POLITECNICO DI TORINO

Master's Degree in Mechatronic Engineering

Politecnico
di Torino

Master's Degree Thesis

Design an optimal trajectory planner for pinpoint landing

Supervisors

Prof. CARLO NOVARA

Ing. PAOLO MARTELLA

Candidate

MARCO BARBON

July 2023

Abstract

Future exploratory missions of celestial bodies in the solar system will require spacecraft capable of landing precisely near to predefined targets, such as places of high scientific interest usually on topographically hazardous terrains, human outposts and pre-positioned assets.

This thesis work analyses and implements the guidance problem for a pinpoint landing on Mars. Which requires finding a feasible reference trajectory to safely land a spacecraft on the Martian surface precisely close to a predefined target, optimizing at the same time the fuel consumption.

The Lossless Convexification (LCvx) theory is identified as a very promising method to solve this type of problem. The LCvx allows to formulate the guidance problem as a convex optimization problem, which can be quickly solved by efficient solvers. Then, based on the derived mathematical formulation, an algorithm is implemented, with particular attention to the reliability of the solutions. An optimization of the algorithm is needed, to increase its computational performance. Tests and simulations are carried out to analyze the performance and the robustness of the algorithm. Finally, a parametric analysis relates the behavior of the algorithm with respect to a selection of meaningful design parameters of the lander, such as the configuration of the Radar Doppler Altimeter, the performance of the parachute and the requirements of the Landing Vision System.

In conclusion, the designed algorithm proves to compute optimal trajectories for feasible guidance problems and sub optimal solutions for unfeasible optimization problems. The sub optimal solutions result from a trade off between landing accuracy and fuel consumption. Finally, the parametric analysis highlights interesting correlations between the driver parameters of the analysis and the developed algorithm, leading to useful guidelines for spacecraft design choices.

Acknowledgements

Innanzitutto, vorrei ringraziare il professore Carlo Novara per i preziosi consigli e il costante supporto che mi ha fornito durante tutto lo svolgimento della tesi. Altresì, vorrei ringraziare Thales Alenia Space per avermi accolto in questo progetto, per me di grande ispirazione e crescita professionale. E in particolare Paolo Martella, che in qualità di tutor aziendale, mi ha affiancato nel lavoro. Lo ringrazio per avermi guidato in questo progetto e per aver condiviso con me la sua esperienza rispondendo sempre con dedizione alle mie domande e curiosità.

Un ringraziamento speciale va alla mia famiglia. Ai miei genitori che mi hanno sempre incoraggiato nelle mie scelte, devo alla loro costante fiducia e al loro sostegno questo traguardo. A mio fratello, che primo fra tutti mi ha ispirato con il suo coraggio nel prendere scelte difficili e con la perseveranza che ha dimostrato nel portarle avanti. E a mia sorella, che più di tutti mi ha fatto sentire la sua vicinanza in questi anni passati lontano da casa.

Infine, la mia gratitudine va ai miei amici. A gli amici di sempre, mi avete accompagnato in questo percorso da ben prima che partissi per Torino. Mi avete consigliato nel momento del bisogno e avete festeggiato con me i miei piccoli e grandi traguardi. Ci siete sempre stati e per questo vi ringrazio. E a gli amici di Torino, vi ringrazio per aver condiviso con me questi anni e per aver reso questa esperienza così speciale. Ho imparato tanto da ognuno di voi.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Brief history of autonomous landings on Mars

The exploration of Mars began in the 60s with the space race between the United States of America and the Soviet Union. In that period, the Cold War led the two countries to compete for the supremacy of the space, as a demonstration of power and technological capabilities. The large amount of money available and the pressure from the governments brought a huge leap in the technological development and in the end, great achievements for humanity itself, such as the first man on the Moon.

In this historical background, the first attempts to reach Mars began. In the beginning, attempts from both sides were disastrous, until in 1965 the American mission Mariner 4 managed to accomplish the first fly-by of Mars, which gave us the first close-up photos of the red planet.

On the other hand, the first mission to achieve a soft touch down of a lander on the Martian surface was the soviet Mars 3 in 1971. The lander reached safely the ground, but unfortunately, it shut down almost immediately due to unknown reasons, without transmitting any information back to the Earth. It owns anyway the record of the first human object to safely reach the Martian surface.

Later in 1976, the landers of the American missions Viking 1 and Viking 2 firstly touched down on the surface and sent back information to the Earth.

In 1969, the first man on the Moon sealed the beginning of the end of the space race. Thus, leading to a significant decrease in the interest of the governments in the space exploration. Planned missions were postponed or canceled and in general space campaigns were downsized.

A renovated interest in the space exploration reemerged only at the end of the

last millennium. While, the no longer URSS was struggling to continue its space campaign, the NASA has resumed launching different types of missions, bringing several orbiters, landers, and rovers to Mars over the years.

In the history of autonomous landings on Mars, it is worth mentioning the missions: Mars Pathfinder, Mars Exploration Rover, Phoenix Mars Lander, Mars Science Laboratory, InSight, Mars 2020. The Mars Pathfinder in the 1997 safely landed the first rover Sojourney on the Mars surface, which was used to test technologies for the next missions. The Mars Exploration Rover in 2003 brought other two rovers Spirit e Opportunity to Mars. The Phoenix Mars Lander in the 2007 was in charge to study the Martian environment, looking for microbial life and water. The Mars Science Laboratory in the 2011 landed the rover Curiosity. The Insight mission in the 2018, to study the internal structure of the planet. And the most recent mission Mars 2020, with the most advanced Martian rover Perseverance, and the first Martian drone Ingenuity.

This new phase is also characterized by new countries involved in the exploration of Mars, such as the European Union with the missions Mars Express and ExoMars, India with the Mars Orbiter Mission, United Arab Emirates with the Emirates Mars Mission, and China with the Tianwen-1 mission.

Future missions on Mars will be the NASA low cost mission EscaPADE, the second Indian mission Mars Orbiter Mission 2, the second part of the ExoMars mission led by the ESA. As well as the mission Mars Sample and Return in collaboration between NASA and ESA, which is planning to gather the Martian soil samples collected by the rover Perseverance over the years and make the first launch from an extraterrestrial planet toward the Earth, to bring back the samples letting the scientists to examine them.

1.2 The new challenge of the precision landing

In recent years, the renewed interest in the exploration and scientific research of celestial bodies of our solar system has led to a growing demand for technological advances in the space field.

An important skill in the upcoming missions will be the ability to perform a precision landing, also known as pinpoint landing, i.e. the autonomous guidance of a lander toward the target, reaching it with an accuracy of less 100 m. Where the precision can be measured using the landing ellipse, i.e. the region where it is 99 percent likely the spacecraft will land [1].

Previous missions had large landing ellipses, as we can see in Figure 1.1.

