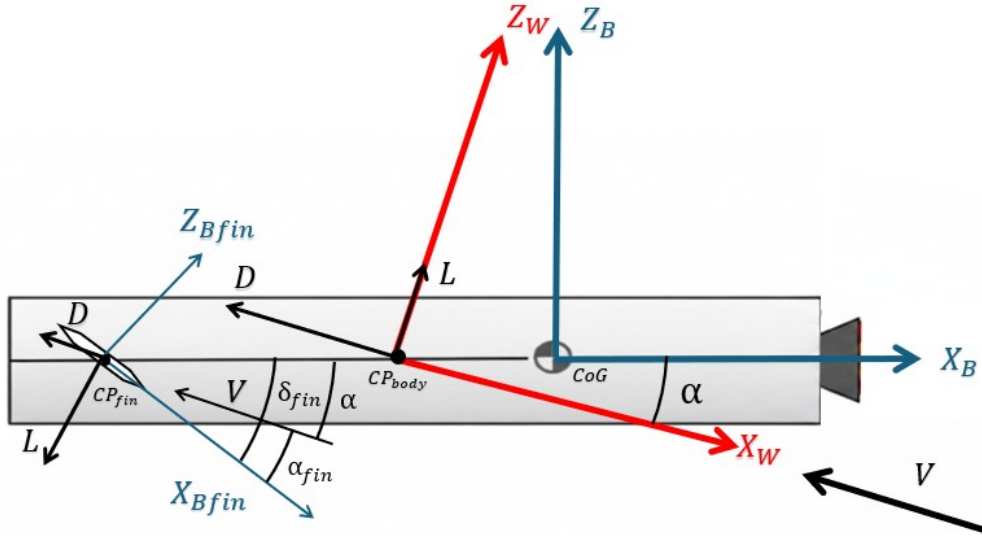


Figure 2.9: System representation for aerodynamics analysis.

For the **cylindrical body aerodynamics**, the relations reported below are analytical equations empirically obtained from sounding rockets aerodynamics studies [11] valid for subsonic flight conditions and small range of AoA that in this thesis work is assumed to be $\alpha \in [-10^\circ, +10^\circ]$:

$$K_{body} = \left[1 + 60 \left(\frac{D_{ref}}{L_{body}} \right)^3 + 0.0025 \frac{L_{body}}{D_{ref}} \right] \frac{S_w}{A_{ref}} \quad (2.3)$$

$$(2.4)$$

$$C_{D_{body}} = \frac{0.074}{Re_y^{\frac{1}{5}}} \quad (2.5)$$

$$(2.6)$$

$$C_{D_{base}} = \left(\frac{D_{base}}{D_{ref}} \right)^3 \frac{0.029}{\sqrt{C_{D_{body}}}} \quad (2.7)$$

$$(2.8)$$

$$C_D = C_{D_{body}} K_{body} + C_{D_{base}} \quad (2.9)$$

$$(2.10)$$

$$C_L = k_{body} \frac{A_p}{A_{ref}} \alpha^2 \quad (2.11)$$

where $D_{ref} = D_{base} = d$, $L_{body} = L$, S_w is the wet surface of the cylindrical body, A_{ref} is the reference area equal to the base area of the cylindrical body and A_p is the cylindrical body plan area. The equation 2.9 shows that the total drag is a sum of two contributions, which are the viscous drag and the pressure drag.

- The viscous or frictional drag $C_{D_{body}}$ derives from the flow viscosity inside the boundary layer developed as the air flows around the vehicle. It depends from the Reynolds number and its formulation differs for laminar and turbulent flows. The one used here is valid for turbulent flow with $Re > 5 \cdot 10^5$ over a flat plate and it is multiplied by a corrective factor K_{body} that takes into account the 3D effect on real-body.
- The pressure drag comes from the projection along the velocity direction of the outer air pressure. Its main contribute is the base drag that derives from the boundary layer separation which occurs at the rear of the rocket.

The equation 2.11 is the correction proposed by Galejs [12] for accounting the body lift. k_{body} is a coefficient to be empirically determined that in this thesis is assumed equal to 1.1 to align the results obtained with the literature.

For the **fins aerodynamics**, the lift and drag coefficients of a single fin are:

$$C_L = 2\pi \frac{AR}{AR + 2} \alpha_{fin} \quad (2.12)$$

$$C_D = \frac{C_L^2}{\pi \cdot AR \cdot e} \quad (2.13)$$

where $\alpha_{fin} = \delta_{fin} + \alpha$, AR is the aspect ratio of the fin and e is the Oswald factor fixed at 0.8. The equation 2.12 evaluates the lift coefficient using the lifting-line theory [13] and equation 2.13 computes the drag coefficient accounting for induced drag and neglecting the pressure drag that is smaller with respect to the pressure drag on the cylindrical body. A variation in the definition of aerodynamic coefficients for fins is made for the PL phase: during this phase the active control system is the TVC system combined with the thrust magnitude throttling system, so, considering that fins are modeled to be fixed and perpendicularly oriented to the longitudinal direction of the body vehicle in order to help slowing down the rocket thanks to their pressure drag. Thus, during this phase:

$$C_L = 0 \quad (2.14)$$

$$C_D = 1.28 \cdot \cos \alpha \quad (2.15)$$

where C_D is the pressure drag coefficient of a flat plate in cross-wise subsonic air flow with a correction that accounts for the misalignment between the fin normal direction and the air flow velocity [14].

2.3 Equations of Motion

Trajectory optimization requires to lastly model the dynamics of the vehicle, i.e. writing the equations of motions in order to be able to propagate the system state and implementing the trajectory optimization method chosen, as explained in 3. As already anticipated, the dynamics problem has been simplified reducing the total DoFs: instead of having a 3D problem with three DoFs for translational and rotational dynamics, a 2D problem has been modeled with three DoFs, two for vertical and horizontal translational dynamics and one for pitching rotational dynamics. Before proceeding with writing the equations of motion, the reference systems used and the assumptions made are listed here and illustrated in Figure 2.10 and Figure 2.11 :

- **ECEF**: Earth Centered Earth Fixed reference frame. It is used to define the initial state of the vehicle through latitude and longitude coordinates at the initial time to be converted in the local reference frame;

- **ENU**: East North Up reference frame. It is a local reference frame with its origin centered on the landing target. X_E and Y_N axis define the local horizon plane, X_E points towards the East and Y_N points towards the North. Z_U is perpendicular to the local horizon plane. it points towards up and it defines the local vertical direction. This local reference frame is not actually inertial but it is assumed to be inertial because the trajectory evolution studied with reference to it lasts for a short enough time to ignore the Coriolis acceleration and the centrifugal acceleration;
- The **g** gravitational acceleration is assumed to be constant and always parallel to the local vertical direction;
- Given the 2D problem assumption, at the initial time the vehicle is assumed to be already positioned on the $Y_N = 0$ target landing position coordinate, thus the trajectory evolves only in the **local horizontal-vertical plane** $X_E Z_U$;
- **Altitude** refers to the distance of the vehicle with respect to the landing target along Z_U axis, while **down-range** is the distance of the vehicle with respect to the landing target along X_E axis;
- The **pitch angle** θ is the angle between X_B and the local horizon. It is zero when X_B is aligned with the local horizon and it is negative when X_B is pitching down, so the desired vertical attitude to reach at landing is represented by $\theta = -\frac{\pi}{2}$ and pitch angle is defined in the range $\theta \in [0^\circ, -180^\circ]$;
- The **thrust deflection angle** β is defined as the angle between the thrust direction and X_B axis. It is positive when TVC system pitches the engine's nozzle down;
- **Translational dynamics** equations are written in **ENU** reference frame assumed to be inertial and the vehicle is modeled as a point mass: in Figure 2.10 the resultant aerodynamic forces, thrust and weight are applied on the CoG;
- **Rotational dynamics** equations are written in **BODY** reference frame. To do so, reference is made to Figure 2.11 where all the forces acting on the vehicle are showed on their actual application point from which pitching moment arms can be computed. In rotational dynamics, the body is assumed to be rigid and mass is assumed to be uniformly distributed.

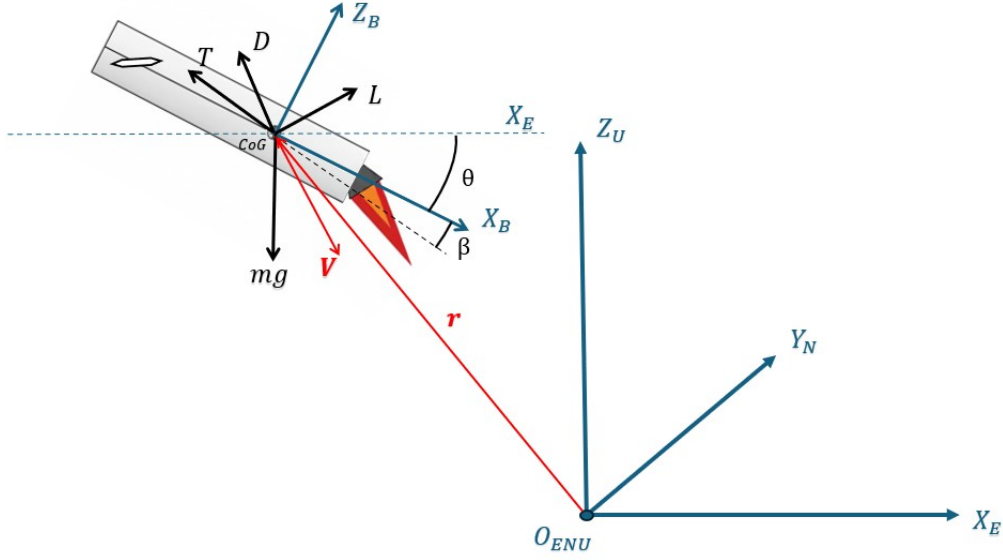


Figure 2.10: System representation for translational equations of motion.

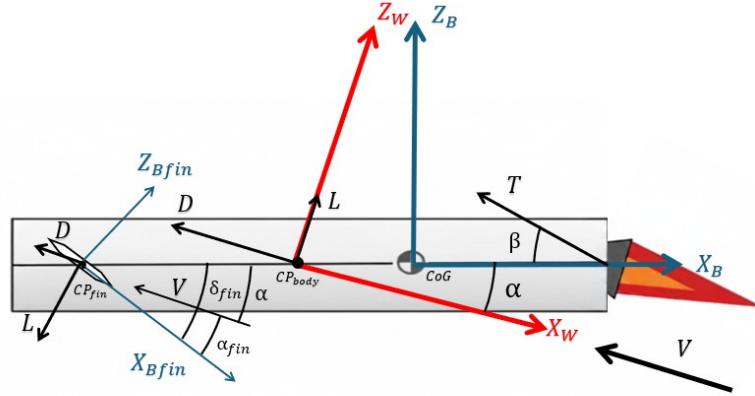


Figure 2.11: System representation for rotational equations of motion.

In the previous section, aerodynamic coefficients for the cylindrical body and for the fins are defined in the WRF:

$$C_{AERO_{cyl}} = \begin{bmatrix} -C_{D_{cyl}} \\ 0 \\ C_{L_{cyl}} \end{bmatrix} \quad (2.16)$$

$$C_{AERO_{fin}} = \begin{bmatrix} -C_{D_{fin}} \\ 0 \\ C_{L_{fin}} \end{bmatrix} \quad (2.17)$$

Using the AoA it is possible to define the rotation matrix R_{W2B} that allows to compute the aerodynamic coefficients expressed with respect to the BRF, starting from their values expressed with respect to the WRF:

$$R_{W2B} = \begin{bmatrix} \cos \alpha & 0 & \sin \alpha \\ 0 & 1 & 0 \\ -\sin \alpha & 0 & \cos \alpha \end{bmatrix} \quad (2.18)$$

$$C_{AERO_{cyl}}^{body} = \begin{bmatrix} C_{A_{cyl}} \\ 0 \\ C_{N_{cyl}} \end{bmatrix} = R_{W2B} \cdot C_{AERO_{cyl}} \quad (2.19)$$

$$C_{AERO_{fin}}^{body} = \begin{bmatrix} C_{A_{fin}} \\ 0 \\ C_{N_{fin}} \end{bmatrix} = R_{W2B} \cdot C_{AERO_{fin}} \quad (2.20)$$

Finally, using the quaternion to avoid gimbal lock singularity for rotation matrix, q_{B2ENU} is defined using the pitch angle θ and it allows to compute the aerodynamic coefficients expressed with respect to the ENU frame starting from the ones in the BRF:

$$q_{B2ENU} = \begin{bmatrix} \cos \theta/2 \\ 0 \\ \sin \theta/2 \\ 0 \end{bmatrix} \quad (2.21)$$

$$C_{AERO_{cyl}}^{enu} = \begin{bmatrix} C_{X_{E_{cyl}}} \\ 0 \\ C_{Z_{U_{cyl}}} \end{bmatrix} = q_{B2ENU} \cdot C_{AERO_{cyl}}^{body} \cdot q_{B2ENU}^{-1} \quad (2.22)$$

$$C_{AERO_{fin}}^{enu} = \begin{bmatrix} C_{X_{E_{fin}}} \\ 0 \\ C_{Z_{U_{fin}}} \end{bmatrix} = q_{B2ENU} \cdot C_{AERO_{fin}}^{body} \cdot q_{B2ENU}^{-1} \quad (2.23)$$

To express thrust vector components in BRF and in ENU reference frame the process is similar to the one followed for the aerodynamic coefficients. Given the magnitude of the thrust vector $\|\mathbf{T}\| = \%T_{sl}$, its components with respect to the BRF are:

$$\mathbf{T}_{body} = \|\mathbf{T}\| \cdot \begin{bmatrix} -\cos \beta \\ 0 \\ \sin \beta \end{bmatrix} \quad (2.24)$$

and in ENU reference frame:

$$\mathbf{T}_{ENU} = q_{B2ENU} \cdot \mathbf{T}_{body} \cdot q_{B2ENU}^{-1} \quad (2.25)$$

The **translational dynamics equations** in ENU reference frame are:

$$\dot{\mathbf{r}}(t) = \mathbf{v}(t) \quad (2.26)$$

$$\dot{\mathbf{v}}(t) = \mathbf{a}(t) \quad (2.27)$$

$$\mathbf{a}(t) = \frac{\mathbf{F}_{AERO_{fin}}(t) + \mathbf{F}_{AERO_{cyl}}(t) + \mathbf{T}_{ENU}(t)}{m(t)} + \mathbf{g}_0 \quad (2.28)$$

$$\dot{m}(t) = -\frac{\|\mathbf{T}(t)\|}{I_{sp}g_0} - \frac{A_{cp_{atm}}(t)}{I_{sp}g_0} \quad (2.29)$$

where

$$\mathbf{r}(t) = [r_x(t), r_z(t)]$$

$$\mathbf{v}(t) = [v_x(t), v_z(t)]$$

$$\mathbf{F}_{AERO_{cyl}}(t) = \frac{1}{2}\rho(t)\|\mathbf{v}(t)\|^2 A_{ref} \begin{bmatrix} C_{X_{E_{cyl}}}(t) \\ C_{Z_{U_{cyl}}}(t) \end{bmatrix}_{aero}$$

$$\mathbf{F}_{AERO_{fin}}(t) = 2 \cdot \frac{1}{2}\rho(t)\|\mathbf{v}(t)\|^2 A_{ref} \begin{bmatrix} C_{X_{E_{fin}}}(t) \\ C_{Z_{U_{fin}}}(t) \end{bmatrix}_{aero}$$

The **rotational dynamics equations** written with respect to the CoG in BRF are:

$$\dot{\theta}(t) = \ddot{\theta}(t) \quad (2.30)$$

$$\ddot{\theta}(t) = \frac{N_T(t) \cdot x_{CoG} - N_{AERO}^{cyl}(t) \cdot (x_{CP_{cyl}} - x_{CoG}) - N_{AERO}^{fin}(t) \cdot (x_{CP_{fin}} - x_{CoG})}{I_{yy}(t)} \quad (2.31)$$

Since the only DoF for the rotational dynamics is the pitch moment, the scalar form of the equation is directly written, where pitch moments contributions are given by the normal components of the forces acting on the vehicle:

$$N_T(t) = \|\mathbf{T}(t)\| \cdot \sin \beta(t)$$

$$N_{AERO}^{cyl}(t) = \frac{1}{2}\rho(t)\|\mathbf{v}(t)\|^2 A_{ref} C_{N_{cyl}}(t)$$

$$N_{AERO}^{fin}(t) = 2 \cdot \frac{1}{2}\rho(t)\|\mathbf{v}(t)\|^2 A_{ref} C_{N_{fin}}(t)$$

The pitch inertia moment I_{yy} with respect to the Y_B axis is computed assuming uniform distribution of mass around the longitudinal axis of the body:

$$I_{yy}(t) = \frac{1}{12}m(t) \left(3 \left(\frac{d}{2} \right)^2 + L^2 \right)$$

This rotational dynamics equation, in the case of $\|\mathbf{T}(t)\| = 0$ when engines are off, is used to also study aerodynamic trim conditions during aerodynamic descent. In fact, the resultant aerodynamic moment around the CoG can be imposed to be zero, in order to solve the equation with respect the fin deflections δ_{fin} that assure no-pitch moment while flying at different values of angle of attacks. This dynamics equations are valid both for the AD phase and the PL phase with the only difference that during AD phase the thrust relative terms must not be accounted because engines are off. In the same way, the total mass variable $m(t)$ is a state variable only during PL phase because it varies during trajectory evolution due to propellant mass expulsion. The propellant mass flow rate is equal and opposite to the total vehicle mass decrease rate as described by equation 2.29 where the second term accounts for the back-pressure losses due to non-negligible ambient pressure; this term adds to the first one which accounts for the propellant mass consumed proportionally to the commanded thrust magnitude.

Chapter 3

Trajectory Optimization

This chapter provides the basic knowledge to do Trajectory Optimization for a generic dynamic system. First, the Optimal Control Problem theory is described and how it should be formulated. Then, Direct Collocation Method is analyzed in order to finally set up the resolution of the aerodynamic descent and powered landing problem in the software tool.

3.1 The Optimal Control Problem

Trajectory optimization problem is an Optimal Control Problem (**OCP**) that arises from the combination of Optimization theory and Control theory. According to Optimization theory, given any system and any number of conditions on it, a feasible system configuration must be determined that can not be improved with respect to a specified criterion. At the same time, Control theory aims at governing the application of system inputs to obtain a desired system state. Thus, OCP results in the finding of a control for a dynamical system over a period of time to optimize a given criterion. The system state and its control are time functions to be determined [15]. To better understand how to solve an OCP and, more specifically, the vertical landing trajectory optimization problem of a reusable first stage launcher, it is necessary to know how an OCP is formulated.

The most general formulation for the system dynamics is a first order system of non-linear time-varying differential equations

$$\dot{x}(t) = f(x(t), u(t), t) \quad (3.1)$$

where $x(t) \in \mathbb{R}^n$ is the vector of the state variables of the system and $u(t) \in \mathbb{R}^m$ is the vector of control parameters. Since $x(t)$ and $u(t)$ are vectors of time continuous functions defined on functional spaces, the OCPs are infinite-dimensional problems.