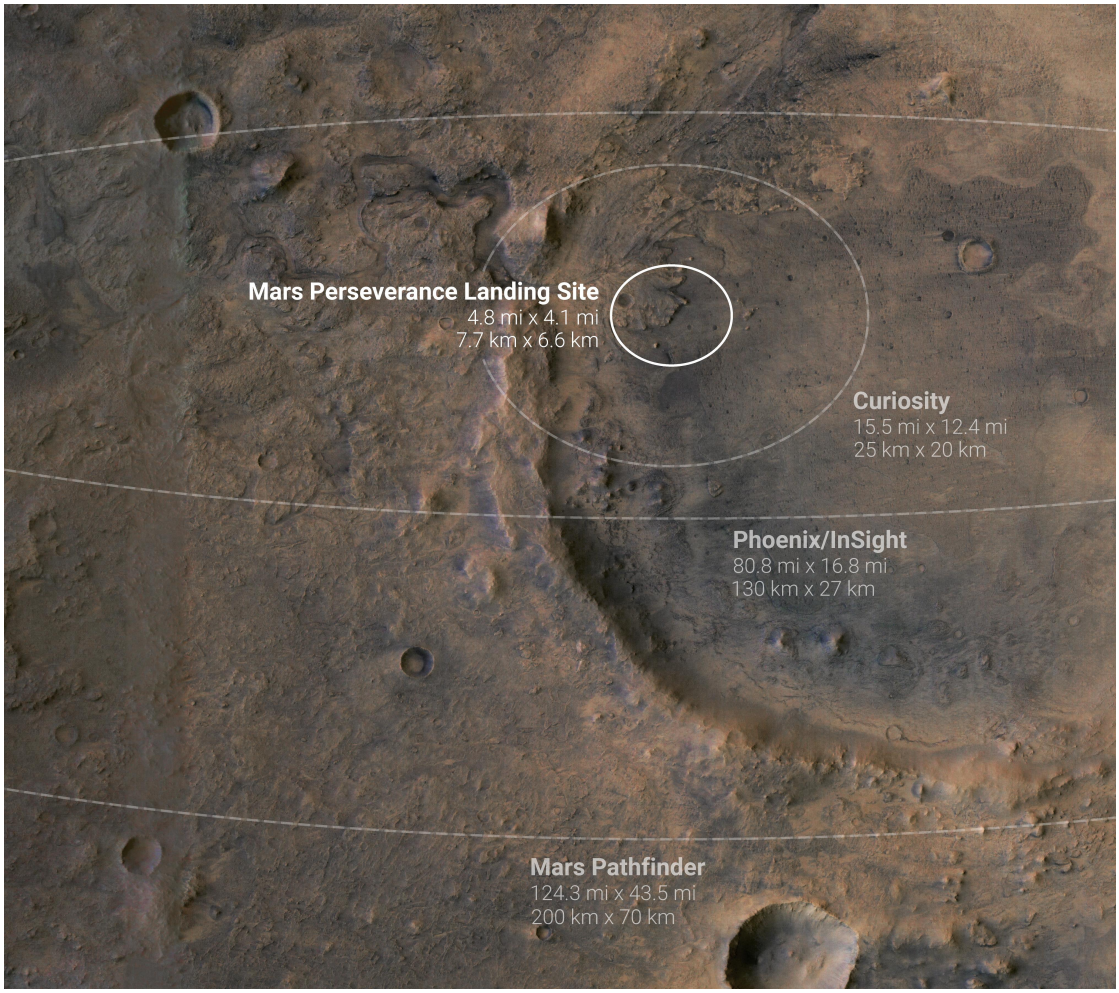


Figure 1.1: Landing ellipses comparison for five different NASA missions to Mars. Image credit: NASA/JPL-Caltech

This was due to multiple factors [2], such as:

- Delivery error at the entry interface. Where the delivery error is defined as the difference between the desired position and velocity and the obtained ones.
- Knowledge uncertainty at the entry interface, resulting from the accumulated sensor errors since the last navigational update.
- Environmental uncertainty, due to the deviation of atmospheric properties from those expected.
- Vehicle performances, that will be inevitably different from those modeled.

The cumulative effect of these factors has always led to a landing ellipse wide kilometers.

Therefore, in order to ensure a safe touch down, the choice of the landing site has always been constrained to areas compatible with the expected landing ellipse. Leading in past missions to a trade off in the choice of the landing site, between scientific interest and landing safety.

But, thanks to the new technologies that are coming in the field of the precision landing, fascinating new opportunities can be explored, such as:

- Explore Martian caves and valleys, which are places of high scientific interest, and appealing sites for future human outposts.
- Explore target locations with topographically hazardous terrain.
- Return samples from other planets, as in the Mars Sample and Return mission.
- Set up permanent outposts through the solar system.
- Land close to pre-positioned assets.
- Make cheaper reusable rockets, that after every flight can be simply refueled and reused, like airplanes. Leading to cheaper space flights and a more affordable space economy.

1.3 Tasks of the GNC in a precision landing mission

The Guidance Navigation and Control (GNC) is the piece of software dedicated to autonomously guide the spacecraft during its descent toward the Martian surface, throughout the Entry, Descending, and Landing phases. The GNC computes the trajectory from the spacecraft's initial location to the target. It estimates position, velocity, and attitude from the sensor measurements. Then, exploiting this information, it implements a control strategy to keep the vehicle on track.

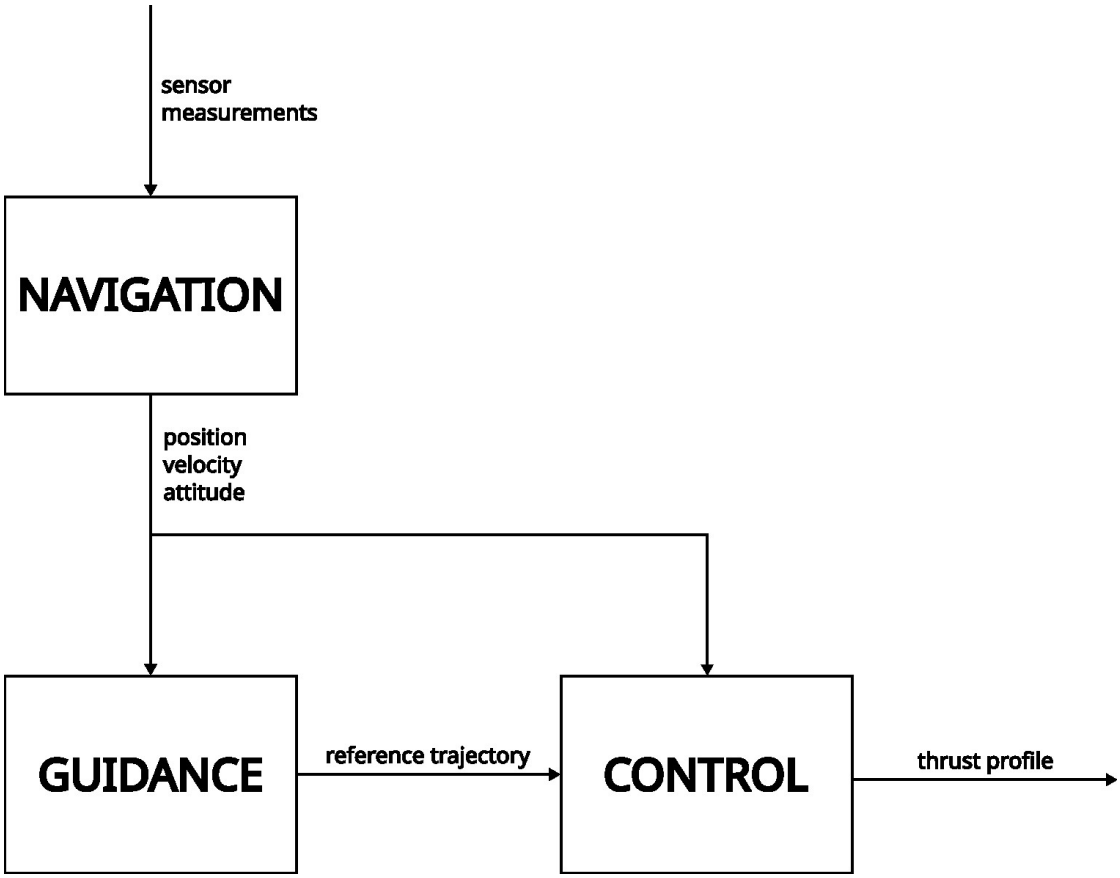


Figure 1.2: GNC scheme.

The Entry phase begins at the end of the interplanetary journey, when the lander encounters the atmosphere of the planet, traveling at hypersonic speed. Previous missions used a ballistic trajectory to guide the lander toward the target, without any control over its effective position. A pinpoint landing, instead, requires a guided entry [3], which through bank reversal maneuvers controls the asset of the

spacecraft and thus the drag and lift forces applied, and ultimately the trajectory itself.

Following, during the descent phase, the parachute is opened to slow down the descent velocity. Unfortunately, the rarefied atmosphere of Mars does not allow to reach sufficiently low speeds, and therefore an additional phase is necessary to land safely.

Finally, in the landing phase, the spacecraft turns on the thrusters to slow down the velocities and to perform a soft landing on the surface. Whereas, previous missions were concerned only with the safety of the maneuver, the next missions must also guarantee the precision of the landing. Then, a divert maneuver is needed to recover the horizontal errors of the previous phases and reach the target. At the same time, the maneuver must be optimized to save as much fuel as possible, since every kilogram saved can reduce mission costs or increase the amount of landed payload. For this maneuver becomes necessary to exactly locate the spacecraft on the map of the Martian surface. But, we can no longer rely on the GPS as on the Earth, then a Landing Vision System (LVS) is used to compute the exact location of the lander, acquiring and comparing images of the surface.

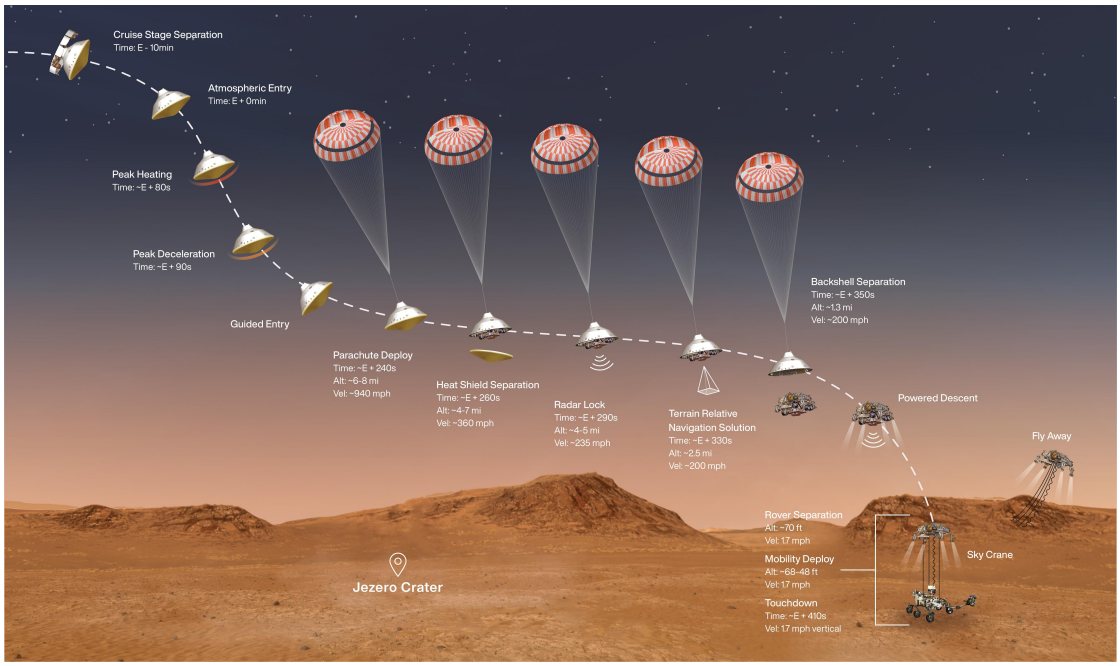


Figure 1.3: Perseverance rover's Entry, Descent, and Landing profile. Image credit: NASA/JPL-Caltech.

1.4 A survey of mathematical methods to compute the landing reference profile

In this section, it is presented a selection of methods used and/or proposed over the time to generate reference trajectories for spacecraft landing. A more detailed description of the proposed methods can be found in the papers [2] [4] [5].

Explicit methods

Explicit methods analytically derive the acceleration and the thrust profile of the spacecraft landing maneuver.

These methods turn out to be computational efficient, thanks to the closed form of the solution of the guidance problem. But, on the other hand, they have difficulty to take into account constraints, which could lead to unfeasible trajectories. Moreover, they compute only sub optimal solutions with respect to the fuel consumption.

- **Gravity Turn**
  The Gravity Turn is a guidance method with a strong heritage. Indeed, it was the first guidance law used by the NASA in the Surveyor missions. It computes the acceleration profile, keeping the thrust vector aligned with the velocity vector, so as to obtain zero velocity at the touch down.

  $$a(t) = -a\frac{v(t)}{\|v(t)\|^2} \tag{1.1}$$

  This guidance law computes the landing trajectory based only on the initial position and velocity of the spacecraft, no adjustments are made during the descent. This makes not possible to achieve precise landings.

- **E-Guidance**
  The E-Guidance solves analytically the two-point boundary-value problem, i.e. computes the acceleration profile of the spacecraft, once defined the initial and final conditions of the position and the velocity. It assumes a linear relationship between acceleration and time to go (time of flight from the instantaneous position to the target).

  $$\begin{aligned} a(t) &= c_0 + c_1 t_{go} \\ t_{go} &= t_f - t \end{aligned} \tag{1.2}$$

- **Apollo Guidance**
  The Apollo guidance is a modified E-guidance, with a quadratic relationship

between acceleration and time, allowing to constrain the final acceleration and thus the final attitude of the spacecraft.

$$a(t) = c_0 + c_1 t + c_2 t^2 \tag{1.3}$$

It has been used in six successful landing missions on the Moon, including the Apollo missions from which it takes its name.

Its strong heritage has led this algorithm to be revised several times through time, leading to the so called Modified Apollo technique. It improves the Apollo guidance, choosing a total flying time that optimize a certain criterion, such as the fuel consumption, usually by means of a linear search of the time of flight.

Numerical methods

Numerical methods compute numerically the acceleration and thrust profile of the spacecraft landing maneuver.

In general, these methods allow to formulate the guidance problem without introducing any simplification and taking into account all the possible constraints. The optimization theory is used to find a solution with a real optimal fuel consumption. On the other hand though, they usually need high computational efforts to be solved, which limit their use in real-time applications.

- **Convex Guidance**
  Convex guidance solve the guidance problem, formulating it as a convex optimization problem. This formulation allows solving the optimization problem with a low computational effort, thanks to algorithms specifically designed to solve efficiently this type of problems. To cast the guidance problem into a convex optimization problem, we need to exploit the Lossless Convexification theory (LCvx), developed by the NASA specially for this use. Unfortunately, this mathematical theory is still under development, and it does not include all the possible formulations of the problem. But, it is still more versatile than any other explicit method.

- **Gradient-based Optimization**
  Gradient-based optimization is a classical numerical optimization technique, that exploits the gradient direction to find the maximum or the minimum of a given function. It can take into account every kind of constraint and formulation of the guidance problem. But unfortunately, it turns out to be computationally expensive and there is no evidence to obtain a global optimal solution.

- **Neural Guidance**
  It is an innovative approach, which proposes the use of an artificial neural network (ANN) to solve the guidance problem. In this case, we do not have any limitation on assumptions and constraints, because the AAN exploits only the numerical relationship between a set of input and output data, without involving any physical formulation of the problem. This method has proved to be very robust in many tests, but there is still no mathematical evidence of the convergence of the solution, this makes it difficult to be used in real applications.

1.5 Tradeoff and justification for the chosen methodology

The method chosen in this thesis work to solve the guidance problem, was selected among those presented in the previous section, according to the following features:

- **Performance**
  The chosen method must guarantee a precise landing, with an accuracy of less than 100 m, and at the same time, the minimum fuel consumption possible.
- **Robustness**
  The chosen algorithm must provide high standards of reliability, to ensure the successes of the mission. A robust algorithm is needed to deal with unpredictable exceptions.
- **Computational efficiency**
  Since the guidance problem is a real time application, we need an efficient algorithm, with a low computational time.

The comparison of the proposed methods is summarized in the Tables 1.1 and 1.2. Otherwise, a more detailed comparison can be found in [2].

	Gravity Turn	E-Guidance	Apollo Guidance
Performance	Poor	Poor	Moderate (modified guidance)
Robustness	Poor	Poor	Poor
Efficiency	Good	Good	Good

Table 1.1: Comparison of explicit methods.

	Convex Guidance	Gradient-based Optimization	Neural Guidance
Performance	Good	Good	Good
Robustness	Good	Good	Good
Efficiency	Moderate	Poor	Poor

Table 1.2: Comparison of numerical methods.