$x_{opt}(t)$ and $u_{opt}(t)$ are the OCP solutions that minimize or maximize the performance index identified also as objective or cost. It is described by the following functional:

$$J_B = \phi(x(t_0), t_0, x(t_f), t_f) + \int_{t_0}^{t_f} \Phi(x(t), u(t), t) dt \quad (3.2)$$

t_0 and t_f are independent variables, respectively the initial and final time instants of trajectory evolution. This formulation is identified as Bolza's form. If $\Phi = 0$, the performance index formulation is said Mayer's form and it is

$$J_M = \phi(x(t_0), t_0, x(t_f), t_f) \quad (3.3)$$

while if $\phi = 0$, it is identified as Lagrange's formulation and it is

$$J_L = \int_{t_0}^{t_f} \Phi(x(t), u(t), t) dt \quad (3.4)$$

According to Mayer's form, the performance index is a boundary objective and it is penalized based only on the initial and final system states. Instead, Lagrange's form states that the performance index is a path integral along the trajectory covered in the time interval so that it is penalized based on the accumulation of a cost during the trajectory evolution. Bolza's form is a combination of both.

The performance index formulations in 3.3, 3.4 and 3.2 are valid for single-phase trajectory optimization problems. In fact, there are many trajectory optimization problems that have a sequence of continuous-motion phases separated by discrete jumps. One common example is the trajectory of a multistage rocket. For these problems J_L performance index term must be formulated as a sum of integral terms defined on each trajectory arc. The aerodynamic descent and powered landing problem of this thesis work could be formulated as a multi-phase problem but in order to not make it too difficult to implement the problem as first approaching to it, it has been decided to solve two single-phase problems separately.

Defining a performance index for the OPC is a necessary but not sufficient condition: constraints have to be defined in order to have a solution to the OPC that is not just mathematically, but also physically, possible. The solutions that satisfy the constraints are said feasible. The solutions that satisfy the constraints and provide also the minimum or maximum value of the objective function are said optimal.

- The first and also the most important type of constraints are the **dynamics constraints** already formulated in equation 3.1: they assure that the optimal solution computed for the controls vector $u_{opt}(t)$ and the state variables vector $x_{opt}(t)$ provide a trajectory evolution that satisfies the system dynamics.
- **Path constraints** define restrictions on state variables and control parameters along the trajectory. They are defined as equalities or inequalities respect to 0.

$$h(t, x(t), u(t)) \leq 0 \quad (3.5)$$

- **Boundary constraints** define restrictions on the initial and final states variables. They are defined as equalities or inequalities respect to 0.

$$g(t_0, x(t_0), t_f, x(t_f)) \leq 0 \quad (3.6)$$

To complete the OCP formulation, constant bounds on the state variables and the control parameters can be set:

$$x_{low} \leq x(t) \leq x_{upp} \quad (3.7)$$

$$u_{low} \leq u(t) \leq u_{upp} \quad (3.8)$$

Finally, local and global optimality are important concepts to know in order to proceed with solving OPC. A globally optimal solution is a feasible solution with the best possible objective value. In general, the global optimum for a problem is not unique. By contrast, a locally optimal solution has the best possible objective value within an open neighbourhood around it. For a convex problem, every local optimum is a global optimum. A problem is said to be convex when it includes only linear or convex quadratic objective function subjected to convex quadratic constraints. When a problem is of a more general nonlinear type, there will typically be many local optima, which are potentially widely spaced, or even in parts of the feasible region which are not connected [16]. The trajectory optimization addressed in this thesis is a non-linear problem that is solved using local optimal software solver, so the solutions found are optimal for a certain region of the input space

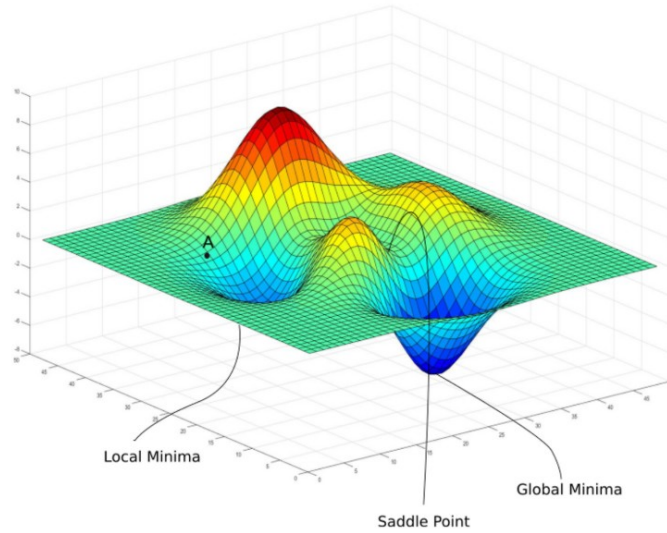


Figure 3.1: Example of local and global optima of a non-linear function. See [17]

targeted by the initial guesses as described in 4. Figure 3.1 shows an example of a non-linear function with local and global minima. Given point A as initial guess, it can be observed that a locally optimal solver finds an optimal solution in the proximity space region that is a local optimal. Starting from a different point could allow the local solver to find a global optimal.

3.2 The Direct Collocation Method

Solving an OCP is a big challenge and since the birth of optimization theory different strategies have been proposed. Figure 3.2 shows how the main strategies to solve an OCP can be categorized. At the highest level there is the distinction

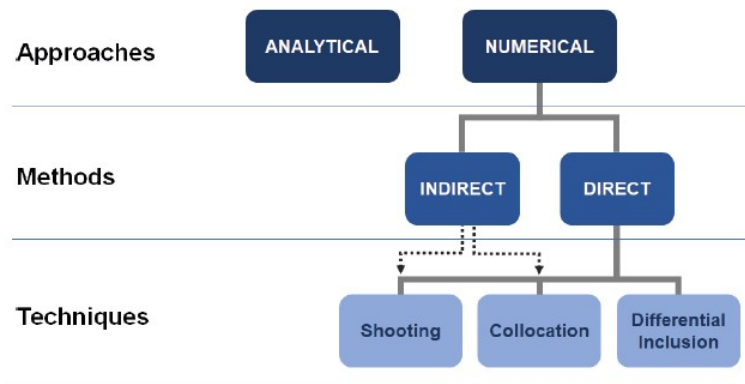


Figure 3.2: OCPs resolution strategies. See [18]

between analytical and numerical approaches. Analytical approach consists in using Pontryagin’s minimum principle that leads to definition and resolution of complex ODEs system. Analytical solutions exist only for linear-quadratic OCPs. Being the trajectory optimization problem a non-linear OCP, the detailed examination of this approach is beyond the objective of this thesis while numerical approach is further examined. Numerical approach leverages computational algorithms to numerically solve the OPCs so the solution obtained is not anymore a set of time-continuous functions but a set of time-discrete numerical values. In fact, every numerical method requires the problem to be discretized but according to how discretization is used different methods can be identified. The indirect methods use calculus of variations and Lagrange multipliers to determine the necessary and sufficient optimality conditions which are then discretized and solved numerically. Thus, indirect methods do not deal directly with the objective function, but rather with the numerical resolution of the equations that describe the optimality conditions obtained with the analytical approach. On the other hand, direct methods attempt to find a minimum of the objective function by discretizing the states and controls

and then transcribing the optimal control problem to a nonlinear programming (**NLP**) problem that is solved using optimization techniques. In short, the direct method constructs a sequence of potential solutions $x(t), u(t)$ so that each of them is an improvement on the last one, resulting in $J(x_1(t), u_1(t)) > J(x_2(t), u_2(t)) > \dots > J(x_{opt}(t), u_{opt}(t))$ [19]. As far as concerns for techniques, differential inclusion is an exclusive technique for direct methods that is numerically instable and problem specific [20] so it has not been further examined. Shooting is a very simple technique that consists of choosing an initial guess for boundary conditions at initial time and then propagating the system dynamics until final time is reached. Once it is reached, an error between the system trajectory and the boundary conditions exists so the initial guess is adjusted and the process is repeated in order to reduce this error. Multiple shooting is a variant of it where the problem is broken down in shorter steps between initial and final time in order to reduce the sensitivity of the problem. Although this technique is simple, it can handle only problems with a small quantity of variables due to its high sensitivity to initial guess so it is not suitable for this thesis work where the vehicle system needs to be actively controlled at each time instant during the all trajectory evolution, resulting in a great amount of state and control variables problem. Finally, direct collocation method (**DCM**) has been chosen to solve the trajectory optimization problem for this thesis work. As already mentioned, DCM requires to transcript the time-continuous functions for state variables and control parameters in time-discrete numerical values in order to have a finite-dimensional problem called NLP which is a constrained parameter optimization problem that has non-linear terms in either its objective or its constraint function. The discretization of the time, states and controls is done by representing the continuous states and controls by values in specific points in time called collocation nodes. The values of the state and control variables in collocation nodes are the decision variables that must be computed by a numerical optimizer. The functions of states and controls can be approximated on each interval between two collocation points using splines. Figure 3.3 shows how a generic time-continuous function, that could represent a state variable, a state dynamics function or a control parameter, appears once it has been discretized on the collocation points and interpolated between them using a spline, specifically a linear one. The discretized variables are defined as below:

$$t \rightarrow t_0 \cdots t_k \cdots t_N \quad (3.9)$$

$$x(t) \rightarrow x_0 \cdots x_k \cdots x_N \quad (3.10)$$

$$u(t) \rightarrow u_0 \cdots u_k \cdots u_N \quad (3.11)$$

where N is the total number of collocation nodes. The system dynamics equation formulated in 3.1 for time-continuous problem can be reformulated in integral form

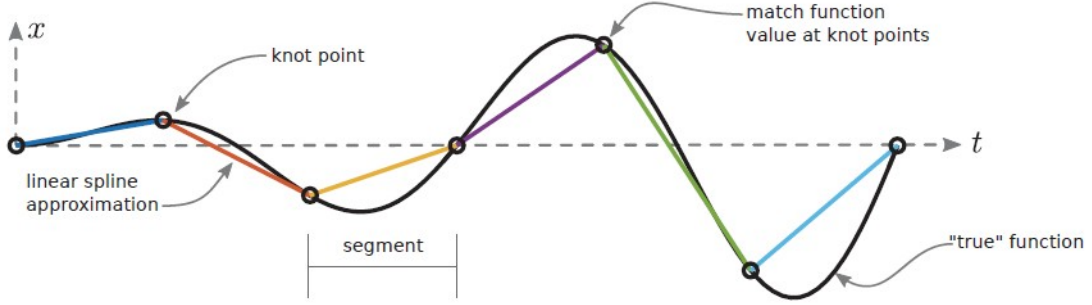


Figure 3.3: Example of function approximation using linear spline. See [21]

between two generic instants of time:

$$x(t_f) - x(t_0) = \int_{t_0}^{t_f} f(x(t), u(t), t) dt \quad (3.12)$$

The integral form allows to define dynamics constraints for discretized state variables but in order to do so the continuous integral $\int_{t_0}^{t_f} f(x(t), u(t), t) dt$ must be approximated with a summation $\sum_{k=1}^{k=N} c_k f_k$ with f_k being the discrete system dynamics evaluated at node x_k . When using direct collocation method, integral dynamics constraints are meant to be used as *collocation constraints*:

$$\zeta_k = x_{k+1} - x_k - \sum_k^{k+1} c_k f_k \quad \forall k \in \{1, \dots, N-1\} \quad (3.13)$$

i.e. the numerical optimizer must compute the discrete states and controls to obtain the optimal solution assuring that the defect value of ζ_k is zero $\forall k \in \{1, \dots, N-1\}$. In order to better understanding the functioning of the direct collocation method, Figure 3.4 contains a blue sphere that represents a collocation node of a discretized state x_k at time t_k . The straight line s is pertinent to the numerical derivative between the points x_{k+1} and x_k and the straight line r is relative to the system dynamics at points x_k and x_{k+1} , that is $f_k = f(x_k, u_k, t_k)$ and $f_{k+1} = f(x_{k+1}, u_{k+1}, t_{k+1})$. The integral difference ζ_k can be imagined to graphically represent the angle between the derivatives slopes. With the direct collocation method, the collocation node is moved up and down according to the purple arrows (i.e. the state x_k has its value increased and decreased) in order to produce $\zeta_k = 0$, making the straight lines r and s coincide. There can be several positions for the collocation point that satisfy the condition $\zeta_k = 0$, such positions are considered admissible. The point is then fixed at an admissible position that best contributes to the minimization of the cost function. This process is performed for each of the N collocation points in order to obtain a vector with the optimal discrete states

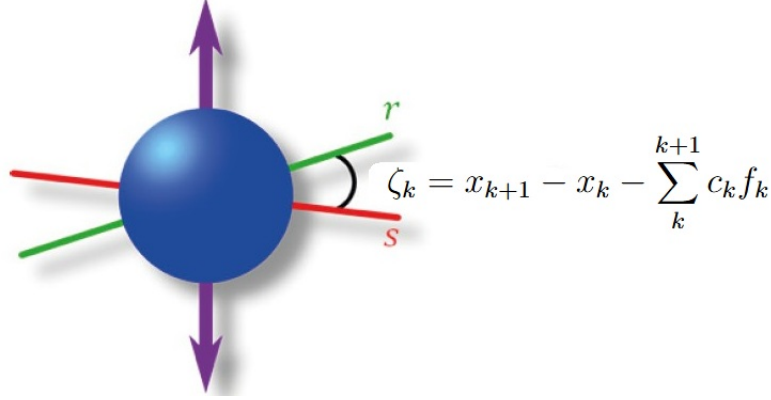


Figure 3.4: Collocation nodes evaluation process. See [22]

and controls that are the solution of the nonlinear programming problem and consequently of the OCP.

Splines of different orders can be used to approximate discrete system dynamics on collocation nodes and based on the spline order used, different transcription methods can be formulated. Here two main examples are presented.

- **Trapezoidal DCM**

Trapezoidal collocation works by approximating the control trajectory and the system dynamics as a linear spline. Consequently, the trajectory, i.e. the discrete state variables approximation, is represented by a quadratic spline. For trapezoidal DCM, the collocations constraints are met when

$$x_{k+1} = x_k + \frac{1}{2}h_k(f_k + f_{k+1}) \quad \forall k \in \{1, \dots, N-1\} \quad (3.14)$$

where $h_k = t_{k+1} - t_k$ and the summation $\sum_k^{k+1} c_k f_k$ to approximate the continuous integral is computed using trapezoidal quadrature. Interpolated values of system dynamics and control variables on a generic instant between two consequential collocation nodes are, respectively:

$$\bar{f}(\bar{t}) = f_k + \frac{\tau}{h_k}(f_{k+1} - f_k) \quad (3.15)$$

$$\bar{u}(\bar{t}) = u_k + \frac{\tau}{h_k}(u_{k+1} - u_k) \quad (3.16)$$

where $\bar{\cdot}$ symbol indicates the approximation made by interpolating and $\tau = \bar{t} - t_k$.

Interpolated values of system state variables are:

$$\bar{x}(\bar{t}) = x_k + f_k \tau + \frac{\tau^2}{2h_k}(f_{k+1} - f_k) \quad (3.17)$$

- **Hermite-Simpson DCM**

The Hermite-Simpson collocation is similar to trapezoidal collocation, but it provides a solution that is higher-order accurate. This is because trapezoidal collocation approximates the control trajectory and the system dynamics as piecewise linear functions, while Hermite-Simpson collocation approximates them as piecewise quadratic functions, as shown in Figure 3.5. Consequently, the trajectory, i.e. the discrete state variables approximation, is represented by a cubic spline. For Hermite-Simpson DCM, the collocations constraints

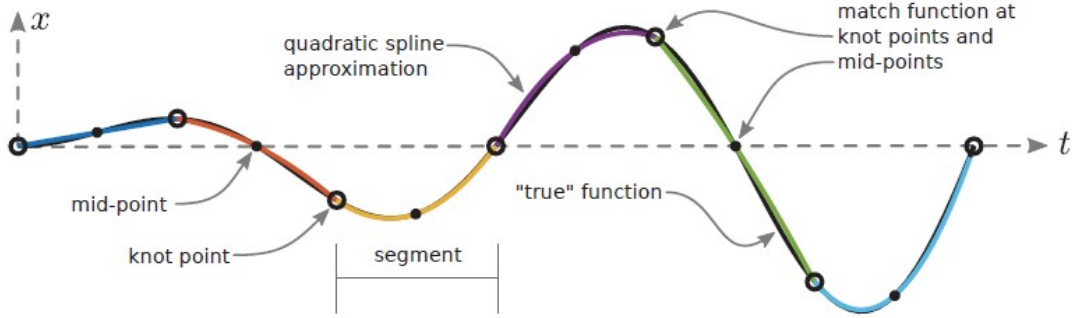


Figure 3.5: Quadratic spline interpolation for system control and dynamics when using Hermite-Simpson DCM. See [23]

are met when

$$x_{k+1} = x_k + \frac{1}{6}h_k(f_k + 4f_{k+\frac{1}{2}} + f_{k+1}) \quad \forall k \in \{1, \dots, N-1\} \quad (3.18)$$

where $h_k = t_{k+1} - t_k$ and the summation $\sum_k^{k+1} c_k f_k$ to approximate the continuous integral is computed using Simpson quadrature. For Hermite-Simpson collocation a second collocation equation is needed to enforce the dynamics. This is because the dynamics at the midpoint of the segment $f_{k+\frac{1}{2}}$ are a function of the state $x_{k+\frac{1}{2}}$ which is not known a priori. It can be computed by constructing an interpolant for the state trajectory and then evaluating it at the midpoint of the interval:

$$x_{k+\frac{1}{2}} = \frac{1}{2}(x_k + x_{k+1}) + \frac{h_k}{8}(f_k - f_{k+1}) \quad (3.19)$$

This second collocation equation is special in that it can be computed explicitly in terms of the state at the collocation nodes.

Interpolated values of system dynamics and control variables on a generic instant between two consequential collocation nodes are, respectively:

$$\bar{f}(\tau) = \frac{2}{h_k^2} \left(\tau - \frac{h_k}{2} \right) (\tau - h_k) f_k - \frac{4}{h_k^2} (\tau - h_k) f_{k+\frac{1}{2}} + \frac{2}{h_k^2} \tau \left(\tau - \frac{h_k}{2} \right) f_{k+1} \quad (3.20)$$

$$\bar{u}(\bar{t}) = \frac{2}{h_k^2} \left(\tau - \frac{h_k}{2} \right) (\tau - h_k) u_k - \frac{4}{h_k^2} (\tau - h_k) u_{k+\frac{1}{2}} + \frac{2}{h_k^2} \tau \left(\tau - \frac{h_k}{2} \right) u_{k+1} \quad (3.21)$$

where $\bar{\cdot}$ symbol indicates the approximation made by interpolating and $\tau = \bar{t} - t_k$.