As we can see from the comparison results presented in Tables 1.1 and 1.2, numerical methods outperform explicit methods in this kind of applications, but at the same time, they struggle with computational effort. Among them, only convex guidance guarantees computational performances high enough to be used in a real time application, motivating its choice as guidance method used in this thesis work.

Chapter 2

Formalization of the guidance problem for Pinpoint landing

In this chapter, the trajectory planning optimization problem for pinpoint landing it is introduced and formalized.

Actually, the mathematical formulation of the guidance problem is not difficult to be determined, rather the problems arise when we want to implement it by means of some numerical algorithms with the intent to obtain an unique and reliable solution. As discussed in the following sections, to accomplish this purpose the optimization problem must be suitable formulated as a convex optimization problem, more specifically for this application it is cast into a second order cone program (SOCP). This formulation indeed guarantees to achieve an unique solution by means of very efficient e reliable algorithms. Indeed, convex optimization problems have been studied for a long time and there are already available a lot of specific solvers deeply studied and tested, supported by a strong mathematical theory. These appealing features make this kind of approach suitable for real-time online applications, as it is required by the nature of the problem.

2.1 Mathematical preliminaries

In order to understand the problem formulation, we need to introduce the fundamental mathematical concepts that are involved in this study. This section gathers a selection of definitions from Boyd's "Convex Optimization" book [6].

2.1.1 Optimization problem

An optimization problem is a class of mathematical problem of the form:

$$\begin{aligned} \min_{x} \quad & f_0(x) \\ \text{subject to} \quad & f_i(x) \leq 0,\ i = 1, \dots, m \\ & h_j(x) = 0,\ j = 1, \dots, p \end{aligned} \tag{2.1}$$

where a cost function $f_0(x) : \mathbb{R}^n \to \mathbb{R}$ is aimed to be minimized over the optimization variable $x \in \mathbb{R}^n$ constrained by m inequalities $f_i(x) \leq 0$ and p equalities $h_j(x) = 0$.

Every optimization variable that satisfy all the constraints is said to be feasible. The set of all the feasible optimization variables defines the feasible set $X = \{x \in \mathbb{R}^n : f_i(x) \leq 0,\ i = 1, \dots, m;\ h_j(x) = 0,\ j = 1, \dots, p\}$. Eventually the problem is said to be feasible if exist at least one feasible solution of the problem.

Similar definitions can be derived for unfeasible optimization variables, an unfeasible set and an unfeasible problem.

Solving an optimization problem means find the optimal minimum value of the cost function, that is called optimal value

$$p^* = inf\{f_0(x) \mid f_i(x) \leq 0,\ i = 1, \dots, m,\ h_j(x) = 0,\ j = 1, \dots, p\} \tag{2.2}$$

For upper and lower unconstrained optimization problems p^* can take values as $\pm\infty$. If the problem is unfeasible than conventionally $p^* = \infty$. If the problem is feasible but no optimal solution exist than we say that the optimal value p^* is not attained at any finite point.

The solution of the problem p^* is associated to a set of feasible points $x^* \in X\ s.t.\ f_0(x^*) = p^*$, that are called optimal points.
The set of the optimal points is called optimal set

$$X_{opt} = \{x \mid f_i(x) \leq 0,\ i = 1, \dots, m,\ h_j(x) = 0,\ j = 1, \dots, p,\ f_0(x) = p^*\} \tag{2.3}$$

Equivalently it can be formulated exploiting the arg min notation

$$X_{opt} = arg \min_{x \in X} f_0(x) \tag{2.4}$$

2.1.2 Local and Global optimal solutions

Whereas the global optimal solution coincides with the solution of the original optimization problem, a local optimal solution is a solution of the optimization problem on a domain restricted to a subset of the feasible set

$$\begin{array}{ll} \min & f_0(z) \\ \text{subject to} & f_i(z) \leq 0,\ i = 1, \ldots, m \\ & h_j(z) = 0,\ j = 1, \ldots, p \\ & \|z - x\|_2 \leq R \end{array} \tag{2.5}$$

Roughly speaking, a local solution is the minimum of f_0 over nearby points, an intuitive example case is reported in the Figure 2.1.

In conclusion, solving an optimization problem means seeking the global solution, but actually problems arise when we try to retrieve it by means of numerical algorithms. Indeed, there are no software able to distinguish between a local and a global solution. Thus, to guarantee the correctness of the computed solution, the problem needs to be properly formulated as a convex optimization problem. Indeed, as we will see in the next sections, this specific type of optimization problem takes advantage of some own property to assure always to find global solutions.

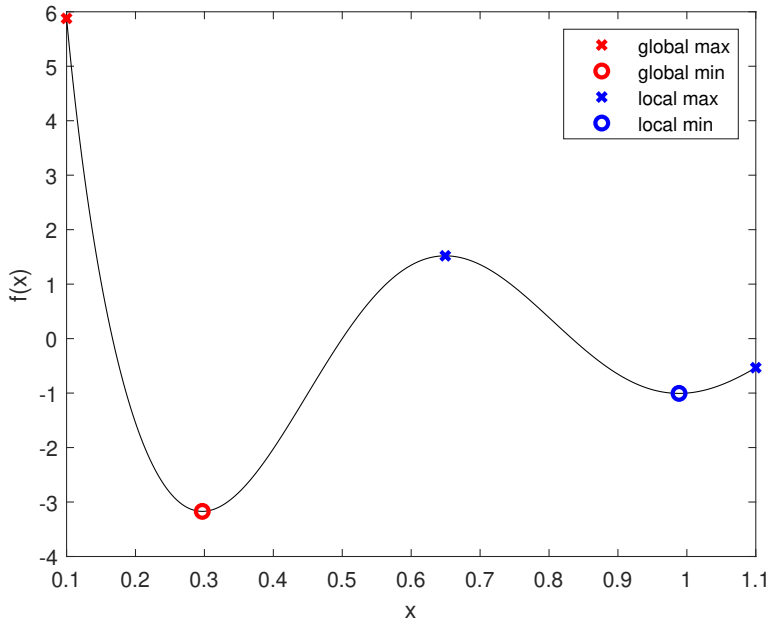


Figure 2.1: Global and local solutions of the function $f(x) = \frac{cos(3\pi x)}{x}$ for $x \in [0.1,1.1]$.

2.1.3 Convexity

Line, half line and line segment

Given two different points $x_1, x_2 \in \mathbb{R}^n$ and a parameter $\theta \in \mathbb{R}$, the line passing through x_1, x_2 can be defined as the set of points $y \in \mathbb{R}^n$ such that

$$y = \theta x_1 + (1 - \theta)x_2 \tag{2.6}$$

Similarly we can define the half line starting from x_2 and passing through x_1 as

$$\begin{aligned} &y = \theta x_1 + (1 - \theta)x_2 \\ &\theta \geq 0 \end{aligned} \tag{2.7}$$

And the line segment between x_1, x_2 as

$$\begin{aligned} &y = \theta x_1 + (1 - \theta)x_2 \\ &0 \leq \theta \leq 1 \end{aligned} \tag{2.8}$$

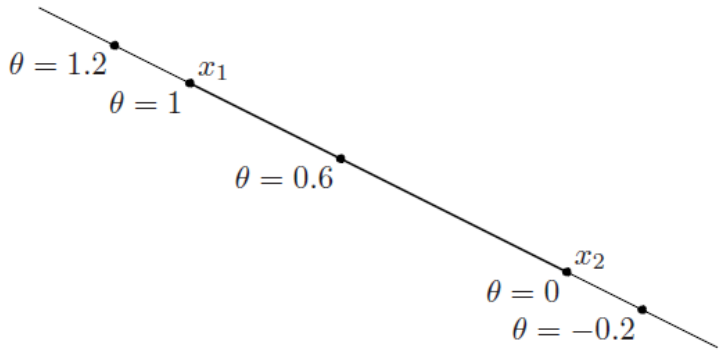


Figure 2.2: Parametric representation of a line and a line segment (darker) between two points $x_1, x_2 \in \mathbb{R}^2$ [6].