Interpolated values of system state variables are:

$$\begin{aligned} \bar{x}(\bar{t}) = & x_k + f_k \left(\frac{\tau}{h_k} \right) + \frac{1}{2} \left(-3f_k + 4f_{k+\frac{1}{2}} - f_{k+1} \right) \left(\frac{\tau}{h_k} \right)^2 + \\ & \frac{1}{3} \left(2f_k - 4f_{k+\frac{1}{2}} + 2f_{k+1} \right) \left(\frac{\tau}{h_k} \right)^3 \end{aligned} \quad (3.22)$$

In addition to these two transcription methods, there are others that can be more or less accurate depending on the order of the spline used to approximate the dynamics and control of the system. Among these, the Euler method is the simplest but also the least accurate because it uses the rectangle quadrature method. In contrast, the fourth-order Runge Kutta method uses a cubic polynomial for dynamics and control, resulting in even greater accuracy than the Hermite-Simpson method. However, it is also clear that as the accuracy of the method increases, so do the steps required to implement it, making it more complex and computationally expensive. Given these considerations, the trapezoidal method was chosen to implement the NLP of this thesis because it ensures an average level of accuracy while remaining easy to implement and therefore suitable for initial software development. Evaluating the accuracy of a specific DCM refers to quantifying the error that was introduced by the choice of discretization. There are many possible error metrics for trajectory optimization. One of them is the error estimation based on how well the candidate trajectory satisfies the system dynamics between the collocation points. The logic is that if the system dynamics are accurately satisfied between the collocation points, i.e. the error is little relatively to the specific problem, then the polynomial spline is an accurate representation of the system, which would then imply that the nonlinear program is an accurate representation of the original trajectory optimization problem [24]. An expression for the error in the solution to the system dynamics along the candidate trajectory can be constructed as follows:

$$\varepsilon(\bar{t}) = |\dot{\bar{x}}(\bar{t}) - \bar{f}(\bar{t}, \bar{x}(\bar{t}), \bar{u}(\bar{t}))| \quad (3.23)$$

where $\dot{\bar{x}}(\bar{t})$ is the system dynamics evaluation in the interpolation points $\bar{x}(\bar{t})$ that approximate the state variables between two collocation nodes and $\bar{f}(\bar{t}, \bar{x}(\bar{t}), \bar{u}(\bar{t}))$ is the system dynamics interpolated values. This error $\varepsilon(\bar{t})$ will be, by definition, zero at each collocation point and nonzero elsewhere as shown by the example in figure 3.6.

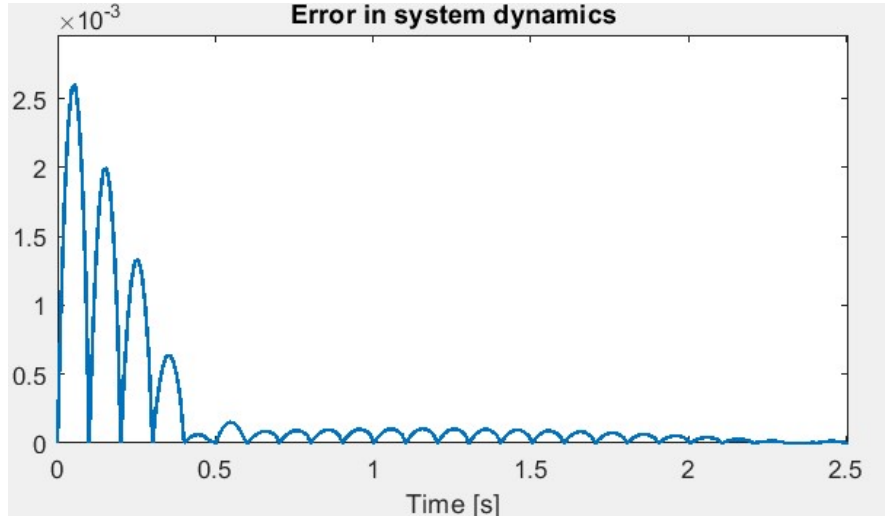


Figure 3.6: Example of error for system dynamics

3.3 Trajectory Optimization for Vertical Landing First Stage Vehicle

Once the dynamic model of the system, the theory for formulating the OCP, and the method for solving it using direct collocation have been established, it is possible to set up the trajectory optimization for the specific case of the aerodynamic descent and powered landing of the first stage of a launch vehicle. As mentioned above, this problem was handled by solving the two flight phases separately as single-phase optimization problems as shown in Figure 3.7. This choice was mainly due to the fact that it simplified the implementation of the overall problem. In fact, even though the method for solving the individual phases is the same, there are differences in the modeling of the system dynamics and control and in the calculation of the objective function.

The two tables below contain the equations and inequalities used to set up the trajectory optimization in the two distinct phases. The information necessary for a complete description of the choices made to formulate the objectives and constraints is provided below.

Table 3.1 refers to the **Aerodynamic Descent** phase. The objective of trajectory optimization in this phase is to slow down the vehicle bringing it closer to the landing target by reducing the downrange distance with reference to the target position coordinate X_T , without descending below a certain altitude, using the control provided by the fins that modulate the aerodynamic force on the vehicle,

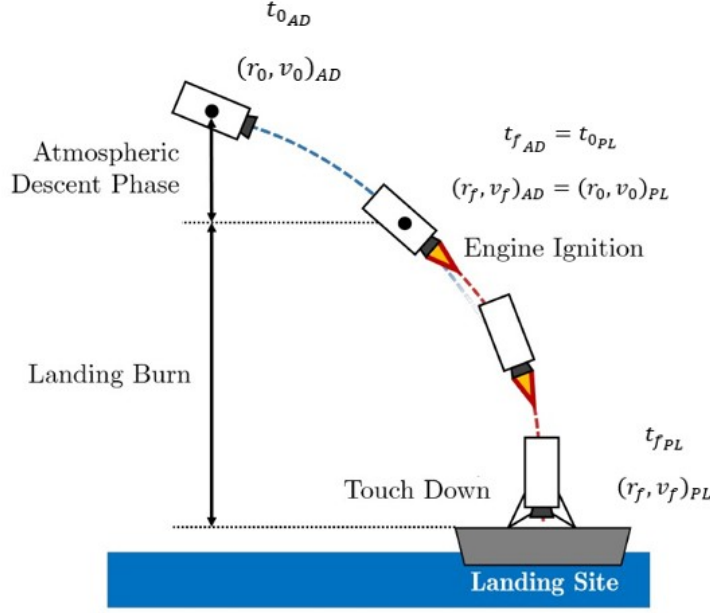


Figure 3.7: Flight phases of VLV. See [25]

allowing both its position dynamics and its attitude dynamics around the pitch rotation axis to be controlled. This phase aims at reaching a final state that allows to successfully perform powered landing.

Using DCM, the total time interval and trajectory arc are divided into $N - 1$ intervals using N equidistant collocation nodes, each identified by the k -th value. Each interval has a duration of $h_k = t_{k+1} - t_k$.

The state vector variables are imposed to respect minimum and maximum constant bounds:

$$[X_{min}, Z_{min}, V_{X_{min}}, V_{Z_{min}}, \Theta_{min}, \dot{\Theta}_{min}]$$

$$[X_{max}, Z_{max}, V_{X_{max}}, V_{Z_{max}}, \Theta_{max}, \dot{\Theta}_{max}]$$

These bounds can be fixed for each collocation node or be specific for certain nodes. The angle of attack is identified by α and its range is always fixed between -10° and 10° as explained in Chapter 2.

The fins deflection angle δ is the actual control parameter to be computed in order to modify the vehicle aerodynamics. In order to guarantee a feasible control actuation of the fins, their rotational velocity is bound to not exceed a maximum value ω_{max} . Considering that for trapezoidal DCM, control function is approximated with a linear spline, fins rotational velocity is a constant value defined for each time interval as $\frac{\delta_{fin}(k+1) - \delta_{fin}(k)}{h_k}$. Furthermore, fins deflection δ is bound to

not exceed a range of values identified by δ_{max} and determined using the trim conditions described in Chapter 2. Finally, the total time for trajectory duration is bound to minimum and maximum values t_{min} and t_{max} .

Table 3.2 refers to the **Powered Landing** phase. The objective of trajectory optimization in this phase is minimizing the fuel mass consumed to land the vehicle with a specific state condition: the vehicle must reach the landing target at zero velocity with a vertical attitude. In order to facilitate the optimization process, these final conditions for system state are provided with small tolerance ranges. Using DCM, the total time interval and trajectory arc are divided into $N - 1$ intervals using N equidistant collocation nodes, each identified by the k -th value. Each interval has a duration of $h_k = t_{k+1} - t_k$.

The state vector variables are imposed to respect minimum and maximum constant bounds:

$$\begin{aligned} &[X_{min}, Z_{min}, V_{X_{min}}, V_{Z_{min}}, \Theta_{min}, \dot{\Theta}_{min}] \\ &[X_{max}, Z_{max}, V_{X_{max}}, V_{Z_{max}}, \Theta_{max}, \dot{\Theta}_{max}] \end{aligned}$$

. These bounds can be fixed for each collocation node or be specific for certain nodes. The angle of attack is identified by α and its range is always fixed between -10° and 10° as explained in Chapter 2.

The actual control is realized thanks to thrust vector deflection and magnitude regulation so the control parameters are the β_{TVC} , which is the deflection angle of thrust actuated by the TVC, and the $T\%$, which is the percentage of the maximum thrust T_{sl} at sea level conditions of a single engine. In order to guarantee a feasible control actuation, TVC system rotational velocity and thrust magnitude changing rate are bound to not exceed a maximum value $\Omega_{\beta, max}$ and $\dot{T}\%$. Furthermore, TVC deflection and thrust percentage are bound to not exceed a range of minimum and maximum values. Finally, the total time for trajectory duration is bound to minimum and maximum values t_{min} and t_{max} .

Aerodynamic Descent

Objective function

$$\min \quad r_x(k) - X_T, \quad k = N \quad (3.24)$$

Collocation constraints:

$$r_x(k+1) = r_x(k) + 0.5 \cdot h_k \cdot (\dot{r}_x(k) + \dot{r}_x(k+1)) \quad (3.25)$$

$$r_z(k+1) = r_z(k) + 0.5 \cdot h_k \cdot (\dot{r}_z(k) + \dot{r}_z(k+1)) \quad (3.26)$$

$$v_x(k+1) = v_x(k) + 0.5 \cdot h_k \cdot (\dot{v}_x(k) + \dot{v}_x(k+1)) \quad (3.27)$$

$$v_z(k+1) = v_z(k) + 0.5 \cdot h_k \cdot (\dot{v}_z(k) + \dot{v}_z(k+1)) \quad (3.28)$$

$$\theta(k+1) = \theta(k) + 0.5 \cdot h_k \cdot (\dot{\theta}(k) + \dot{\theta}(k+1)) \quad (3.29)$$

$$\dot{\theta}(k+1) = \dot{\theta}(k) + 0.5 \cdot h_k \cdot (\ddot{\theta}(k) + \ddot{\theta}(k+1)) \quad (3.30)$$

$$\forall k \in \{1, \dots, N-1\}$$

Path constraints:

$$\frac{|\delta_{fin}(k+1) - \delta_{fin}(k)|}{h_k} \leq \omega_{act} \quad \forall k \in \{1, \dots, N-1\} \quad (3.31)$$

$$|\alpha(k)| \leq 10^\circ \quad \forall k \in \{1, \dots, N\} \quad (3.32)$$

Constant bounds:

$$X_{min} \leq r_x(k) \leq X_{max} \quad (3.33)$$

$$Z_{min} \leq r_z(k) \leq Z_{max} \quad (3.34)$$

$$V_{X_{min}} \leq v_x(k) \leq V_{X_{max}} \quad (3.35)$$

$$V_{Z_{min}} \leq v_z(k) \leq V_{Z_{max}} \quad (3.36)$$

$$\Theta_{min} \leq \theta(k) \leq \Theta_{max} \quad (3.37)$$

$$-\dot{\Theta}_{max} \leq \dot{\theta}(k) \leq \dot{\Theta}_{max} \quad (3.38)$$

$$-\delta_{max} \leq \delta(k) \leq \delta_{max} \quad (3.39)$$

$$\forall k \in \{1, \dots, N\}$$

$$t_{min} \leq t_{tot} \leq t_{max} \quad (3.40)$$

Table 3.1: Setup for aerodynamic descent trajectory optimization.

Powered Landing

Objective function

$$\min \quad m_{fuel} = \sum_k 0.5 \cdot h_k \cdot (-\dot{m}(k) - \dot{m}(k+1)), \quad \forall k \in \{1, \dots, N-1\} \quad (3.41)$$

Collocation constraints:

$$r_x(k+1) = r_x(k) + 0.5 \cdot h_k \cdot (\dot{r}_x(k) + \dot{r}_x(k+1)) \quad (3.42)$$

$$r_z(k+1) = r_z(k) + 0.5 \cdot h_k \cdot (\dot{r}_z(k) + \dot{r}_z(k+1)) \quad (3.43)$$

$$v_x(k+1) = v_x(k) + 0.5 \cdot h_k \cdot (\dot{v}_x(k) + \dot{v}_x(k+1)) \quad (3.44)$$

$$v_z(k+1) = v_z(k) + 0.5 \cdot h_k \cdot (\dot{v}_z(k) + \dot{v}_z(k+1)) \quad (3.45)$$

$$\theta(k+1) = \theta(k) + 0.5 \cdot h_k \cdot (\dot{\theta}(k) + \dot{\theta}(k+1)) \quad (3.46)$$

$$\dot{\theta}(k+1) = \dot{\theta}(k) + 0.5 \cdot h_k \cdot (\ddot{\theta}(k) + \ddot{\theta}(k+1)) \quad (3.47)$$

$$m(k+1) = m(k) + 0.5 \cdot h_k \cdot (\dot{m}(k) + \dot{m}(k+1)) \quad (3.48)$$

$$\forall k \in \{1, \dots, N-1\}$$

Path constraints:

$$\frac{|\beta_{TVC}(k+1) - \beta_{TVC}(k)|}{h_k} \leq \Omega_{\beta,act} \quad \forall k \in \{1, \dots, N-1\} \quad (3.49)$$

$$\frac{|T_{\%}(k+1) - T_{\%}(k)|}{h_k} \leq \dot{T}_{\%} \quad \forall k \in \{1, \dots, N-1\} \quad (3.50)$$

$$|\alpha(k)| \leq 10^\circ \quad \forall k \in \{1, \dots, N\} \quad (3.51)$$

Constant bounds:

$$X_{min} \leq r_x(k) \leq X_{max} \quad (3.52)$$

$$Z_{min} \leq r_z(k) \leq Z_{max} \quad (3.53)$$

$$V_{Xmin} \leq v_x(k) \leq V_{Xmax} \quad (3.54)$$

$$V_{Zmin} \leq v_z(k) \leq V_{Zmax} \quad (3.55)$$

$$\Theta_{min} \leq \theta(k) \leq \Theta_{max} \quad (3.56)$$

$$-\dot{\Theta}_{max} \leq \dot{\theta}(k) \leq \dot{\Theta}_{max} \quad (3.57)$$

$$(3.58)$$

$$-\beta_{TVC_{max}} \leq \beta_{TVC}(k) \leq \beta_{TVC_{max}} \quad (3.59)$$

$$T_{\%min} \leq T_{\%}(k) \leq T_{\%max} \quad (3.60)$$

$$(3.61)$$

$$\forall k \in \{1, \dots, N\}$$

$$t_{min} \leq t_{tot} \leq t_{max} \quad (3.62)$$

Table 3.2: Setup for powered landing trajectory optimization.

Chapter 4

Software Tool Development

This chapter presents the software tool developed in **MATLAB** to model the VLV and optimize its trajectory for vertical landing solving the OCP. In particular, it describes the structure of the tool, the choices and methodologies adopted to write the codes, how they interface, and the operating algorithm.

The software tool is written entirely in **MATLAB** code and it is saved in the **VerticalLandingTrajOpt** folder. The structure of the tool mirrors the structure adopted to model the problem of vertical landing of a first stage of a launcher, as described and explored in detail in the previous chapters. Therefore, the software tool consists of two main modules that deal, respectively, with the modeling and optimization of the trajectory in the two distinct phases of flight: the first phase in which the vehicle is aerodynamically controlled as it approaches the landing target and the second in which it is controlled by the main propulsion system to achieve vertical landing on the landing target. Each of the two modules consists of several **.m** and **.mat** files, which are illustrated below and saved separately in the **AerodynamicDescent** and **PoweredLanding** folders. Both software modules are structured in the same way given the similarity in the modeling of the two phases of the problem, so the files that constitute them have the same nomenclature, except for differences in the code due to the implementation of problems and procedures that vary according to the phase of the trajectory being analyzed. The ways in which the modules and files interface with each others and with the user and how the codes are built are described in the following subsections, going through each file and describing their main features.

Each module has a main submodule called **Optimization Submodule** which contains the files that allow to run trajectory optimization using previous results and outputs obtained from the other **MATLAB** files and functions in the other submodules; thus the **Optimization Submodule** will be the last one to be presented

in the subsections below.

4.1 Aerodynamics Submodule

- **AeroCyl.m**

This function computes the aerodynamics features of the vehicle cylindrical body w.r.t. the wind axis using the equations illustrated in Chapter 2.

Inputs: Mach number, Altitude, Angle of attack, Cylinder Geometry data of the vehicle.

Outputs: Lift and Drag coefficients of the cylindrical body of the vehicle, Center of pressure longitudinal coordinate w.r.t. the base of the cylinder.

- **AeroFin.m**

This function computes the aerodynamics features of the vehicle single fin w.r.t. the wind axis using the equations illustrated in Chapter 2.

Inputs: Fin Geometry data, Angle of attack, Fin position longitudinal coordinate w.r.t. the base of the cylinder.

Outputs: Lift and Drag coefficients of a single fin of the vehicle, Center of pressure longitudinal coordinate w.r.t. the base of the cylinder.

- **AeroAnalysis.m**

This file allows to study the aerodynamics of a specific cylindrical body and of a specific single fins implementing the equations described in Chapter 2.

The first part of the code can be especially modified with data of the cylindrical body one wants to study. First, an evaluation of Reynolds number on the cylindrical body can be run for a range of altitude and Mach number in order to establish whether the flow is laminar or turbulent. Then aerodynamic coefficients can be computed and plotted. The second part of the code needs geometry data for the specific fin to be studied computing its aerodynamic coefficients and plot.

- **TrimAnalysis.m**

This file allows to evaluate the range of angle deflection requested to fins to trim the vehicle at different angle of attacks as described in Chapter 2.

The code can be especially modified to set the flight conditions - Mach number and altitude- in order to have deflection angle trim range for different conditions. Once set the flight conditions, the code solves a non linear equation to compute fin deflection angle that gives zero pitch moment around the center of mass of the vehicle for different angles of attack.

4.2 Dynamics Submodule

- **DynAero.m** and **DynPwd.m**

These functions compute the system dynamics of the vehicle: **DynAero.m** is used when the vehicle is controlled by the aerodynamic resultant produced by the fins deflection while **DynPwd.m** is used when the vehicle is controlled by regulating the thrust magnitude and its direction during powered landing. These functions shall be used during optimization routine to compute discrete system dynamics in each collocation point. Each function can be also be implemented in algorithms using MATLAB Symbolic Toolbox. System dynamics is computed using equations of motions in Chapter 2.

Inputs: Data vector, Control value.

Data vector collects the system design parameters which are fixed and given as inputs of the entire Trajectory Optimization problem.

Control value is the value of fin deflection, thrust magnitude or thrust direction computed for each single collocation point given by optimizer software during optimization routine.

Outputs: System dynamics.

- **DynAeroVariableControl.m** and **DynPwdVariableControl.m**

These functions have been coded as the previous system dynamics functions

described above but they shall be used for trajectory propagation when control time profile is known a priori. The system dynamics output of these functions is then given as input to the `MATLAB` function `ode45` with the function handle for time and state vector and the timespan vector and the initial state vector to propagate the trajectory.

4.3 Optimization Submodule

Chapter 3 illustrates the theory behind the formulation of an OCP applied to a vehicle's trajectory and how this problem is modeled to be solved using numerical methods combined with computational algorithms that exploit transcription methods such as the *Direct Collocation* method. This leads to the definition of an NLP, i.e. a constrained parametric non-linear optimization problem. The files that make up this submodule contain the most important codes of the entire software tool, as they allow the NLP defined for the optimization of the descent and landing trajectory of the VLV to be solved. The strategy and methodology used to write the necessary codes to solve the OCP depended on the initial choice of implementing in the tool either a complete software package capable of directly solving the optimization problem by transcribing the continuous problem into a discrete problem, or integrating one that would only solve the NLP obtained after the transcription had been performed independently by the tool. The choice fell on the latter. In this way, the tool can be integrated with different transcription methods so as to be as open and adaptable as possible.

The software chosen to solve the NLP is `sn0pt`. SNOPT is a general-purpose system for constrained optimization. It minimizes a linear or nonlinear function subject to bounds and sparse linear or nonlinear constraints on the variables. It is suitable for large-scale general nonlinear programs [26] like the one faced in this thesis, of the form

$$\min_x f_0(x) \quad \text{subject to} \quad l \leq \begin{pmatrix} x \\ f(x) \\ A_L x \end{pmatrix} \leq u \quad (4.1)$$

where x is the vector of the optimization variables the solver has to compute in order to have the minimum value of the smooth scalar objective function $f_0(x)$ without violating the vector of smooth non-linear constraints functions $\{f_i(x)\}$, the sparse matrix A_L of linear constraints and the lower and upper bounds fixed by scalar vectors l and u . SNOPT has several interfaces that allow to formulate problems in different formats. The interface A allows one to define in one user routine the non-linear constraints functions and objective functions and to enter

arbitrary the linear and non-linear variables and functions. The B and C interfaces are better from an efficiency point of view, but the `snOptA` interface is simpler to integrate into the code and it makes it easier to modify the tool coding with a view of future developments and improvements [27]. Below, it is shown the code where SNOPT interface A is called to launch the optimization routine and a description of the main data structures is provided.

```

1 [x_opt, F_result] = snopt(x0, xlow, xupp, xmul, xstate, ...
2                           Flow, Fupp, Fmul, Fstate, ...
3                           userfun, ...
4                           ObjAdd, ObjRow, ...
5                           A, G, options);

```

Figure 4.1: SNOPT Call Interface.

- **Vector of the optimization variables x**

This vector is made by the concatenation of the following data structures.

- The i -th state vector

$$\mathbf{x_state(i)} = \begin{bmatrix} r_x(i) \\ r_z(i) \\ v_x(i) \\ v_z(i) \\ \theta(i) \\ \dot{\theta}(i) \\ m_{tot}(i) \end{bmatrix} \quad \forall i \in \{1, \dots, N\} \quad (4.2)$$

where N is the total number of collocation nodes. Note that m_{tot} is a state vector component only for the vertical landing phase of trajectory, during aerodynamic descent it is constant because engines are off.

- The i -th control vector

$$\mathbf{x_control(i)} = \begin{bmatrix} u_k(i) \\ \vdots \\ u_n(i) \end{bmatrix} \quad \forall k \in \{1, \dots, n\}, \forall i \in \{1, \dots, N\} \quad (4.3)$$

where N is the total number of collocation nodes and n is the total number of control system parameters on the i -th collocation nodes. For

aerodynamic descent phase is $n = 1$ (fin deflection angle) and during powered landing phase is $n = 2$ (thrust direction deflection and thrust magnitude regulation).

- The variable

$$t_{control} = t_{tot} \quad (4.4)$$

where t_{tot} is a control parameter for the total duration of the trajectory and for the definition of timespan $\Delta t = \frac{t_{control}}{N}$ of each segment between two consequent collocation nodes.

Finally, the vector of the optimization variables is

$$\mathbf{x} = \begin{bmatrix} x_state(i) \\ \vdots \\ x_state(N) \\ x_control(i) \\ \vdots \\ x_control(N) \\ t_control \end{bmatrix} \quad (4.5)$$

It is observed that the total length of this vector varies basing on which trajectory phase is optimized. The initial guess for the vector of optimization variables is $\mathbf{x0}$. It is built rearranging the result from the trajectory propagation made using guess starting point and control parameters time profile.

xlow, xupp are two distinct vectors that collects, respectively, the lower and upper bounds for the vector of optimization variables $\mathbf{x}$.

- **Vector of objective and constraints functions F**

This vector is made by the concatenation of the following data structures.

- The scalar objective function

$$f_0(x) \quad (4.6)$$

where x is the vector of optimization variables.

- The i -th dynamics constraints vector

$$\mathbf{ceq_dyn(i)} = \begin{bmatrix} r_x(i+1) - r_x(i) - 0.5 \cdot \Delta t \cdot (\dot{r}_x(i) + \dot{r}_x(i+1)) \\ r_z(i+1) - r_z(i) - 0.5 \cdot \Delta t \cdot (\dot{r}_z(i) + \dot{r}_z(i+1)) \\ v_x(i+1) - v_x(i) - 0.5 \cdot \Delta t \cdot (\dot{v}_x(i) + \dot{v}_x(i+1)) \\ v_z(i+1) - v_z(i) - 0.5 \cdot \Delta t \cdot (\dot{v}_z(i) + \dot{v}_z(i+1)) \\ \theta(i+1) - \theta(i) - 0.5 \cdot \Delta t \cdot (\dot{\theta}(i) + \dot{\theta}(i+1)) \\ \dot{\theta}(i+1) - \dot{\theta}(i) - 0.5 \cdot \Delta t \cdot (\ddot{\theta}(i) + \ddot{\theta}(i+1)) \end{bmatrix} \quad (4.7)$$

$$\forall i \in \{1, \dots, N - 1\}$$

Note that for powered landing phase there is an ulterior constraint function defined for m_{tot} . The label *eq* indicates that dynamics constraints are defined as equality constraints. Each dynamics constraint is defined between a couple of consequential collocation nodes.

- The i -th constraints vector on system control parameters

$$\mathbf{c}_{ineq_control_k}(i) = \left[\frac{(u_k(i+1) - u_k(i))}{\Delta t} - C \right] \quad (4.8)$$

$$\forall k \in \{1, \dots, n\}, \forall i \in \{1, \dots, N - 1\}$$

where C is a constant. For each k -th system control parameter, a constraints vector is defined between each couple of consequential collocation nodes. The label *ineq* indicates that dynamics constraints are defined as inequality constraints.

- The i -th constraints vector on general system parameters

$$\mathbf{c}_{ineq_system_j}(i) = \left[\frac{(s_k(i+1) - s_k(i))}{\Delta t} - C \right] \quad (4.9)$$

$$\forall j \in \{1, \dots, S\}, \forall i \in \{1, \dots, N\}$$

where s is a general system parameter for whom a constraint function is defined on each collocation node, S is the total system parameters for whom a constraint function is defined and C is a constant. The label *ineq* indicates that dynamics constraints are defined as inequality constraints. In this thesis work this kind of constraints is defined for the angle of attack.

Finally, the vector of objective and constraints function is

$$\mathbf{F} = \begin{bmatrix} f_0(x) \\ \begin{bmatrix} ceq_dyn(i) \\ \vdots \\ ceq_dyn(N-1) \end{bmatrix} \\ \begin{bmatrix} cineq_control_k(i) \\ \vdots \\ cineq_control_k(N-1) \end{bmatrix} \\ \begin{bmatrix} cineq_control_n(i) \\ \vdots \\ cineq_control_n(N-1) \end{bmatrix} \\ \begin{bmatrix} cineq_system_j(i) \\ \vdots \\ cineq_system_j(N-1) \end{bmatrix} \\ \begin{bmatrix} cineq_system_S(i) \\ \vdots \\ cineq_system_S(N) \end{bmatrix} \end{bmatrix} \quad (4.10)$$

Flow, **Fupp** are two distinct vectors that collect, respectively, the lower and upper bounds for the vector of optimization variables **F**. For equalities constraints, **Flow** and **Fupp** must both be set at 0, while for inequalities constraints, upper bounds is set to 0 and lower is a function of the constant C .

- **userfun**

This function is assigned with the `ObjCons.m` function (presented below) with the function handle for the vector of optimization variables **x**, in order to compute the vector F and the Jacobian matrix of **F** with respect to **x** during optimization routines.

- **G**

This is a structure made of two columns vectors that collect, respectively, the rows and columns indexes of non-zero elements of the Jacobian matrix of **F** with respect to the vector **x**, that is to say the sparsity structure of the Jacobian matrix.

- **xmul,xstate,Fmul,Fstate**

These optional vectors are not provided as inputs to the optimization routine. The *mul* vectors collect an estimation of Lagrangian multipliers,, while the *state* vectors collect a set of initial state for $\mathbf{x}$ and $\mathbf{F}$ that give information about a bound or a constraint being active or not.

- **ObjAdd,ObjRow**

The first one is a scalar that can be add to objective expression if it depends from a fixed value. It is set to 0 in this thesis work. The second one is a scalar to specify the objective function position in $\mathbf{F}$ vector. It is 1 by default so this means that the first value of $\mathbf{F}$ is the objective function value.

- **A**

This is a sparse matrix that can be used to define linear constraints as a sum to the non-linear ones. It is not used in this thesis work.

- **xopt**

The resulting vector from optimization routine. It collects the state vectors and the control parameters of the optimized trajectory.

- **Fresult**

This vector collects the final objective scalar and the constraints functions evaluation once the optimization routine is concluded.

The files MATLAB presented below constitute the Optimization submodule.

- **ProblemStructure.m**

The user should ideally provide SNOPT with all the gradients of vector $\mathbf{F}$ with respect to the vector of optimization variables $\mathbf{x}$ because this makes SNOPT faster and more efficient [28]. This was observed during the first development stages of the software tool when the Jacobian matrix was not provided to SNOPT, resulting in very slow and inefficient optimization routines due to the large scale of the problem. Thus, it resulted necessary to analytically compute the Jacobian matrix instead of letting SNOPT numerically estimate it. To do so, **MATLAB Symbolic Toolbox** was used in order to use computational capabilities to automatize the process. In fact, the entire constraints functions vector $\mathbf{F}$ was written using symbolic variables and then

MATLAB command *jacobian* computed its Jacobian matrix with respect to the vector of optimization variables $\mathbf{x}$. The vector $\mathbf{F}$ and its Jacobian matrix obtained with the **MATLAB Symbolic Toolbox** needed to be converted and saved as a **MATLAB** function in order to be then numerically evaluated but, due to their large scale, this operation required too much time. Time required was in the order of units of hours which was considered too long. The solution to this problem was to leverage the direct collocation method formulation. In particular, two considerations were made:

- Each collocation node is independent from the others. This means that the first derivative of a constraint expression defined on a certain collocation node or on a certain couple of consequential collocation nodes is non-zero only when computed with respect to those specific collocation nodes.
- Constraints of the same type have the same expression independently from which collocation node they are defined on.

This allowed to:

- compute only one symbolic expression for each type of constraint function and only one Jacobian matrix for each of them;
- save two **MATLAB** functions for each type of constraint function: one for its expression and one for their Jacobian matrix;
- build the total symbolic Jacobian matrix in order to save its sparsity structure.

This strategy produces more **MATLAB** functions but significantly smaller, instead of only two but large scaled. This allows to reduce the time to save them to units of minutes for total numbers of collocation points in the range of 300 to 500 collocation nodes.

Note that for the aerodynamic descent phase module, in **ProblemStructure.m** a **MATLAB** function for the objective function is not saved. Instead, it is directly defined in **ObjCons.m**. On the contrary, for the powered landing phase module, the **MATLAB** function for the objective function is provided by **ProblemStructure.m**.

In conclusion, this file produces as output the **MATLAB** functions that need to be numerically evaluated to create $\mathbf{F}$ and its Jacobian matrix and the sparsity matrix, as showed in **ObjCons.m**. It is important to know that this file has to be run only one time once the input design parameters, number of collocation points and constraints formulations are fixed. They can be set at the beginning

of the file. Then different optimization routines can be run several times varying the initial guess and the the bounds of the problem in `Optimizer.m`.

- **ObjCons.m**

This function is recalled during optimization routines by SNOPT and it is evaluated on the current vector of optimization variables $\mathbf{x}$ in order to compute $\mathbf{F}$, that contains the scalar objective function and the vector of constraints functions, and its Jacobian matrix. To do so, MATLAB functions provided by `ProblemStructure.m` are used in `for` routines with specific indexing strategies that allow to build $\mathbf{F}$ respecting the structure described in 4.10 and its Jacobian matrix.

- **Optimizer.m**

This file is the main file of each module and it is articulated in several sections.

- **SECTION 1: Mission data setting**

Having a good initial guess to start the optimization routine for a non-linear programming is very important because it really influences the optimization results. Depending on the initial guess, an optimization problem can be solved identifying the globally optimal solution in the best case or fail in the worst one. Constraints make it all more difficult because without the correct initial guess an optimization problem can result infeasible even if it is correctly defined [29]. Furthermore, SNOPT is a locally optimal solver which means that when it is used to solve non-convex problems like the one of this thesis, it is able to find more than one locally optimal solution depending on the initial guess from whom it starts the optimization routines.

The strategy adopted is to compute different initial guesses in order to launch different optimization routines and analyze different solutions for the same problem. The best initializations for trajectory optimization usually require some problemspecific knowledge, but there are a few general approaches that can be useful. In this way, initialization is more of an art than a science. Considering how the descent and landing problem has been modeled in this thesis, first aerodynamic descent trajectory is optimized and its final condition is fixed as the initial condition for the powered landing trajectory. The initial guesses for aerodynamic descent are computed propagating the vehicles dynamics on the collocation nodes from different initial positions, for the same total time and under the same time profile of the control parameters randomly generated a priori.

It is assumed that the vehicle is flying at a fixed value of $Mach = 0.8$ when it starts the lower atmosphere aerodynamic descent at a maximum altitude of 20 km. The user shall provide:

- * Total number $\mathbf{N}$ of collocation nodes to use.
- * Total number $\mathbf{n}$ of initial guesses to be randomly generated.
- * Fixed target landing coordinates.
- * Range for initial flight conditions and position values.
- * Range for total time propagation and for control parameters values.
- * Fixed set of input design parameters.

The initial guesses for the powered landing are computed propagating the vehicle dynamics on the collocation points from the fixed initial condition under different time profiles of the control parameters generated a priori. The user shall provide:

- * Total number $\mathbf{N}$ of collocation nodes to use.
- * Total number $\mathbf{n}$ of initial guesses to be randomly generated.
- * Fixed initial conditions saved from previous aerodynamic descent optimization.
- * Range for total time propagation and for control parameters values.
- * Fixed set of input design parameters.

– **SECTION 2: Iterative guess propagation and Optimization routine**

In this section two nested **for** routines are coded. For each initial altitude and flight path angle of the vehicle, different initial guesses propagation are ran varying the other parameters such as downrange distance from the landing target and initial state conditions of the vehicle. Then, for each different initial guess, the optimization routine is performed and all the resulting optimal solutions x_{opt} are stored to be subsequently analyzed. In this section SNOPT routine is called as showed in figure 4.1 and user can set SNOPT options and $\mathbf{F}$ and $\mathbf{x}$ lower and upper bounds.

– **SECTION 3: Optimization Result Analysis**

In this section, optimization routines results are analyzed. After examining SNOPT exit conditions in term of feasibility and optimality, the trajectory time profile of vehicle state variables and control parameters can be plot to further analyze it.

– **SECTION 4: Post Processing**

In this section, the best trajectory optimization result selected from section 3 is post processed. Post processing comprehends interpolation of the discrete optimization results in order to have continuous time profiles, according to the interpolation order used to model direct collocation transcription. It also comprehends error analysis due to the accuracy of transcription process as described in 3.

Chapter 5

Software Tool Testing

This chapter describes the testing of the software tool conducted to verify its operation and potential. First, the choices and procedures followed to define the vehicle model to be used and the trajectory optimization settings are described. It should be noted, as already explained in this document, that the software was developed from scratch and separately from any mission design program. Therefore, every quantitative or qualitative choice regarding the vehicle or mission scenario was made based on knowledge derived by the literature referred to the missions already completed by operational vehicles such as Falcon 9.

5.1 Case study

In the context of the new Space Economy, the race to develop reusable vehicles has led to numerous research projects in this field. Among these, one that stands out is the project launched in 2016 by German Aerospace Centre DLR called ENTRAIN (European Next Reusable Ariane) which has the aim of investigating recovery methods on a system level for two-stage-to-orbit launch vehicles with a reusable winged or no-winged first stage and an expendable upper stage to be operated within a European context. One of the outputs produced during this research project is a comparative analysis of different first stages of launchers developed using MDO (Multidisciplinary Optimization Process) with high-level mission and design requirements and parameters as input [30]:

- 7000 kg + 500 kg margin payload to Geostationary Transfer Orbit (GTO) of $250 \text{ km} \times 35786 \text{ km} \times 6^\circ$ (standard Ariane 5 GTO) via a LEO parking orbit of $140 \text{ km} \times 330 \text{ km} \times 6^\circ$;
- Launch from Centre Spatial Guyanais (CSG), Kourou;
- Two Stage to Orbit (TSTO) configurations;

- Same propellant combination in both stages;
- Same engines in both stages with exception of different nozzle expansion ratios.

With these inputs, various solutions were examined, as shown in Figures 5.1, obtained by combining various design degrees of freedom such as the propellant used, the type of engine cycle (staged combustion or gas generator), and the Δv deliverable by the second stage. Each of these solutions is valid depending on the

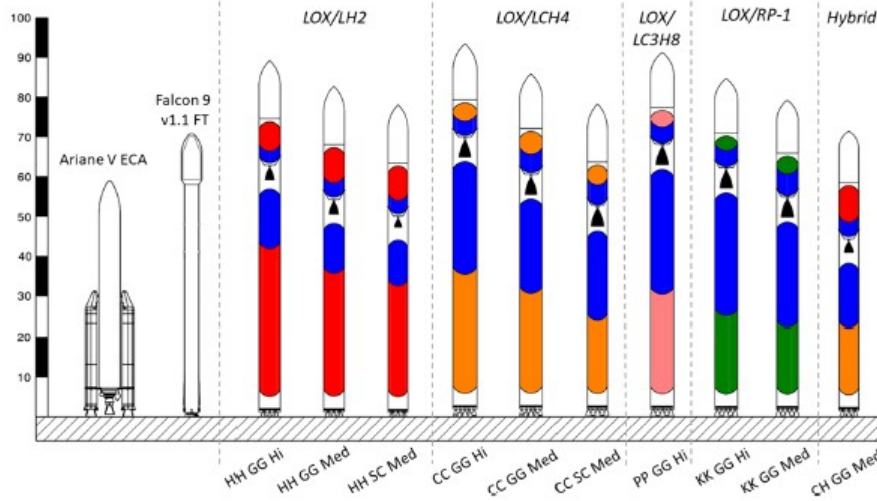


Figure 5.1: Sketches of investigated reusable first stages. See [30]

specific criterion used to evaluate it. In particular, launchers that use hydrogen as fuel have proven to be better in terms of performance, such as specific impulse, while those using hydrocarbons have higher thrust-to-weight ratios [31]. To test the trajectory optimization software tool, the first of the models shown in Figures 5.1 was chosen. Table 5.1 lists all the high-level design parameters used to define the specific model of the first stage analyzed and provided as input to the software.