Affine sets

A set $A \subseteq \mathbb{R}^n$ is an affine set if for any two points $x_1, x_2 \in A$ the line passing through x_1, x_2 lies on the set A itself

$$\theta x_1 + (1 - \theta)x_2 \in A, \ \theta \in \mathbb{R} \tag{2.9}$$

Alternatively a set $A \subseteq \mathbb{R}^n$ can be defined as an affine set if it contains every affine combination of its points. Where an affine combination of a set of points

$x_1, \ \dots \ , x_k$ is a linear combination of the points with coefficients that sum up to one

$$\begin{aligned} &\theta_1 x_1 + \cdots + \theta_k x_k \\ &\theta_1 + \cdots + \theta_k = 1 \end{aligned} \tag{2.10}$$

Convex sets

A set $C \subseteq \mathbb{R}^n$ is a convex set if for any two points $x_1, x_2 \in C$ the line segment between x_1, x_2 lies on the set C itself

$$\theta x_1 + (1-\theta) x_2 \in C, \ 0 \leq \theta \leq 1 \tag{2.11}$$

This means that every affine set is also convex.
Similarly to affine sets, we can define a convex set in an alternative way exploiting the concept of convex combination. A set $C \subseteq \mathbb{R}^n$ is a convex set if it contains every convex combination of its points. Where a convex combination of a set of points $x_1, \ \dots \ , x_k$ is a linear combination of the points with positive coefficients that sum up to one

$$\begin{aligned} &\theta_1 x_1 + \cdots + \theta_k x_k \\ &\theta_1 + \cdots + \theta_k = 1 \\ &\theta_i \geq 0, \ i = 1, \ \dots \ k \end{aligned} \tag{2.12}$$

Cones

A set C is a cone if for every point $x \in C$ the half line passing through x and starting from the origin lies on the set C itself

$$\theta x \in C, \ \theta \geq 0 \tag{2.13}$$

If a set is both a cone and a convex set, then it is called convex cone. It follows that for any points $x_1, x_2 \in C$ and $\theta_1, \theta_2 \geq 0$ we have $\theta_1 x_1 + \theta_2 x_2 \in C$.

Some important convex sets

- **Hyperplane** $\{x \in \mathbb{R}^n \mid a^T x = b\},\ a \in \mathbb{R}^n,\ a \neq 0,\ b \in \mathbb{R}$
  It is a generalization of a plane in more dimensions. It is an affine set and thus also a convex set. Geometrically, an hyperplane is characterized by a normal vector $a \in \mathbb{R}^n$ and an offset from the origin $d = \frac{|b|}{\|a\|_2} \in \mathbb{R}$.

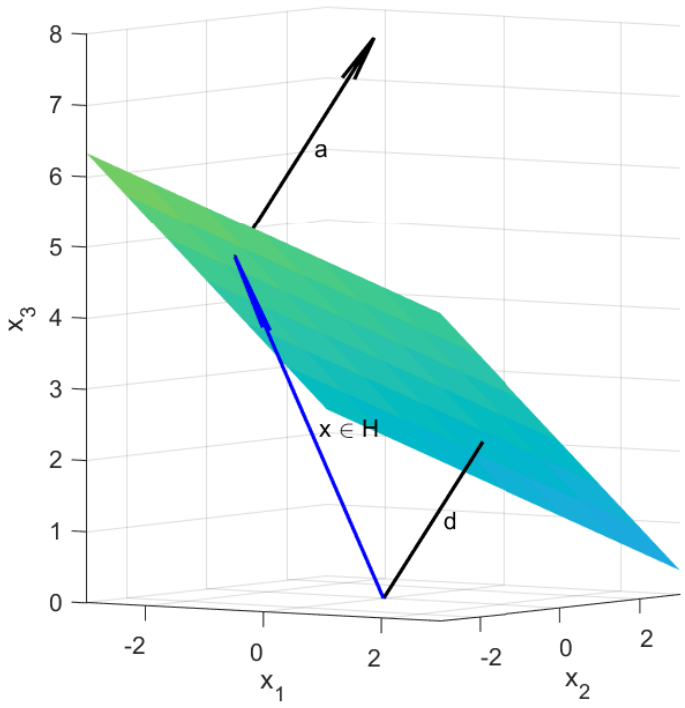


Figure 2.3: Hyperplane $H \subseteq \mathbb{R}^3$, with $a = [1,2,3]^T$ and $b = 10$.

- **Halfspace** $\{x \in \mathbb{R}^n \mid a^T x \leq b\},\ a \in \mathbb{R}^n,\ a \neq 0,\ b \in \mathbb{R}$
  The related hyperplane $H = \{x \in \mathbb{R}^n \mid a^T x = b\}$ separates $\mathbb{R}^n$ in two parts, these portions of space are called halfspaces.

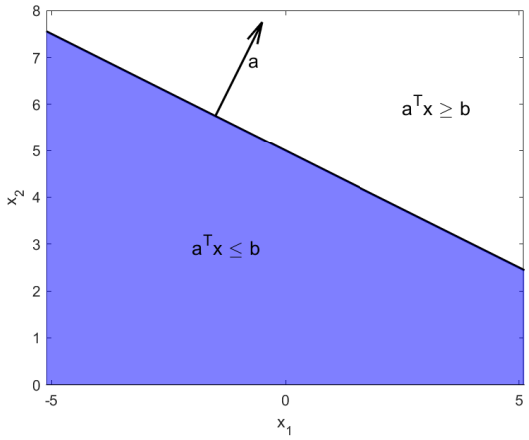


Figure 2.4: The hyperplane $H \subseteq \mathbb{R}^2$, with $a = [1,2]^T$ and $b = 10$, separates $\mathbb{R}^2$ in two halfspaces $H_{--} = \{x \in \mathbb{R}^n \mid a^T x \leq b\}$ and $H_{++} = \{x \in \mathbb{R}^n \mid a^T x \geq b\}$.

- **Euclidian ball** $\{x \in \mathbb{R}^n \mid \|x - x_c\|_2 \leq r\},\ r > 0$
  It is a generalization of a sphere in more dimensions. Geometrically, the euclidian ball is characterized by a center $x_c \in \mathbb{R}^n$ and a radius $r \in \mathbb{R}$.

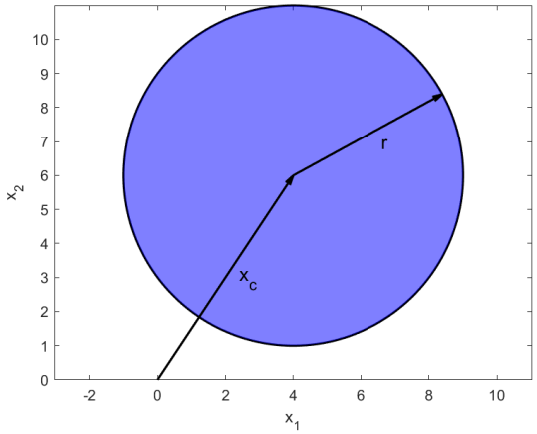


Figure 2.5: Euclidian ball $B \subseteq \mathbb{R}^2$, with $x_c = [4,6]^T$ and $r = 5$.

- **Second order cone** $\{(x,t) \in \mathbb{R}^{n+1} \mid \|x\|_2 \leq t\}$
  It is an instance of a convex cone. It is used in the second order cone programming theory to describe constraints on the euclidian norm of a vector, as we will see in the SOCP section.