In parallel with defining the input design parameters, it was also necessary to define mission requirements that identified the mission scenario for aerodynamic descent and powered landing. The mission requirements used are listed below:

- **LANDING TARGET**

The landing target has been set on a vertical landing platform, the ECEF coordinates of which are provided to the software and shown in table 5.2.

- **INITIAL FLIGHT CONDITIONS**

Constant fixed values for Mach and flight path angle are defined to compute initial velocity:

$$Mach_0 = 0.8, FPA_0 = -50^\circ$$

Table 5.1: Design parameters input set for software tool testing

Design parameters input set	
$M_0 = 59,875 \cdot 10^3 \text{ Kg}$	$M_d + M_{prop}$, total initial mass
$M_d = 47,9 \cdot 10^3 \text{ Kg}$	First stage dry mass
$M_{prop} = 11,975 \cdot 10^3 \text{ Kg}$	Initial propellant mass, 20% of M_0
$L = 60 \text{ m}$	First stage length
$d = 5.4 \text{ m}$	First stage diameter
$c_{fin} = 3 \text{ m}$	Fin aerodynamic mean chord
$b_{fin} = 5 \text{ m}$	Fin span
$X_{LE_{fin}} = 45 \text{ m}$	Fin leading edge longitudinal position w.r.t the base, at 3/4 of L
$CoG = 15 \text{ m}$	Centre of gravity longitudinal position w.r.t. the base, at 1/4 of L
$A_e = 2.06 \text{ m}^2$	Exit nozzle area for expansion ratio at s.l.
$I_{sp} = 405.4 \text{ s}$	Single engine specific impulse in vacuum
$T_{sl} = 782 \text{ kN}$	Single engine thrust at sea level

Table 5.2: Landing target coordinates in ECEF

Landing Target coordinates	
-25.179189 deg	Longitude
36.994582 deg	Latitude
72 m	Altitude

$$V_{X_0} = M_0 a_{snd} \cdot \cos(FPA_0), V_{Z_0} = M_0 a_{snd} \cdot \sin(FPA_0)$$

- **INITIAL POSITION FOR AD PHASE**

The exact position in terms of cross-range distance and altitude is not known a priori so a range is defined for both of them, assuming that the vehicle at the end of the re-entry mission in high atmosphere is behind the landing target and moving forward:

$$-9 \leq X_0 \leq -3 \quad Km$$

$$10 \leq Z_0 \leq 20 \quad Km$$

- **FINAL CONDITIONS FOR PL PHASE**

The primary mission requirement is the one relative to the final state vector at landing that has to guarantee a specific attitude of the vehicle, in a specific position (position is referred to the point mass vehicle in ENU reference frame) and with a specific velocity:

$$X_{Land} = Z_{Land} = 0 \quad m$$

$$V_{X_{Land}} = V_{Z_{Land}} = 0 \quad m/s$$

$$\theta_{Land} = -\frac{\pi}{2}$$

$$\dot{\theta}_{Land} = 0 \quad rad/s$$

It is important to observe that in order to not over-constraint the system making difficult for the optimization solver to find a solution, a little tolerance margin is used for tolerances in the order of the unit of meter and meter per second for position and velocity and in the order of unit of degree for attitude.

- **FINAL CONDITIONS FOR AD PHASE**

Knowing the final state of the state vector and having set the design parameters for the vehicle model, tuning was performed to identify the range of position and velocity in which the vehicle must be at the beginning of the powered landing phase in order to be able to reach the final conditions. Considering that the initial conditions of the vehicle at the beginning of the powered landing and the final conditions at the end of the aerodynamic descent must coincide, this range was used to define the constant bounds of the vehicle state variables on the last collocation node of the aerodynamic descent phase. The critical state variables to be controlled at the final node of aerodynamic descent are cross-range distance, altitude and flight path angle. However, vertical velocity tends to always reach its terminal value around 300 m/s during aerodynamic descent and since it was observed to be always manageable by the successive

powered breaking, the flight path angle bounds are instead formulated as horizontal velocity bounds.

$$-300 \leq X_f \leq -1000 \quad m$$

$$4000 \leq Z_f \leq 7000 \quad m$$

$$70 \leq V_{X_f} \leq 90 \quad m/s$$

This last consideration on the final conditions for AD phase allowed to fix the cross-range distance to be used in cost function evaluation for aerodynamic descent phase at $X_T = -800 \text{ m}$.

Once the design parameters for modeling the vehicle and mission requirements had been defined, the values necessary for evaluating the constraints were identified in order to set up trajectory optimization as summarized in tables 3.1 and 3.2. In a general context of software application in an iterative design flow, these values must be decided downstream of mission and system requirements derived from previous iterations of the mission and system design process. Alternatively, they can be set downstream of a tuning process that guarantees an evaluation of the constraints such that the optimization problem is defined within a field of feasible solutions. This is the case in this thesis; this method must be adopted, ensuring that the tuning of the constraints is as consistent as possible with the technological capabilities of the real physical system. Table 5.3 contains the values used to define path constraints and constant bounds during the AD phase and PL phase for control systems and table 5.4 contains the values used to define constant bounds for system states during the AD phase and PL phase.

Finally, it is important to make two observations. The first concerns the angle of attack range, which is limited to ± 10 degrees. As already explained in chapter 2, this limitation is due to the restricted validity range of the aerodynamic relationships used. However, in addition to this, the limitation of the permissible range for the angle of attack with respect to re-entry into the upper atmosphere was also chosen with a view to reducing aerodynamic loads, since air density is greater in the lower atmosphere, and finally to avoid stalling on the fins and ensure a stable aerodynamic force that allows control of the vehicle. The second observation is related to the first one and concerns the flight path angle. In fact, no bound or constraint was defined for the flight path angle to control its evolution over time. This was not considered necessary because the flight path angle is calculated as the difference between the pitch angle and the angle of attack, and considering that the angle of attack is restricted to the range ± 10 degrees and that the pitch angle is constrained to tend towards -90 degrees of attitude, the flight path angle also ‘naturally’ follows the same evolution.

Table 5.3: Path constraints and constant bounds definition for control systems

Control systems path constraints	
$\omega_{act} = 20 \text{ deg/s}$	Max fin actuation angular velocity
$\Omega_{\beta,act} = 10 \text{ deg/s}$	Max TVC actuation angular velocity
$\dot{T}_{\%} = 13\% \text{ 1/s}$	Max thrust magnitude regulation rate, 100 kN/s
Control systems constant bounds	
$\delta_{max} = 13.5 \text{ deg}$	Max fin deflection
$\delta_{min} = -13.5 \text{ deg}$	Min fin deflection
$\beta_{TVC_{max}} = 8 \text{ deg}$	Max TVC deflection
$\beta_{TVC_{min}} = -8 \text{ deg}$	Min TVC deflection
$T_{\%_{max}} = 0.4$	Max % of total thrust magnitude
$T_{\%_{min}} = 1$	Min % of total thrust magnitude
$t_{min_{AD}} = 30 \text{ s}$	Min total time for aerodynamic descent trajectory
$t_{max_{AD}} = 100 \text{ s}$	Max total time for aerodynamic descent trajectory
$t_{min_{PL}} = 10 \text{ s}$	Min total time for powered landing trajectory
$t_{max_{PL}} = 60 \text{ s}$	Max total time for powered landing trajectory

5.2 Results

The methodology for finding solutions to the problem of optimizing the trajectory of the first stage of a launch vehicle is described in detail in chapters 3, 4 above and in the previous section, which illustrates the optimization setup for the specific case study. The results presented below constitute two distinct solutions to the same trajectory optimization problem for a first stage of a launch vehicle that performs the final aerodynamic descent and powered landing. Both solutions were found using a number of collocation nodes equal to $N = 300$.

The two proposed solutions are for two slightly different mission scenarios and are designed to meet the need for a trajectory optimization software tool capable of completing the soft vertical landing mission for different initial conditions resulting

Table 5.4: System states constant bounds

System states constant bounds			
AD phase		PL phase	
Min	Max	Min	Max
X_0	0	X_f	20
Z_f	Z_0	0	Z_f
0	V_{X_0}	-20 m/s	V_{X_g}
-500 m/s	V_{Z_0}	V_{Z_f}	0
$-\frac{\pi}{2}$	θ_0	$-\pi$	θ_f
$-5^\circ/\text{s}$	$5^\circ/\text{s}$	$-15^\circ/\text{s}$	$15^\circ/\text{s}$

from the previous phase of re-entry into the upper atmosphere. For this reason, the two solutions presented below refer respectively to a first case in which the vehicle in initial conditions is at a cross-range distance closer to the target and at a lower altitude, and to a second case in which the cross-range distance and altitude are greater. First case is shown in table 5.5, while second case is shown in table 5.6. From the analysis of the results collected and presented in the following two

Table 5.5: Solution A

Time	Pos X	Pos Z	Vel X	Vel Z	Pitch	Pitch rate
t_{AD_0}	-4 Km	13.9 Km	150 m/s	-180 m/s	-60°	0 °/s
t_{AD_f}	-540 m	5.9 Km	70 m/s	-306 m/s	-79°	-4.5 °/s
t_{PL_0}	-540 m	5.9 Km	70 m/s	-306 m/s	-79°	-4.5 °/s
t_{PL_f}	-1.8 m	2 m	0.01 m/s	-10 m/s	-90.5°	0 °/s

subsections, several conclusions can be drawn. As expected, the total time required to complete the aerodynamic descent and perform the vertical landing is 80 s for case A, compared to 95 s required in case B, since in the second case the distances to be covered are greater. In terms of the division of the total mission into the two individual phases, the time required to complete the vertical landing trajectory is similar for both cases and equal to approximately 40 s, while the aerodynamic

Table 5.6: Solution B

Time	Pos X	Pos Z	Vel X	Vel Z	Pitch	Pitch rate
t_{AD_0}	-6.7 Km	20 Km	150 m/s	-180 m/s	-56°	0 °/s
t_{AD_f}	-800 m	4.7 Km	75 m/s	-324 m/s	-68°	-4.9 °/s
t_{PL_0}	-800 m	4.7 Km	75 m/s	-324 m/s	-68°	-4.9 °/s
t_{PL_f}	-2 m	0 m	0.01 m/s	-8 m/s	-89°	0 °/s

descent phase is longer in the second case. The reason for this can be traced back to what was previously mentioned regarding the ability to find solutions for the vertical landing phase only for a certain range of state values at the beginning of the powered landing phase; therefore, it follows that aerodynamic control in the lower atmosphere can be a tool to cope with any dispersions in the state of the vehicle coming from re-entry into the upper atmosphere that would not allow to reach at the end of the aerodynamic phase the conditions required for a successful powered vertical landing.

The second interesting consideration is related to the previous one and concerns the amount of propellant mass needed to complete the vertical landing. This value was used by the optimization software tool as a cost function, therefore the calculated propellant masses are minimized. For solution A, the propellant mass used is 8.1 *tons*, while for solution B it is 8.5 *tons*. As expected, the values are practically equal given the similarity of the initial conditions for both phases of powered landing. Furthermore, these values comply with the upper limit imposed by the initial estimate made for the propellant mass of 11.9 *tons* with a surplus of approximately 30% that can be used as a safety margin. Comparing the results obtained for the aerodynamic and powered landing phases, it can be seen that aerodynamic control is generally able to ensure a more stable evolution of the state variables trajectory, unlike control via TVC and thrust magnitude regulation. This is mainly due to the fact that control during the powered landing phase must be much faster in order to cope with a more demanding task that requires specific final state conditions to be achieved with a reduced margin of maneuver in terms of time and space. Comparing the two overall solutions, however, it appears that solution B has less stable control time profiles with oscillatory characteristics. This suggests that there is room for improvement in solution B, which could be achieved by investigating possible modifications to the design of the control systems themselves to enhance them in mission scenarios where the initial cross-range distance and altitude are greater.

5.2.1 Solution A

The results for solution A are presented below. Figures 5.2 - 5.20 illustrate the **results of trajectory optimization interpolated between collocation nodes**, separately for aerodynamic descent (5.2-5.10) and powered landing (5.11-5.20). Figures 5.21 - 5.25 illustrate the **total trajectory optimization solution** (blue is aerodynamic descent and red is powered landing).

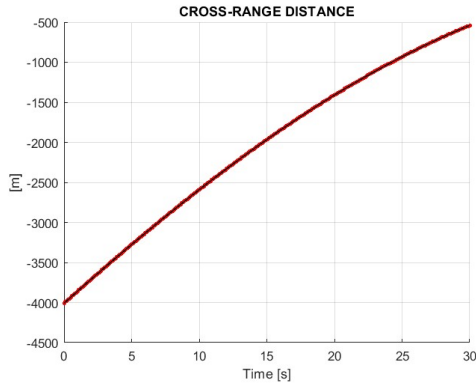


Figure 5.2: Cross-range in AD phase

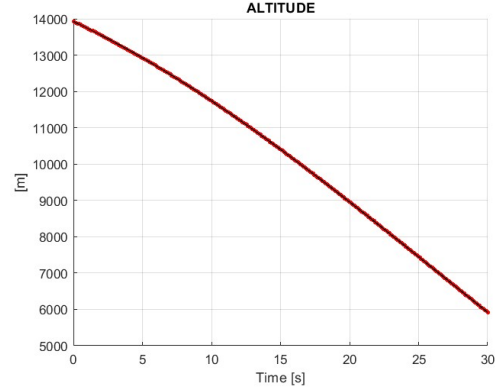


Figure 5.3: Altitude in AD phase

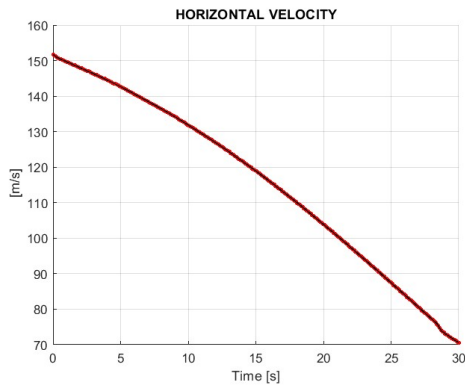


Figure 5.4: Horizontal velocity in AD phase

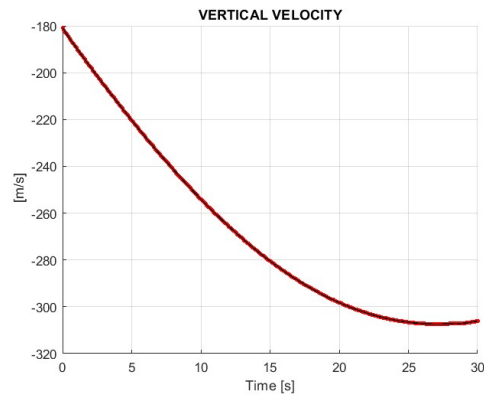


Figure 5.5: Vertical velocity in AD phase

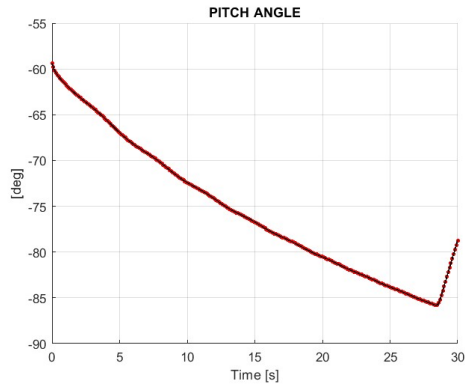


Figure 5.6: Pitch angle in AD phase

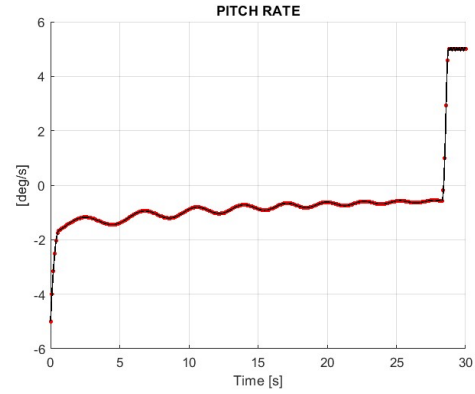


Figure 5.7: Pitch rate in AD phase

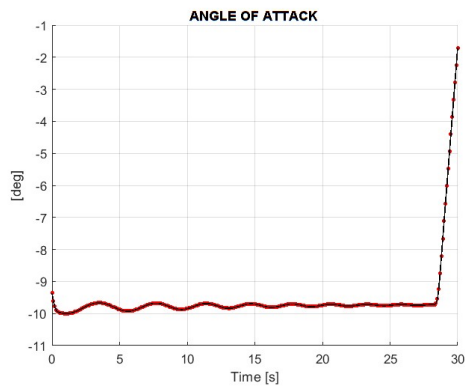


Figure 5.8: Angle of attack in AD phase

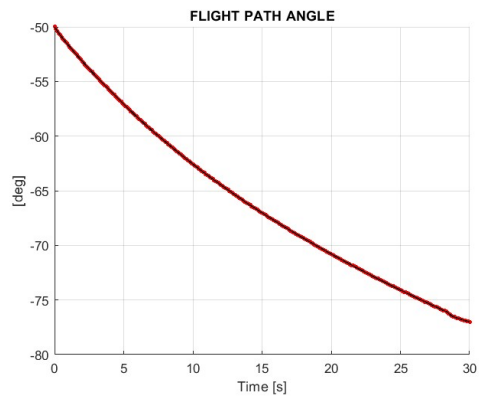


Figure 5.9: Flight path angle in AD phase

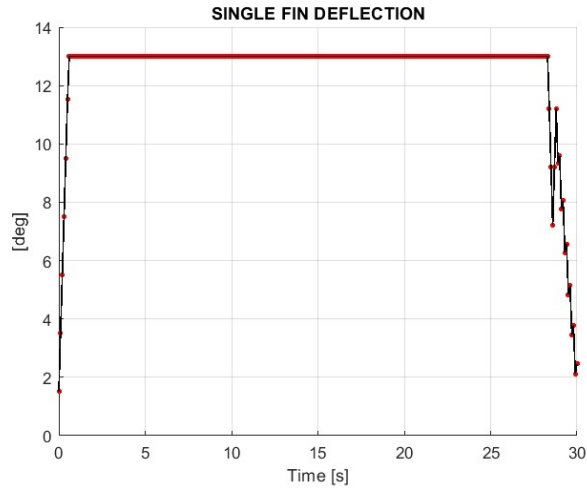


Figure 5.10: Single fin deflection for aerodynamic control

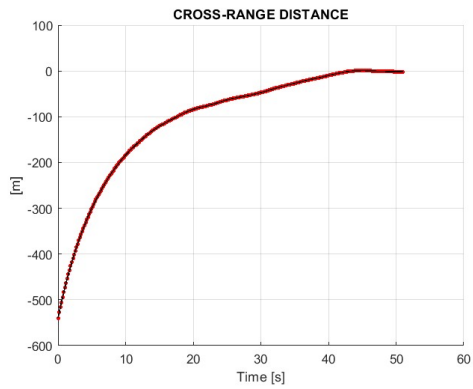


Figure 5.11: Cross-range in PL phase

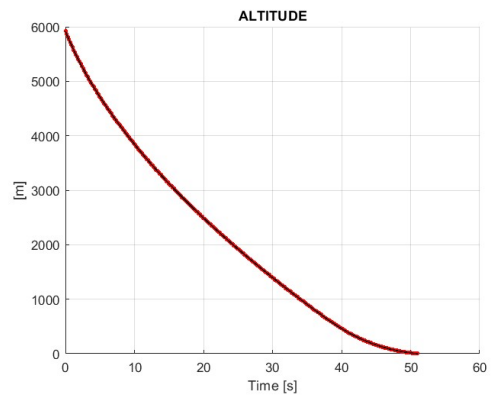


Figure 5.12: Altitude in PL phase

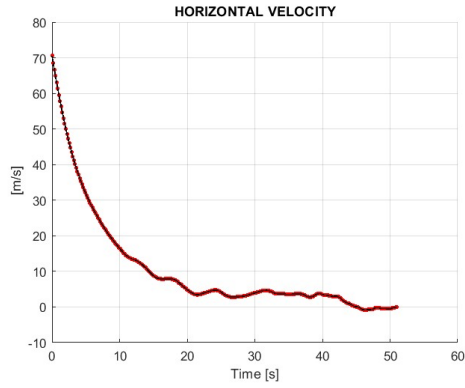


Figure 5.13: Horizontal velocity in PL phase

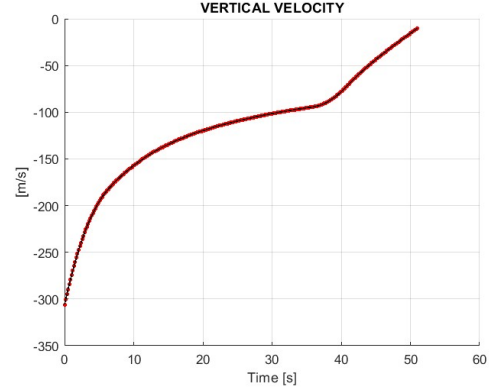


Figure 5.14: Vertical velocity in PL phase

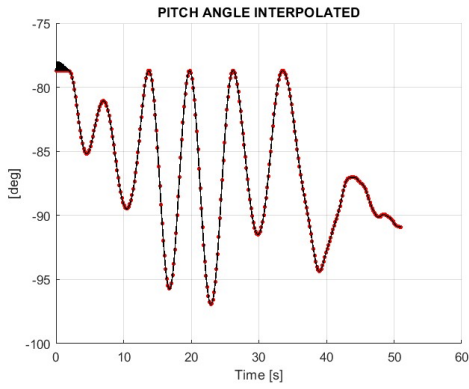


Figure 5.15: Pitch angle in PL phase

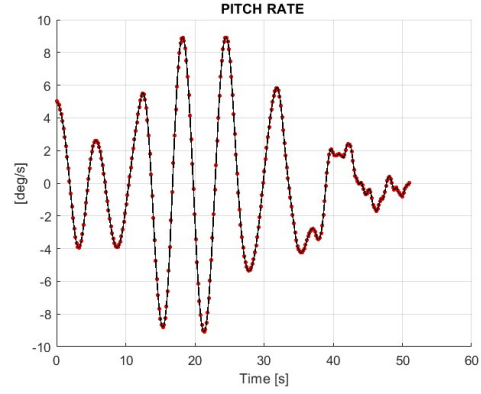


Figure 5.16: Pitch rate in PL phase

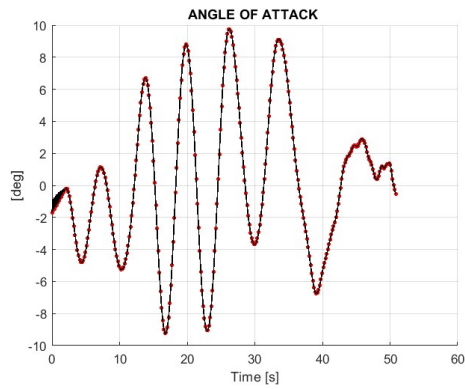


Figure 5.17: Angle of attack in PL phase

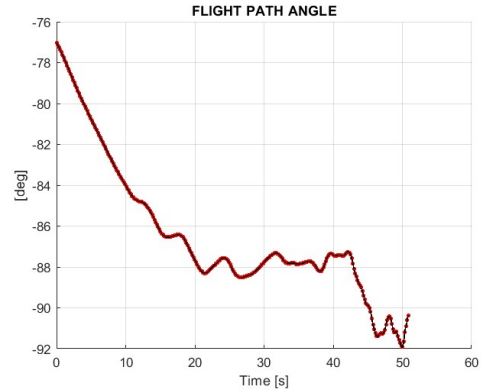


Figure 5.18: Flight path angle in PL phase

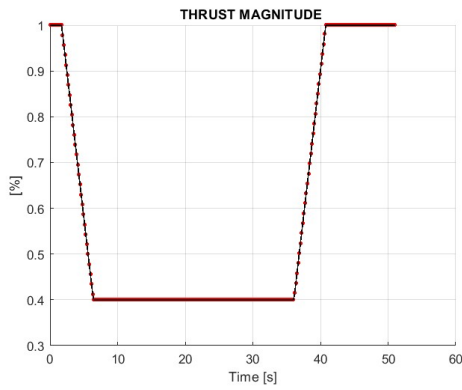


Figure 5.19: Thrust magnitude as % of total thrust

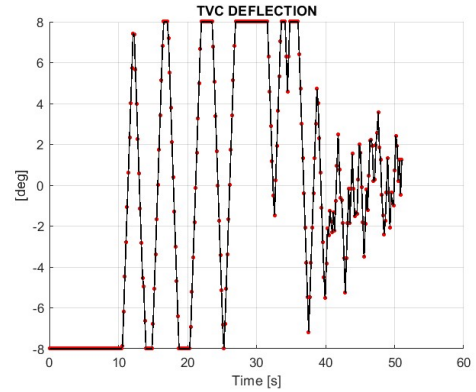


Figure 5.20: Thrust deflection angle by TVC

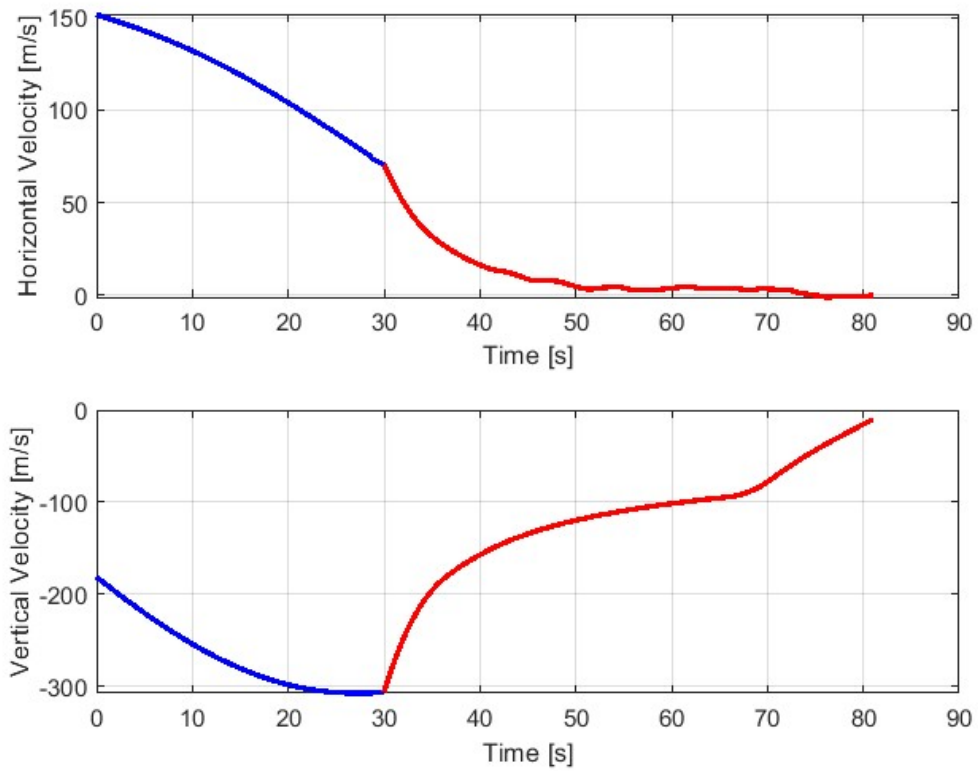


Figure 5.21: Velocity time profile for solution A

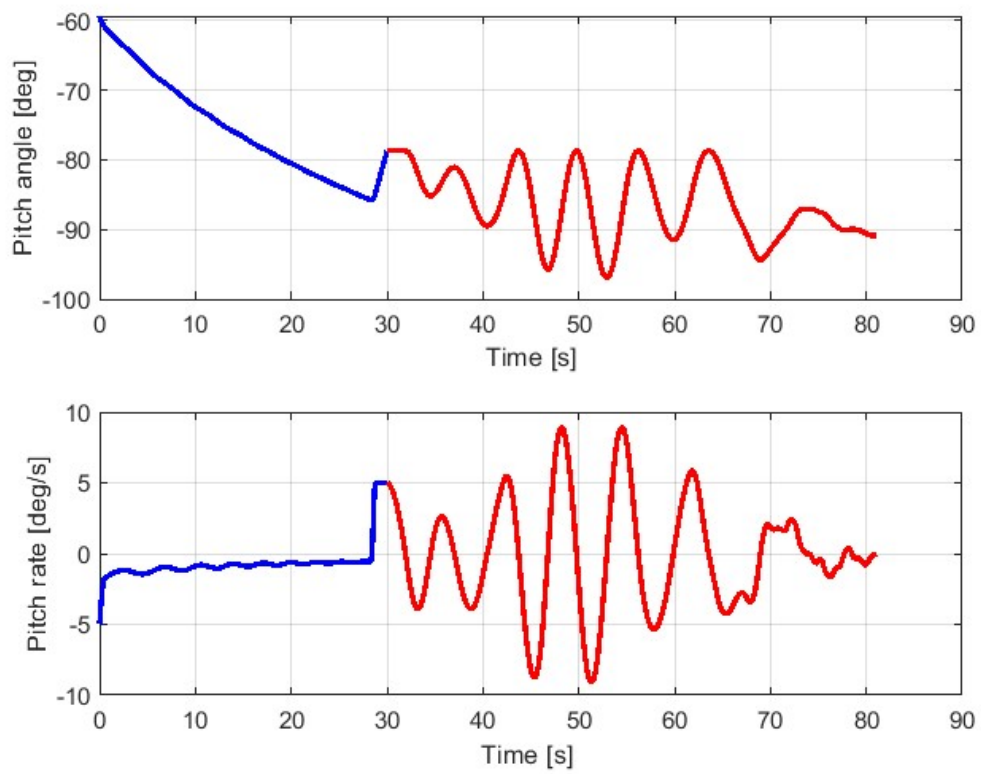


Figure 5.22: Pitch angle and pitch rate time profile for solution A

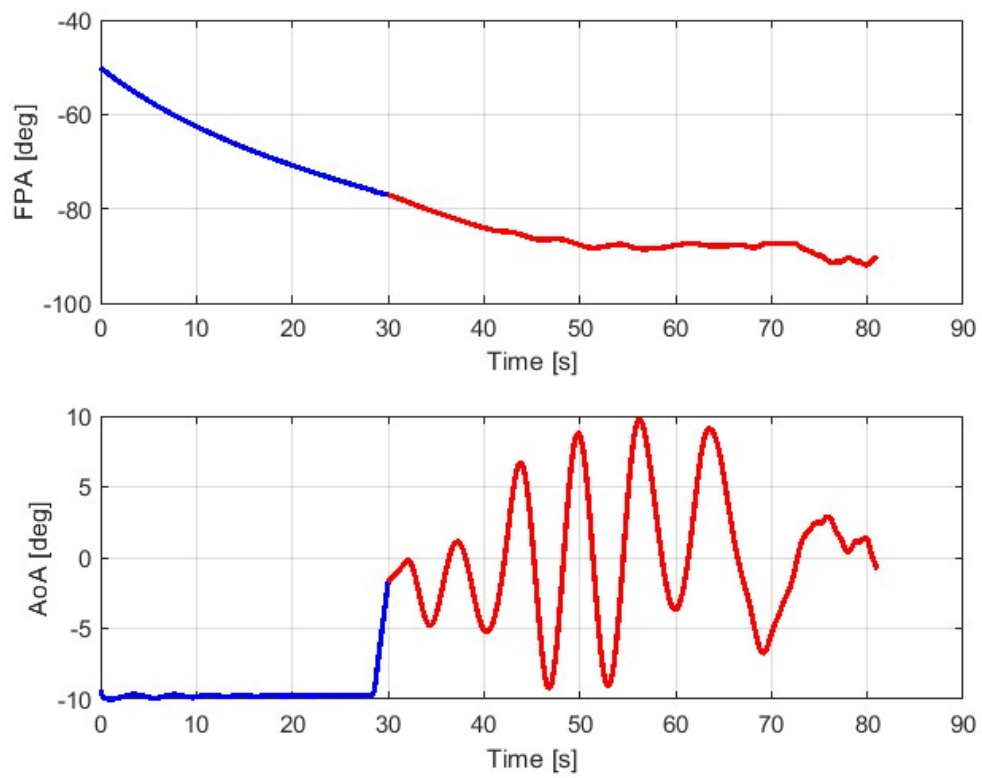


Figure 5.23: AoA and FPA time profile for solution A

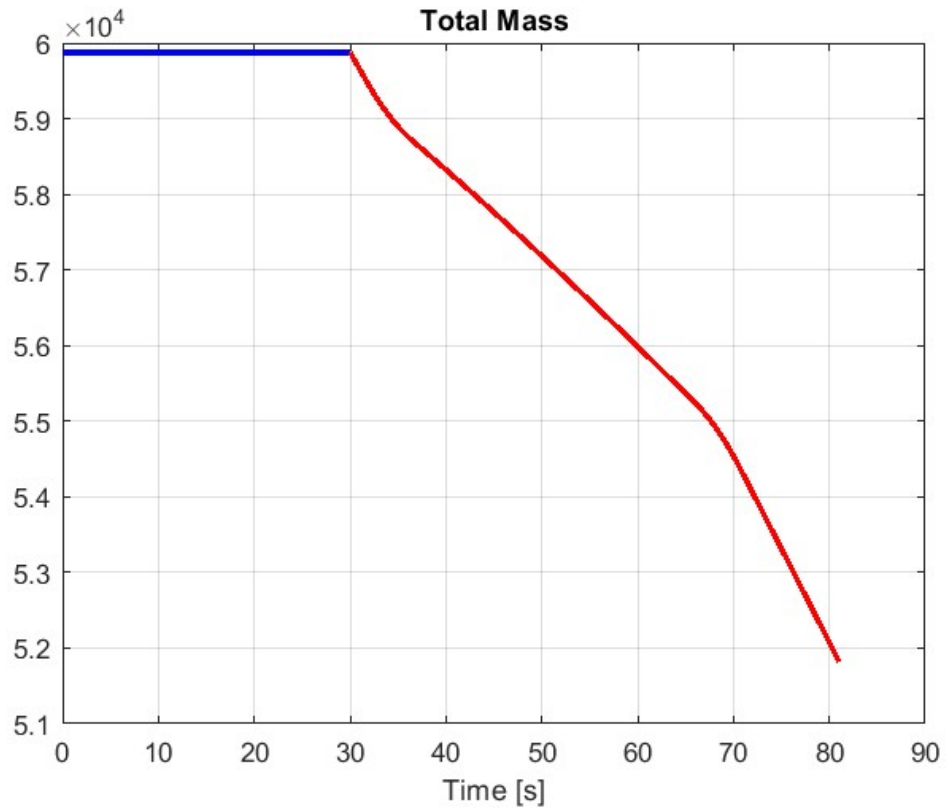


Figure 5.24: Total vehicle mass profile for solution A

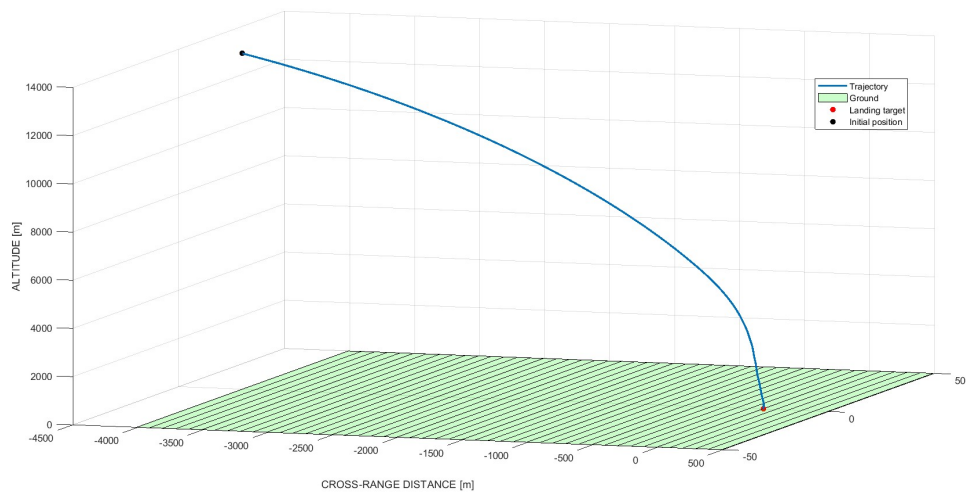


Figure 5.25: Trajectory position visualization for solution A

5.2.2 Solution B

The results for solution B are presented below.

Figures 5.26 - 5.44 illustrate the **results of trajectory optimization interpolated between collocation nodes**, separately for aerodynamic descent (5.26-5.34) and powered landing (5.35-5.44).

Figures 5.45 - 5.49 illustrate the **total trajectory optimization solution** (blue is aerodynamic descent and red is powered landing).