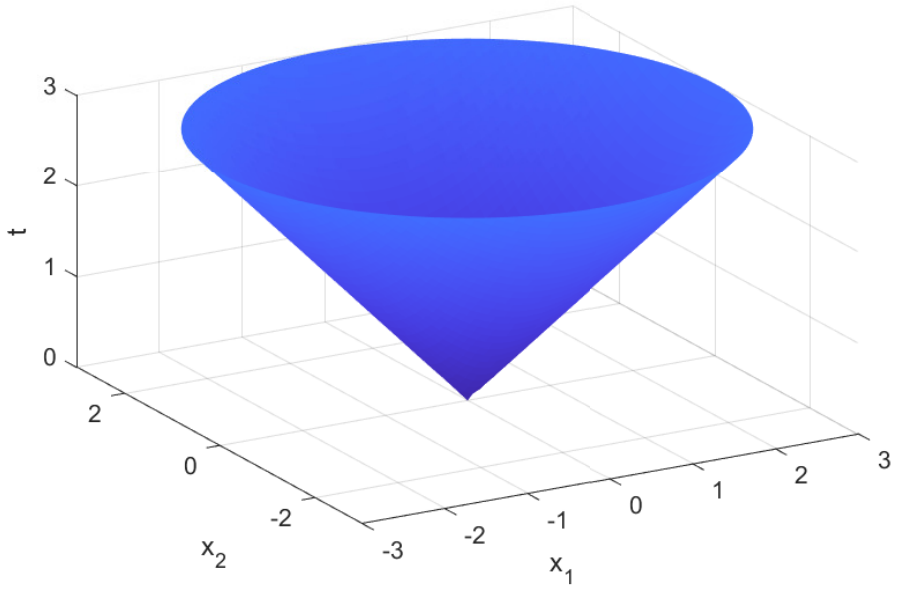


Figure 2.6: Second order cone $C \subseteq \mathbb{R}^3$.

Convex functions

Given a multivariate function $f : \mathbb{R}^n \to \mathbb{R}$ well defined in its domain $domf = \{x \in \mathbb{R}^n : \ -\infty < f(x) < \infty\}$, the function f is said to be convex if $domf$ is a convex set and for all $x_1, x_2 \in domf$ and all $\lambda \in [0,1]$ it holds that

$$f(\lambda x_1 + (1-\lambda)x_2) \leq \lambda f(x_1) + (1-\lambda) f(x_2) \tag{2.14}$$

Geometrically, it means that any line segment between x_1, x_2 lies above the function's graph, as we can see in the Figure 2.7.

Figure 2.7: Graph of a convex function $f(\cdot)$, where the line segment between two generic points $(x, f(x))$ $(y, f(y))$ lies above the function itself [6].

The function f is concave if $-f$ is convex.
Finally, f is strictly convex if the above inequality holds strictly, for all $x_1 \neq x_2$ and for all $\lambda \in (0,1)$. Similarly, f is strictly concave if $-f$ is strictly convex.

2.1.4 Convex optimization problem

A convex optimization problem is defined as a mathematical problem in the form

$$\begin{aligned} \min_{x} \quad & f_0(x) \\ \text{subject to} \quad & f_i(x) \leq 0,\ i = 1, \ldots, m \\ & h_j(x) = 0,\ j = 1, \ldots, p \end{aligned} \tag{2.15}$$

where

- the objective function f_0 is convex
- the inequality constraints $f_1, \ \ldots \ , f_m$ are convex
- the equality constraints $h_1, \ \ldots \ , h_p$ are affine $\Big(h_j(x) = a_j^T x - b_j = 0,\ j = 1, \ldots, p\Big)$

With these additional features, the optimization problem has a feasible set $X = \{x \in \mathbb{R}^n : f_i(x) \leq 0,\ i = 1, \ldots, m;\ h_j(x) = 0,\ j = 1, \ldots, p\}$ that is an intersection of convex sets, therefore it is a convex set itself. Thus in a convex optimization problem a convex function f_0 is minimized over a convex feasible set X.

Finally, we introduce the theorem that we were looking for to find the global optimal solution of an optimization problem.

Theorem 1. *If an optimization problem is convex, then any locally optimal solution is also globally optimal.*
Moreover the set of all the optimal solutions X_{opt} is convex .

Proof. Let x^* be a local optimum point, i.e. there is $R > 0$ s.t.

$$f_0(x^*) \leq f_0(z) \quad \forall z \in X \cap B(x^*, R) \tag{2.16}$$

where $B(x^*, R) = \{z \mid \|z - x^*\|_2 \leq R\}$ is the euclidian ball of radius R around x^*.

By contradiction, assume that x^* is not a global optimum point, i.e. there is $y \in X$ s.t. $f_0(y) < f_0(x^*)$.
Take $\theta \in (0,1)$ s.t. $x_\theta = \theta x^* + (1-\theta)y \in B(x^*, R)$. Then by the local optimum condition

$$f_0(x^*) \leq f_0(x_\theta) = f_0(\theta x^* + (1-\theta)y) \tag{2.17}$$

by the convexity property of the function f_0

$$f_0(\theta x^* + (1-\theta)y) \leq \theta f_0(x^*) + (1-\theta) f_0(y) \tag{2.18}$$

and eventually by the assumption that $f_0(y)$ is the global minimum

$$\theta f_0(x^*) + (1-\theta) f_0(y) < \theta f_0(x^*) + (1-\theta) f_0(x*) = f_0(x^*) \tag{2.19}$$

Putting together all of these inequality we obtain

$$f_0(x^*) \leq f_0(\theta x^* + (1-\theta)y) \leq \theta f_0(x^*) + (1-\theta) f_0(y) < f_0(x^*) \tag{2.20}$$

Therefore, the assumption that x^* is not the global optimal point results incorrect, indeed the first and last term of the inequality (2.20) realize an impossible statement. □

Since numerical algorithms are not able to distinguish between local and global solutions the only way to be sure to find a global solution is to solve an optimization problem that has only global solutions, that is a convex optimization problem.

2.1.5 The methodology of Second-Order Cone Programming

A generalization of the most common classes of convex optimization problems, such as linear programming (LP) and quadratic programming (QP), is the second order cone programming (SOCP), where a linear cost function is minimized over a special convex set, that is a second order cone:

$$\begin{aligned} \min_{x} \quad & f^T x \\ \text{subject to} \quad & \|A_i x + b_i\|_2 \leq c_i^T x + d_i, \ i = 1, \ \dots \ , m \\ & Fx = g \end{aligned} \tag{2.21}$$

where inequality constraints are expressed in the standard form of a second order cone (SOC) on variable $x \in \mathbb{R}^n$

$$\|Ax + b\|_2 \leq c^T x + d \tag{2.22}$$

with $A \in \mathbb{R}^{m,n}$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$, $d \in \mathbb{R}$.

SOCP includes several other simpler classes of convex optimization problems, such as linear programs (LP), quadratic programs (QP) and quadratic constrained quadratic programs (QCQP). All of them can be reformulated to be cast into a SOCP.

SOCP has been discussed in details because it turns out to be the best structure to describe the physical problem under analysis. Indeed, constraints related to the square norm of the optimization variables need to be cast into SOC constraints.

2.2 Physical formulation of the problem and the constraints

Considering the original problem to find the optimal trajectory for pinpoint landing, we need to formulate a convex optimization problem and solve it to find the optimal control action sequence that allow us to perform a large divert maneuver from an uncertain initial location of the lander to the desired landing site with the lowest possible fuel consumption. This problem must be solved in real time by means of a numerical algorithm that provides the global optimal solution of the problem in the shortest possible time.