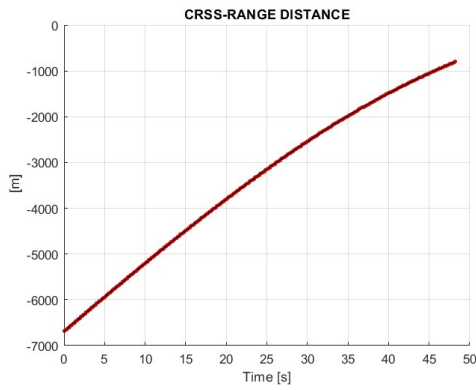


Figure 5.26: Cross-range in AD phase

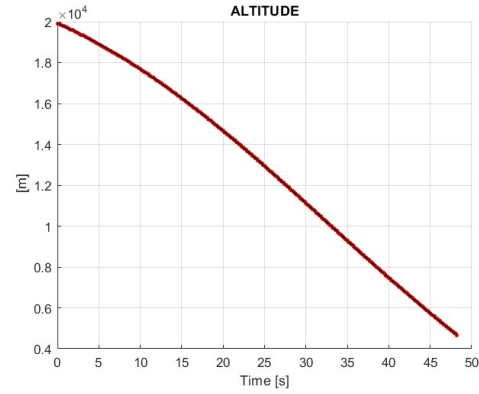


Figure 5.27: Altitude in AD phase

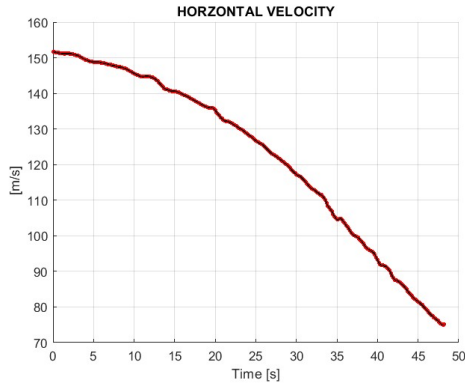


Figure 5.28: Horizontal velocity in AD phase

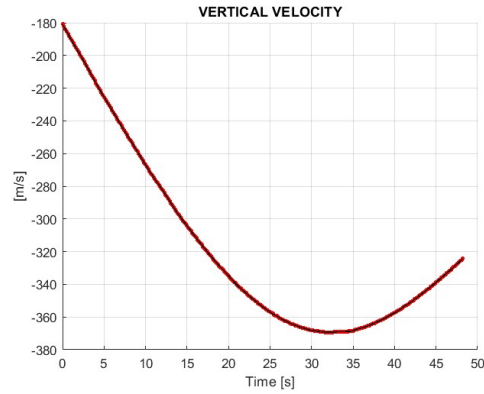


Figure 5.29: Vertical velocity in AD phase

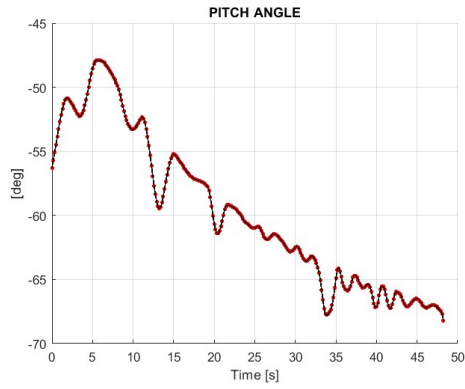


Figure 5.30: Pitch angle in AD phase

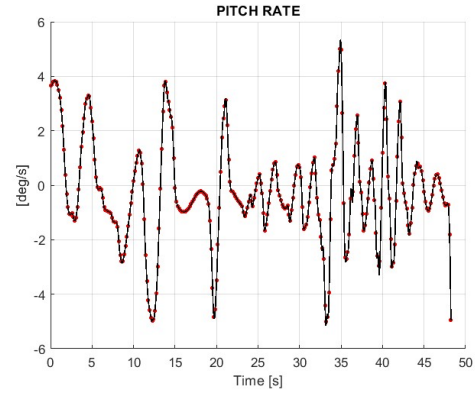


Figure 5.31: Pitch rate in AD phase

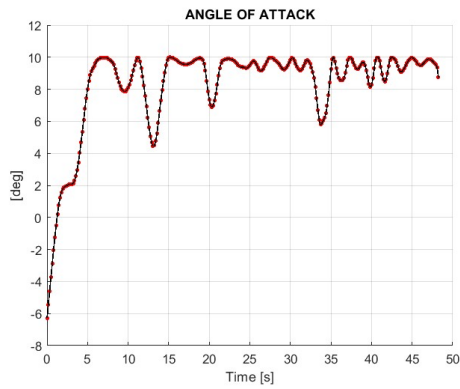


Figure 5.32: Angle of attack in AD phase

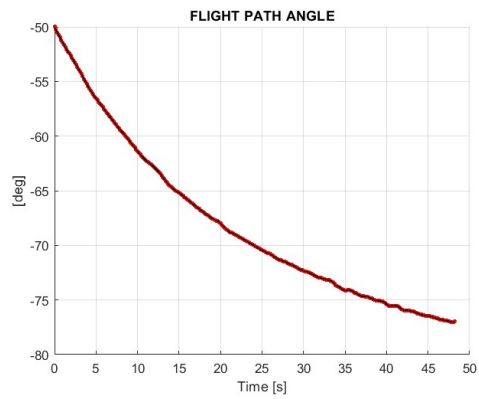


Figure 5.33: Flight path angle in AD phase

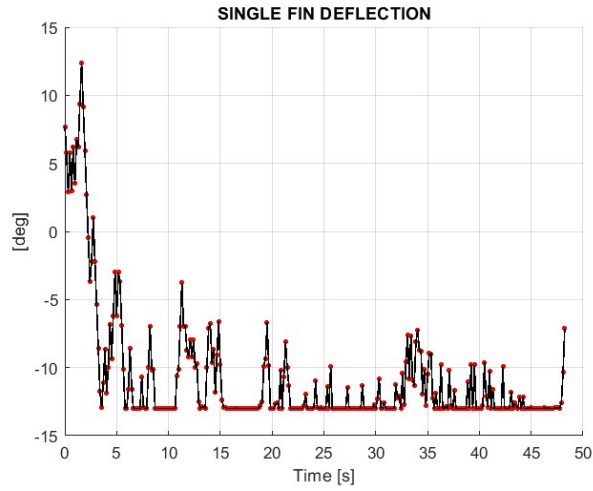


Figure 5.34: Single fin deflection for aerodynamic control

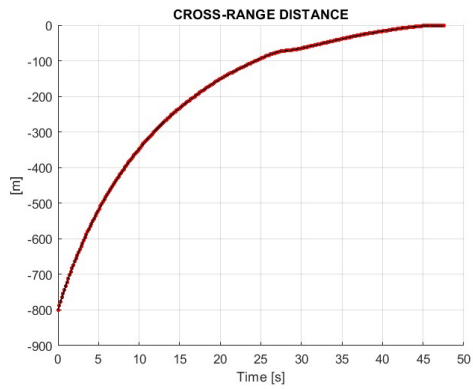


Figure 5.35: Cross-range in PL phase

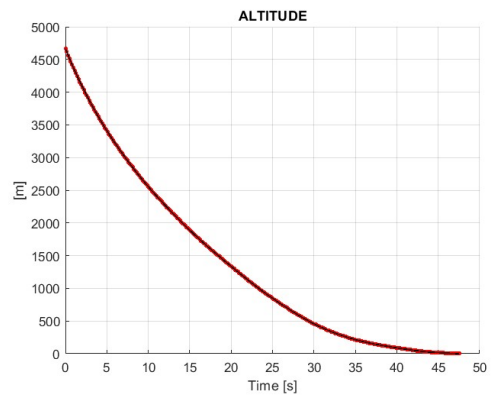


Figure 5.36: Altitude in PL phase

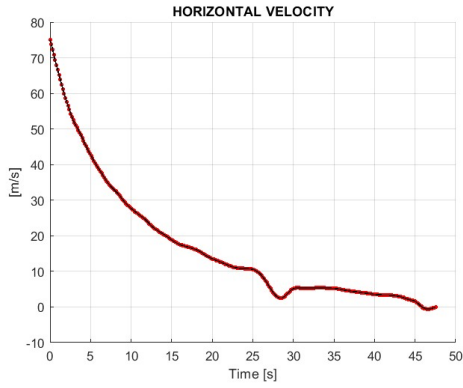


Figure 5.37: Horizontal velocity in PL phase

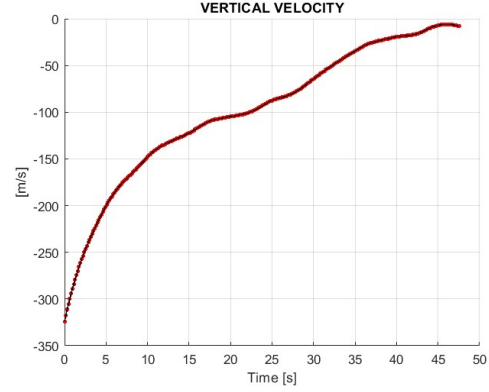


Figure 5.38: Vertical velocity in PL phase

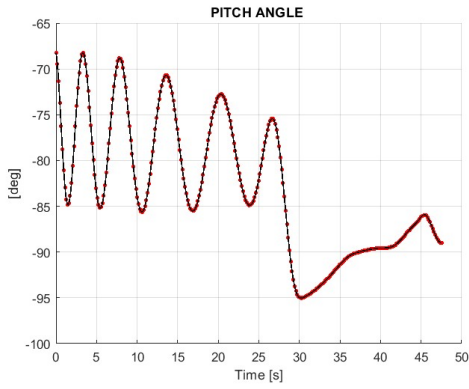


Figure 5.39: Pitch angle in PL phase

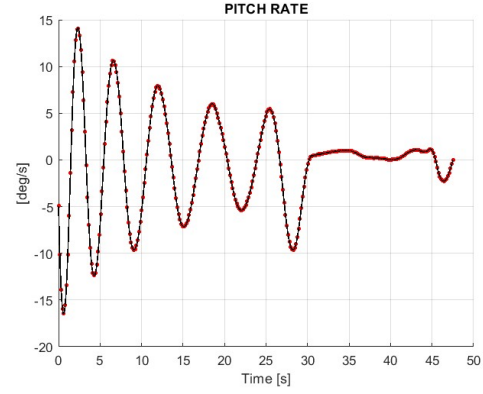


Figure 5.40: Pitch rate in PL phase

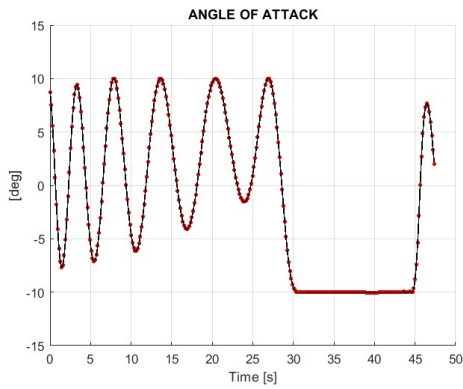


Figure 5.41: Angle of attack in PL phase

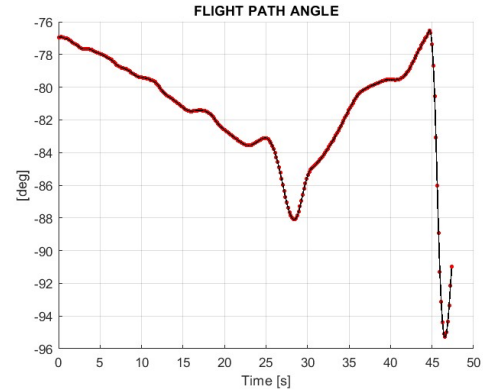


Figure 5.42: Flight path angle in PL phase

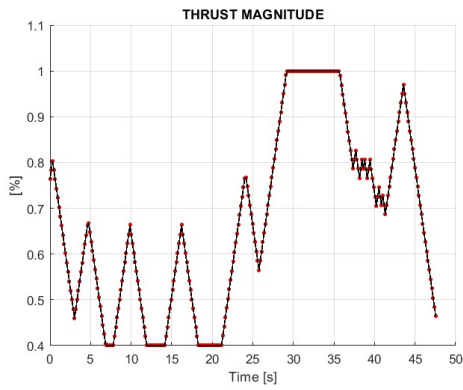


Figure 5.43: Thrust magnitude as % of total thrust

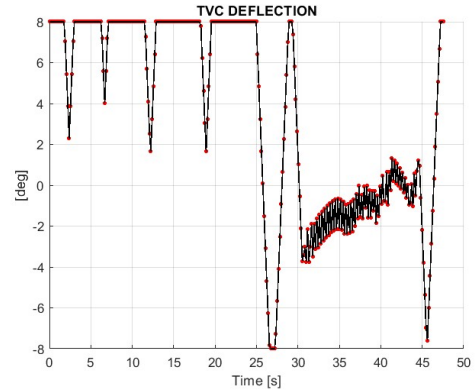


Figure 5.44: Thrust deflection angle by TVC

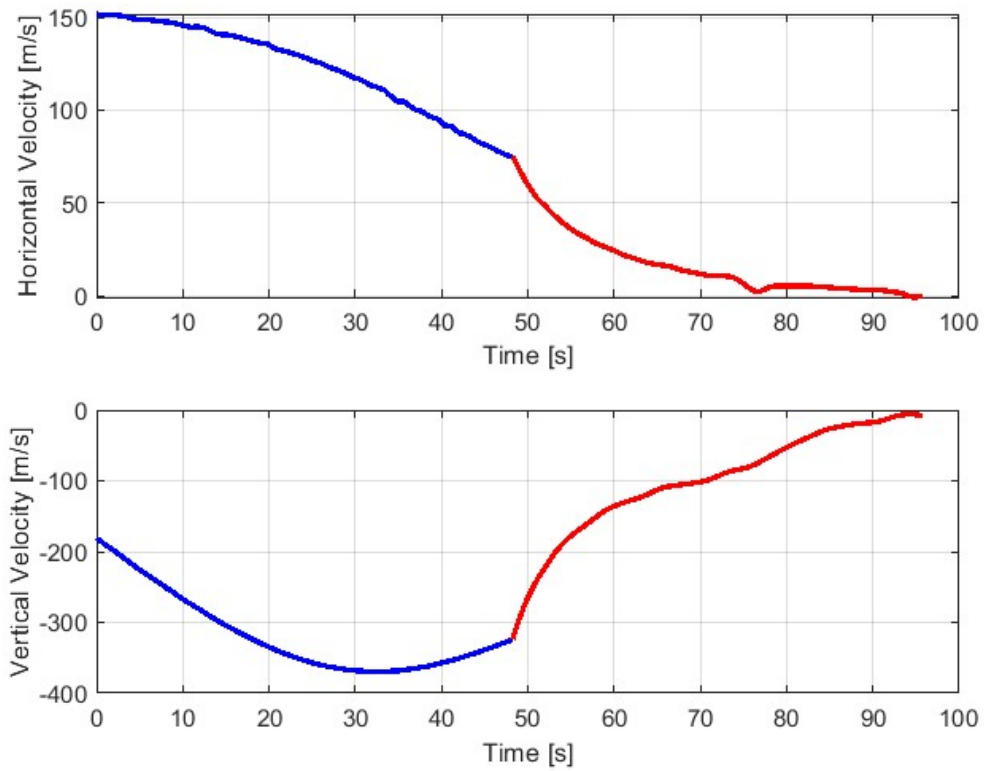


Figure 5.45: Velocity time profile for solution B

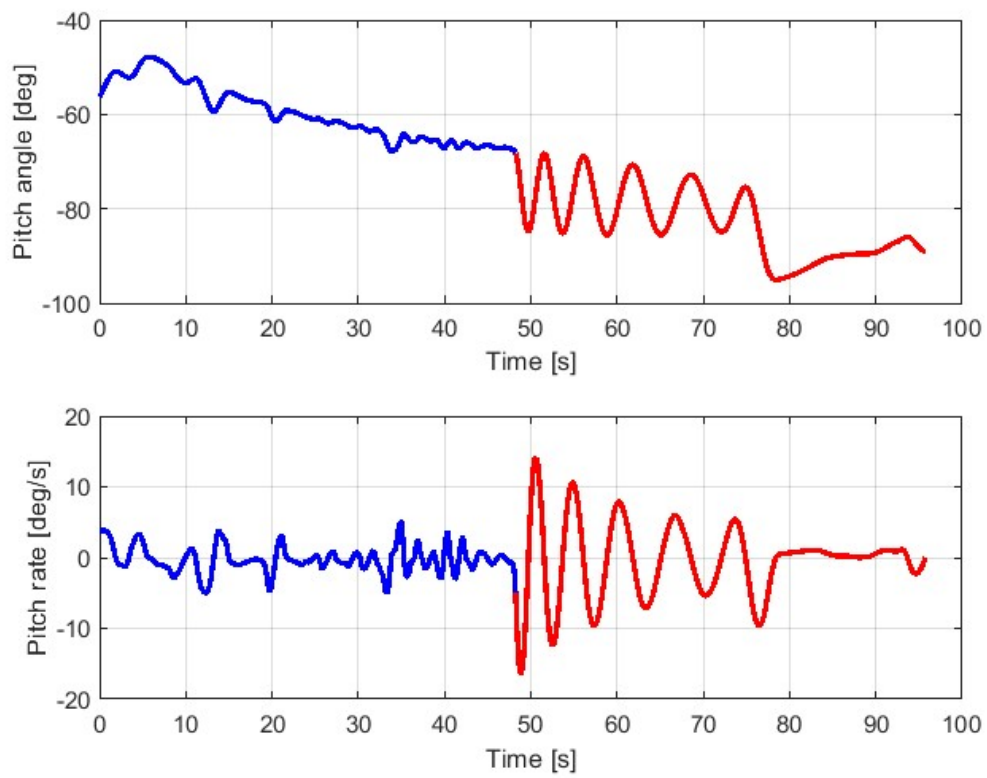


Figure 5.46: Pitch angle and pitch rate time profile for solution B

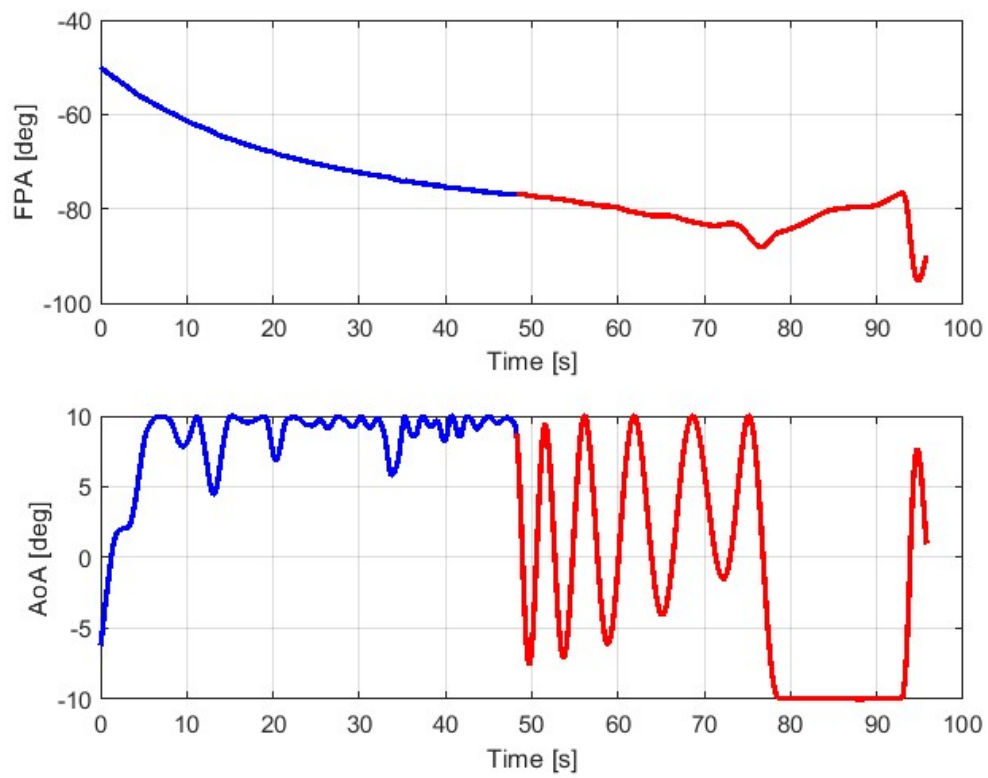


Figure 5.47: AoA and FPA time profile for solution B

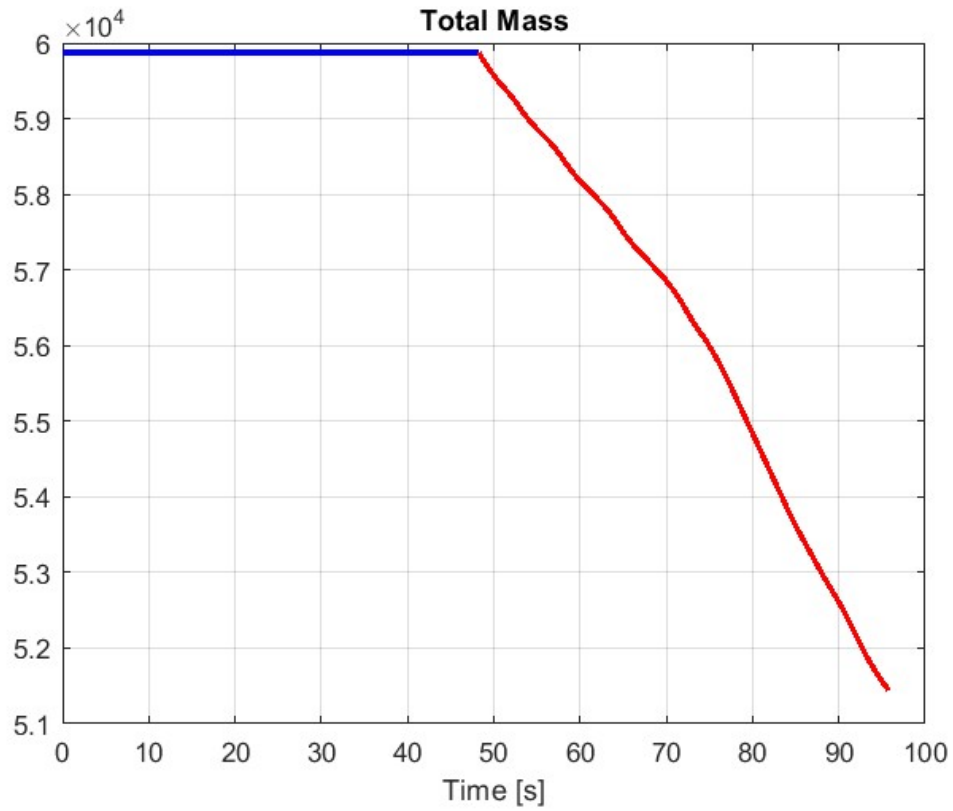


Figure 5.48: Total vehicle mass profile for solution B

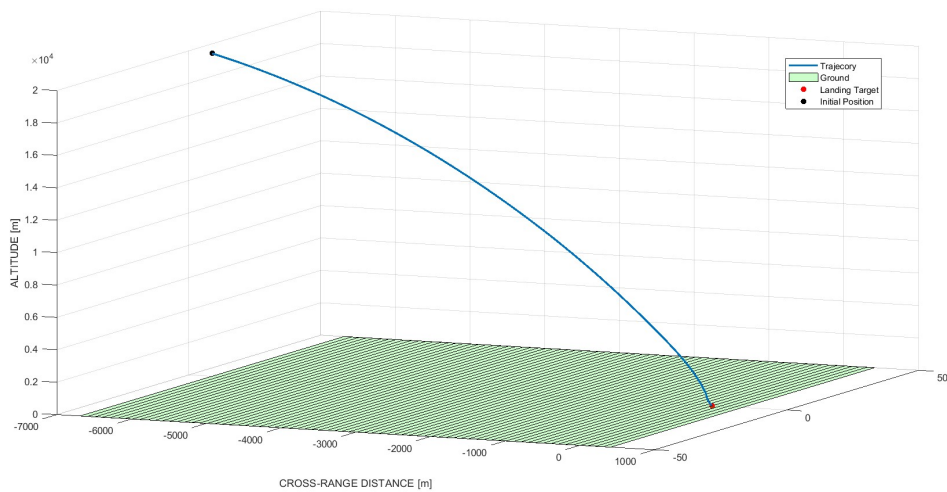


Figure 5.49: Trajectory position visualization for solution B

5.2.3 Error analysis

Error analysis is referred to the error estimation based on how well the candidate trajectory satisfies the system dynamics between the collocation points as already explained in chapter 3. To evaluate this, the absolute value of the difference between the system dynamics interpolated on the collocation nodes and the dynamics evaluated as first derivative of the states interpolated on the collocation nodes was calculated. If this error is small, then it means that the dynamics of the system are well respected between the collocation nodes and therefore the nonlinear program is an accurate representation of the original trajectory optimization problem. Furthermore, the assessment of this error is also an indication of the feasibility of the solution. For brevity, the results of the error evaluation for each state variable are not reported, but only some examples. The first two in Figures 5.50 and 5.51 are relating to horizontal velocity and acceleration, and the last two in Figure 5.52 and 5.53 are relating to the pitch angular rate and acceleration. The error trend for each state variable reflects the examples shown here, confirming an error associated with velocities and accelerations lower than 0.1 m/s and 0.1 m/s^2 , respectively; and for angular pitch rates and accelerations, an error lower than 1 deg/s and 1 deg/s^2 , respectively. The error time profile in figure 5.53 to the powered landing phase of solution B and is proposed here because it is the only case found for all state variables of both solutions provided that slightly exceeds the maximum threshold considered acceptable. However, this is limited to only a few initial moments of the trajectory, so the solution has been retained as valid. The problem can be solved by refining the solution, for example by increasing the number of collocation nodes in that initial time frame to make the NLP more accurate.

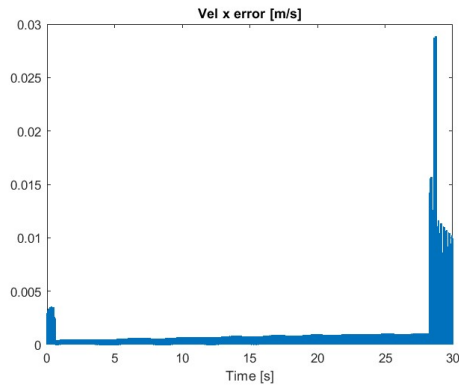


Figure 5.50: Error for horizontal velocity

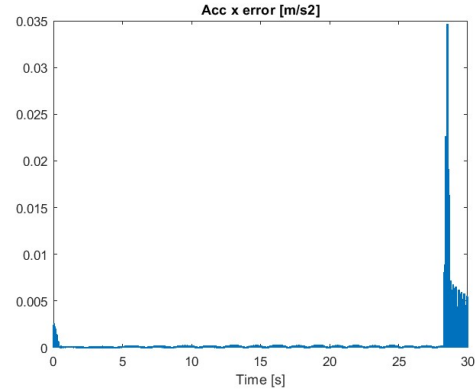


Figure 5.51: Error for horizontal acceleration

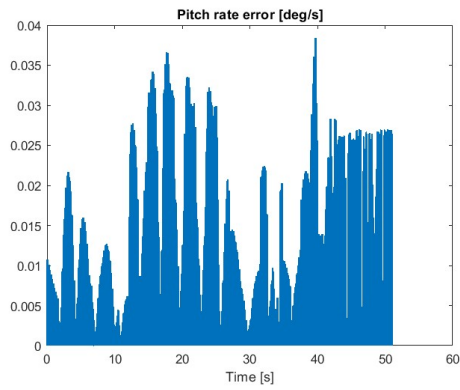


Figure 5.52: Error for pitch angular rate

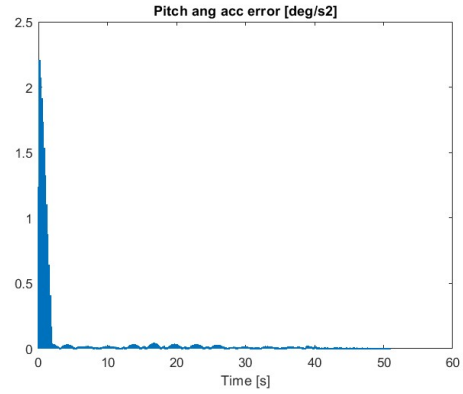


Figure 5.53: Error for pitch acceleration rate

Chapter 6

Conclusion

The primary objective of this thesis, to develop a software tool for optimizing the trajectory of a first stage launch vehicle capable of powered vertical landing, can be considered achieved. To achieve this goal, several sub-goals were completed. These include an analysis of the current state of the art in the field of reusable vehicles and, specifically, strategies for recovering the first stages of a launch vehicle, which then allowed for the preliminary modeling of a vehicle that simulated the first stage of a launch vehicle in order to optimize its trajectory. The software tool developed in **MATLAB** was then efficiently integrated not only with the **SNOPT** solver for optimization but also with functions that allow the user to conduct aerodynamic analyses of the vehicle and propagate its dynamics. The results produced allow to evaluate the performance required of the control systems, fins and TVC, respectively. Furthermore, the results can be used to assess which flight conditions must be achieved in terms of position and speed in order to complete the vertical landing based on the mission scenario defined in the input. Consequently, the trajectory optimized in terms of both control and states systems constitutes a nominal reference that can be used in a possible iterative design cycle as feedback for verifying the system and mission designs provided as input to the optimizer. Future developments of the software may focus on two main areas. The first relates to the engineering modeling of the vehicle, which can be improved by integrating aerodynamic databases, mass models, and geometries with a view to integrating the software to perform mission analysis in a complete and iterative design flow. The other refers to optimization and, in particular, to the optimization algorithm, which can be improved by using more accurate methods such as the Hermite-Simpson direct collocation method or by increasing the number of collocation nodes to refine the grid. These changes shall be made considering that, while the accuracy of the optimization increases, the software tool becomes more time-consuming to use.

Bibliography

- [1] Pavan Daswani et al. *Space: The Dawn of a New Age*. Report. Disponibile su: https://www.citigroup.com/global/insights/space_20220509 (o altro URL stabile). New York: Citi Global Perspectives & Solutions (Citi GPS), May 2022 (cit. on p. 2).