2.2.1 Reference frames

First of all, we need to define the reference frames used to unambiguously describe the vectors in the following sections. Every reference frame R can be defined given his center P and his three main orthogonal axes $\{i, j, k\}$

- **Mars-centered Mars Mean Equator and IAU-vector of Date**
  It is an important reference frame for any landing operations on Mars.
  $R_{MMED} = \{P_{MMED}, i_{MMED}, j_{MMED}, k_{MMED}\}$
  - P_{MMED}: center of Mars
  - i_{MMED}: oriented along the IAU-vector (along the intersection of the Mars mean equator plane and the Earth mean equator of epoch J2000 plane)
  - j_{MMED}: over the Mars mean equator plane, such that $j_{MMED} = k_{MMED} \times i_{MMED}$
  - k_{MMED}: orthogonal to the Mars mean equator plane, oriented to North
- **Mars centered Mars Fixed**
  Reference frame that rotates together with the planet.
  $R_{MMF} = \{P_{MMF}, i_{MMF}, j_{MMF}, k_{MMF}\}$
  - P_{MMF}: center of Mars
  - i_{MMF}: over the Mars mean equator plane, oriented toward the prime meridian (which passes through a crater named Airy-0, located in the Southern hemisphere)
  - j_{MMF}: over the Mars mean equator plane, such that $j_{MMF} = k_{MMF} \times i_{MMF}$
  - k_{MMF}: orthogonal to the Mars mean equator plane, oriented to North

- **East North Up at the target location**
  It is the main reference frame used in this thesis work
  $R_{ENU} = \{P_{ENU}, i_{ENU}, j_{ENU}, k_{ENU}\}$
  - P_{ENU}: the target location on the surface of Mars
  - i_{ENU}: oriented to East
  - j_{ENU}: oriented to North
  - k_{ENU}: oriented upwards along the nadir-zenith direction
- **Body Reference Frame**
  $R_{BRF} = \{P_{BRF}, i_{BRF}, j_{BRF}, k_{BRF}\}$
  - P_{BRF}: center of the mass of the lander spacecraft
  - i_{BRF}, j_{BRF}: over the plane orthogonal to the symmetry axis and passing trough P_{BRF}
  - k_{BRF}: oriented along the symmetry axis of the spacecraft downwards

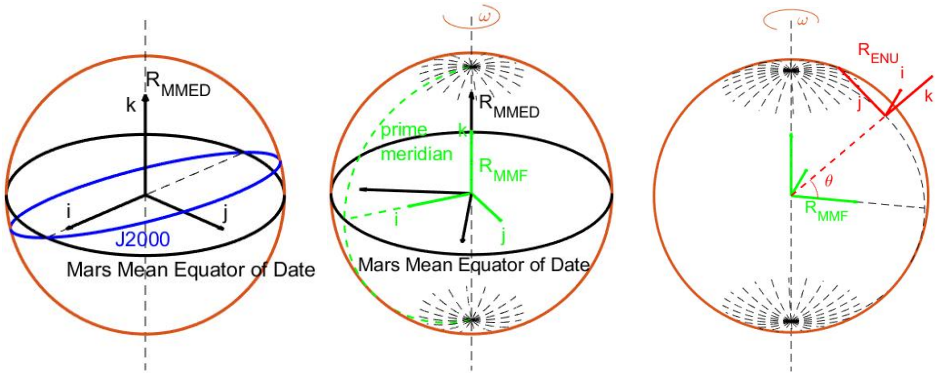


Figure 2.8: Representation of the reference frames defined on Mars. From the left to the right MMED, MMF, ENU.

2.2.2 Dynamics behaviour of the system

To simplify the model, the lander is represented by a point mass that moves inside a three dimensional space. The attitude dynamic instead is neglected, because the attitude reference profile can be derived from the result of the optimization problem. It is obtained by the orientation of the thrust vector in the space. Indeed, since the thrusters are fixed in the lander spacecraft, a given partition of the thrust components implies a well identified attitude of the spacecraft in the working reference frame R_{ENU}. This approach in practice is correct as far as the attitude

dynamic is faster than the translational dynamic. Therefore, this condition gives some design guide lines on the control algorithm, we need always to assure the desired dynamic of the attitude, even if it implies a worse performance on the translational dynamic.

The forces acting on the lander considered in this study are the martian gravitational force and the thrust force, we neglect the aerodynamic forces due to the winds, because the speed of the lander is such that the entity of these forces is much smaller than the others taken into account.

We represent the translational dynamic of the vehicle with respect to the reference frame fixed on the desired landing site on the Mars surface R_{ENU}

$$\begin{aligned} \dot{r}(t) &= v(t) \\ \dot{v}(t) &= g + \frac{T_c(t)}{m(t)} - \omega^{\times}\omega^{\times}r(t) - 2\omega^{\times}v(t) \end{aligned} \tag{2.23}$$

where $r \in \mathbb{R}^3$ is the position, $v \in \mathbb{R}^3$ is the velocity, $g \in \mathbb{R}^3$ is the Mars gravity force, $\omega \in \mathbb{R}^3$ is the Mars angular velocity, $\omega^{\times}$ is the skew-symmetric matrix representation of the cross product $\omega \times (\cdot)$, $T_c \in \mathbb{R}^3$ is the rocket thrust, $m \in \mathbb{R}$ is the mass of the lander.
The Mars gravity force g can be represented in R_{ENU} as $g = [0,0,g_M]^T$, with $g_M = -3.7114\ m/s^2$. This value is considered constant in this study, because the spacecraft remains always sufficiently near to the surface during the maneuver.
The Mars angular velocity ω can be computed in R_{MMF} as

$$\omega = \frac{2\pi}{T_{siderealMars}}[0,0,1]^T \tag{2.24}$$

and therefore in R_{ENU} becomes

$$\omega = \frac{2\pi}{T_{siderealMars}}[cos(\theta),0,sin(\theta)]^T \tag{2.25}$$

where $T_{siderealMars} = 24.6229\ days$ is the period of a sidereal revolution of Mars, $\theta \in \mathbb{R}$ is the latitude of the landing site.

Then we need to take into account the mass dynamic of the lander

$$\dot{m}(t) = -\alpha\|T_c(t)\|_2 \tag{2.26}$$

where $m \in \mathbb{R}$ is the mass of the lander (with $m = m_{dry} + m_{fuel}$, m_{dry} is the mass of the lander without fuel, m_{fuel} is the mass of the fuel that decreases over the time), $\alpha = \frac{1}{I_{sp}g_e}$ is the fuel consumption rate, $g_e \approx 9.807\ m/s^2$ is the gravitational acceleration of the Earth, I_{sp} is the rocket engine's specific impulse.

2.2.3 Constraints

- **Lower and upper bounds on the thrust magnitude**

$$\rho_{min} \leq \|T_c(t)\|_2 \leq \rho_{max} \tag{2.27}$$

The thrust vector $T_c(t)$ is the result of the overall action of the n_{eng} thrusters $T_{single}(t)$ mounted on the bottom of the spacecraft. As shown in the Figure 2.9, they are fixed in place with a certain cant angle ϕ with respect to the symmetric axis of the lander. Therefore, the norm of the thrust vector $T_c(t)$ results to be

$$\|T_c(t)\|_2 = n_{eng}\|T_{single}(t)\|_2 cos(\phi) \tag{2.28}$$

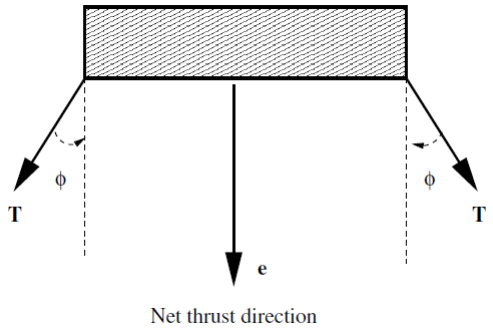


Figure 2.9: Thrusters configuration on the bottom of the spacecraft, where $e = k_{BRF}$ [7].

Every thrust vector T_{single} is bounded in magnitude above by the physical capability limitation of the rocket and below to avoid instabilities and turbulence in the ejected flux. Moreover, we need to leave some security margins on these values to be able later to apply accurately the control law for the attitude dynamics. Indeed, to obtain an overall torque action τ with the same thrusters without interfering with the overall thrust vector T_c, some rockets have to increase their thrust whereas others have to decrease to the same amount.
Therefore, the upper and lower bounds of T_c are defined as

$$\begin{aligned} \rho_{min} &= 0.3\ n_{eng} T_{max} cos(\phi) \\ \rho_{max} &= 0.8\ n_{eng} T_{max} cos(\phi) \end{aligned} \tag{2.29}$$

where $T_{max} = \max\ \|T_{single}\|_2$.

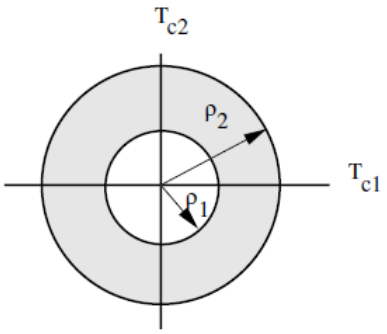


Figure 2.10: Representation of the thrust bounds $\rho_{max} = \rho_2$ and $\rho_{min} = \rho_1$ in a two dimensional case $T_c = \begin{bmatrix} T_{c1} \\ T_{c2} \end{bmatrix} \in \mathbb{R}^2$ [7].