- [2] Paolo Baiocco. «Overview of reusable space systems with a look to technology aspects». In: *Acta Astronautica* 189 (Dec. 2021), pp. 10–25. DOI: 10.1016/j.actaastro.2021.07.039. URL: <https://doi.org/10.1016/j.actaastro.2021.07.039> (cit. on pp. 2, 3, 5).
- [3] Quantum News. «Space Technology & Energy: Reusable Rockets and the New Space Race: Lowering the Barriers to Orbit – A Commercial Space Revolution». In: *Quantum Zeitgeist* (Mar. 2025). URL: <https://quantumzeitgeist.com/space-technology/> (cit. on p. 2).
- [4] B.N. Suresh and K.Sivan. *Integrated Design for Space Transportation System*. Springer, 2015 (cit. on p. 8).
- [5] J. Wilken and S. Stappert. «Comparative analysis of European vertical landing reusable first stage concepts». In: *CEAS Space Journal* 17.1 (2025), pp. 113–130 (cit. on p. 8).
- [6] Tamas Bykerk. «A standard model for the investigation of aerodynamic and aerothermal loads on a re-usable launch vehicle». In: *Aerospace Europe Conference 2023 - 10th EUCASS - 9th CEAS*. German Aerospace Center (DLR). Goettingen, Germany, 2023 (cit. on pp. 10, 11).
- [7] Marco Sagliano, Ansgar Heidecker, José Macés Hernández, Stefano Fari, Markus Schlotterer, Svenja Woicke, David Seelbinder, and Etienne Dumont. «Onboard Guidance for Reusable Rockets: Aerodynamic Descent and Powered Landing». In: (2021) (cit. on p. 11).
- [8] Michael Szmuk, Behçet Açıkmeşe, Andrew W. Jr. Berning, and Geoffrey Huntington. «Successive Convexification for Fuel-Optimal Powered Landing with Aerodynamic Drag and Non-Convex Constraints». In: *AIAA Guidance*,

- Navigation, and Control Conference*. San Diego, California, USA, Jan. 2016 (cit. on p. 11).
- [9] Anett Krammer, Luc Blecha, and Marc Lichtenberger. «Fin actuation, thrust vector control and landing leg mechanisms design for the RETALT VTVL launcher». In: *CEAS Space Journal* 14 (2022), pp. 577–591 (cit. on p. 12).
- [10] Laura Sopegno, Patrizia Livreri, Margareta Stefanovic, and Kimon P. Valavanis. «Thrust Vector Controller Comparison for a Finless Rocket». In: *Article* (2022) (cit. on p. 13).
- [11] Lorenzo Vallini. «Static and Dynamic Analysis of the Aerodynamic Stability and Trajectory Simulation of a Student Sounding Rocket». Relatori: Prof. Luca D’Agostino, Ing. Christian Bach. Corso di Laurea Magistrale in Ingegneria Aerospaziale. Pisa, Italy: Università di Pisa, Facoltà di Ingegneria, 2014 (cit. on pp. 15, 17).
- [12] R. Galejs. *Wind instability / What Barrowman left out*. Retrieved 14 May 2009. 2009 (cit. on pp. 15, 17).
- [13] Alice De Oliveira. «Guidance and Control System Design for Reusable Launch Vehicle Descent and Precise Landing». Doctoral Dissertation. Milano, Italia: Politecnico di Milano, Department of Aerospace Science and Technology, 2023 (cit. on p. 18).
- [14] NASA Glenn Research Center. *Shape Effects on Drag*. National Aeronautics and Space Administration (NASA). URL: <https://www1.grc.nasa.gov/beginners-guide-to-aeronautics/shape-effects-on-drag/> (cit. on p. 18).
- [15] G. Fasano. *Modeling and Optimization in Space Engineering*. Università di Torino. Lectures slides. 2023-24 (cit. on p. 24).
- [16] FICO. *FICO Xpress Optimization Suite: Overview*. URL: <https://www.fico.com/fico-xpress-optimization/docs/dms2021-01/overview.html> (cit. on p. 26).
- [17] wngaw. *Linear Regression*. URL: <https://wngaw.github.io/linear-regression/> (cit. on p. 26).
- [18] M. Vahidi, N. Abar, S. Farokhi, and M. Soltanpour. «Optimal control of a class of nonlinear systems with time-varying state constraints». In: *Automatica* 91 (2018), pp. 235–242 (cit. on p. 27).
- [19] João Luiz da Silva, Marcelo Messias, and Eduardo Jesus de Pinho. «Dynamic Optimal Control Problem of the Nonhomogeneous Heat Equation with State Constraint». In: *Journal of Optimization in Industrial Engineering* 16.2 (2023), pp. 293–301 (cit. on p. 28).

- [20] J. T. Betts. «Survey of numerical methods for trajectory optimization». In: *Journal of Guidance, Control, and Dynamics* 21.2 (1998), pp. 193–207 (cit. on p. 28).
- [21] Matthew Kelly. «An Introduction to Trajectory Optimization: How to Do Your Own Direct Collocation». In: 59 (2017), p. 859 (cit. on p. 29).
- [22] Lucas da Silveira Nascimento, Lídia Nayara de Souza Mota, Rômulo Cezar da Costa, Jaiane da Silva Gomes, Ana Célia de Souza Pires, and Antônio Lúcio Santos Pires. «Direct Collocation Method for Solving Optimal Control Problems». In: *O Cenário da Inovação Tecnológica na Sociedade Brasileira 2*. Ed. by Lídia Nayara de Souza Mota, Rômulo Cezar da Costa, Antônio Lúcio Santos Pires, Jaiane da Silva Gomes, Lucas da Silveira Nascimento, and Ana Célia de Souza Pires. Editora Científica, 2023, pp. 257–272 (cit. on p. 30).
- [23] Matthew Kelly. «An Introduction to Trajectory Optimization: How to Do Your Own Direct Collocation». In: 59 (2017), p. 860 (cit. on p. 31).
- [24] Matthew Kelly. «An Introduction to Trajectory Optimization: How to Do Your Own Direct Collocation». In: 59 (2017), pp. 865–866 (cit. on p. 32).
- [25] Ki-Wook Jung, Sang-Don Lee, Cheol-Goo Jung, and Chang-Hun Lee. *Model Predictive Guidance for Fuel-Optimal Landing of Reusable Launch Vehicles*. arXiv preprint. May 2024. arXiv: 2405.01264 [eess.SY] (cit. on p. 34).
- [26] Philip E. GILL, Elizabeth WONG, Walter MURRAY, and Michael A. SAUNDERS. «User’s Guide for SNOPT Version 7.7: Software for Large-Scale Nonlinear Programming». In: (Mar. 2021), p. 4 (cit. on p. 42).
- [27] Philip E. GILL, Elizabeth WONG, Walter MURRAY, and Michael A. SAUNDERS. «User’s Guide for SNOPT Version 7.7: Software for Large-Scale Nonlinear Programming». In: (Mar. 2021), p. 4 (cit. on p. 43).
- [28] Philip E. GILL, Elizabeth WONG, Walter MURRAY, and Michael A. SAUNDERS. «User’s Guide for SNOPT Version 7.7: Software for Large-Scale Nonlinear Programming». In: (Mar. 2021), p. 16 (cit. on p. 47).
- [29] Matthew Kelly. «An Introduction to Trajectory Optimization: How to Do Your Own Direct Collocation». In: 59 (2017), pp. 849–904 (cit. on p. 49).
- [30] Jascha Wilken and Sven Stappert. «Comparative analysis of European vertical landing reusable first stage concepts». In: *CEAS Space Journal* 17 (1 2024). Published online: 17 April 2024, pp. 113–130. DOI: 10.1007/s12567-024-00549-9. URL: <https://doi.org/10.1007/s12567-024-00549-9> (cit. on pp. 52, 53).

- [31] Martin Sippel and Jascha Wilken. «Selection of propulsion characteristics for systematic assessment of future European RLV-options». In: *CEAS Space Journal* 17 (1 2025). Published online: 18 September 2024, pp. 89–111. DOI: 10.1007/s12567-024-00564-w. URL: <https://doi.org/10.1007/s12567-024-00564-w> (cit. on p. 53).