- **Pointing angle [1] constraint**

$$T_c(t)^T \hat{e}_z \geq \|T_c(t)\|_2 cos(\gamma_p) \tag{2.30}$$

During the divert maneuver the lander cannot tilt itself too much to guarantee the correct behaviour of the radar. Indeed the radar, that is mounted on the bottom of the spacecraft, works exploiting the slant and Doppler measurements along 4 directions (one direction parallel to the symmetry axis and the other ones 20° skewed). The combination of singular slant input allows altitude determination. The system of the doppler measurements along the beam directions can be solved in the body reference frame R_{BRF} identifying the velocity of the lander, provided that at least 3 of the 4 measurements are available. If the lander is too sloped, one of them could stop working because the measured distance turns out to be too large, when this happens even to a second beam the radar cannot relay on enough beams and it is not able to work properly anymore.
Therefore, the thrust vector T_c has to be constrained in his attitude within a maximum angle $\gamma_p \in [0, \pi]$ from the vertical direction $\hat{e}_z = k_{ENU}$.

[1] **The pointing angle $\gamma = \angle(T_c, \hat{e}_z)$ is defined as the separation angle of the symmetry axis $k_{BRF}//T_c$ with respect to the local vertical $k_{ENU} = \hat{e}_z$**

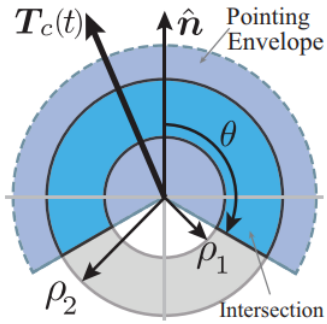


Figure 2.11: Representation of the feasible set of the thrust vector in a two dimensional case $T_c = \begin{bmatrix} T_{c1} \\ T_{c2} \end{bmatrix} \in \mathbb{R}^2$ as an intersection of the upper and lower bounds on the magnitude and the pointing angle constraint, where $\hat{n} = k_{ENU}$ is the vertical direction and $\gamma_p = \theta$ [8].

- **Glideslope constraint**

$$H_{gs} r(t) \leq h_{gs} \tag{2.31}$$

  During the descent toward the surface, it is essential to prevent collisions with the nearby terrain. Hence, we need to maintain the lander in a safe region that can be modelled as an intersection of a set of halfspaces centered on the target.

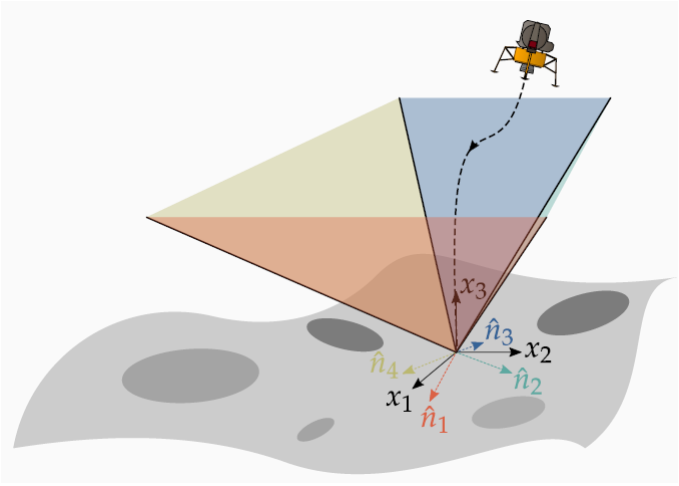


Figure 2.12: Representation of the glideslope constraint where 4 hyperplanes (with normal vectors $\hat{n}_1, \ldots, \hat{n}_4$) define a safe region where the lander can move through [9].

In this study, to define the safe region, we choose 4 hyperplanes, as in Figure 2.12, passing through the desired landing site $[0,0,0]^T$ in R_{ENU} with different inclinations. Each hyperplane has a normal vector related to the glideslope angle $\gamma_{gs} \in [0, \pi/2]$

$$\begin{aligned} \hat{n}_1^T &= [cos(\gamma_{gs}),0, -sin(\gamma_{gs})] \\ \hat{n}_2^T &= [0, cos(\gamma_{gs}), -sin(\gamma_{gs})] \\ \hat{n}_3^T &= [-cos(\gamma_{gs}),0, -sin(\gamma_{gs})] \\ \hat{n}_4^T &= [0, -cos(\gamma_{gs}), -sin(\gamma_{gs})] \end{aligned} \tag{2.32}$$

thus the H_{gs} matrix can be defined as

$$H = \begin{bmatrix} \hat{n}_1^T \\ \hat{n}_2^T \\ \hat{n}_3^T \\ \hat{n}_4^T \end{bmatrix} \tag{2.33}$$

eventually since all of them pass through the origin the offset term is equal to 0 for all of them.

$$h_{gs} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \tag{2.34}$$

- **Velocity constraint**

$$\|v(t)\|_2 \leq v_{max} \tag{2.35}$$

The velocity is constrained in magnitude to not exceed a security value v_{max}.

- **Mass constraint**

$$m_{dry} \leq m(t_f) \tag{2.36}$$

The lander will decrease monotonically his mass over the time due to the fuel consumption. This leads to the last constraint on the ending mass, that cannot be less than m_{dry}, that is, the mass of the lander without the fuel.

2.2.4 Initial and final conditions

To solve the problem, we need to impose initial conditions on the position $r(0) \in \mathbb{R}^3$ and the velocity $v(0) \in \mathbb{R}^3$ of the lander when it begins the divert maneuver, as well his initial mass $m(0) \in \mathbb{R}$

$$r(0) = r_0, \quad v(0) = v_0, \quad m(0) = m_{wet} = m_{dry} + m_{fuel} \tag{2.37}$$

Similarly, we need to define also the final position $r(t_f) \in \mathbb{R}^3$ and the final velocity $v(t_f) \in \mathbb{R}^3$

$$r(t_f) = r_N, \quad v(t_f) = v_N \tag{2.38}$$

where N identifies the last point of the landing trajectory subjected to the optimization and not necessarily the last point of the landing trajectory.

2.2.5 Optimization problem

At this point, we gather all the above information to cast them into an optimization problem. Whereas the constraints have been already defined in the previous section, we still need to define the optimization variable and the cost function.

As optimization variable we obviously need the thrust profile over the time $T_c(t),\ t \in [0, t_f]$, but also the time of flight $t_f \in \mathbb{R}$.

The cost function aims to reduce as much as possible the fuel consumption, thus we can formulate such function directly taking into account the mass or in a smarter way the thrust that is strictly correlated to the mass variation. In this way, we aims to minimize the thrust effort over the time, since it means also minimize the fuel consumption. Therefore, the cost function is defined as

$$\int_o^{t_f} \|T_c(t)\|_2 dt \tag{2.39}$$

The overall optimization problem can be assembled using the previously defined equations (2.23), (2.26), (2.27), (2.30), (2.31), (2.35), (2.36), (2.37), (2.38), (2.39)

$$\begin{aligned}
\min_{T_c, t_f} \quad & \int_o^{t_f} \|T_c(t)\|_2 dt \\
\text{subject to} \quad & \dot{r}(t) = v(t) \\
& \dot{v}(t) = g + \frac{T_c(t)}{m(t)} - \omega^{\times}\omega^{\times} r(t) - 2\omega^{\times} v(t) \\
& \dot{m}(t) = -\alpha \|T_c(t)\|_2 \\
& \rho_{min} \leq \|T_c(t)\|_2 \leq \rho_{max} \\
& T_c(t)^T \hat{e}_z \geq \|T_c(t)\|_2 cos(\gamma_p) \\
& H_{gs} r(t) \leq h_{gs} \\
& \|v(t)\|_2 \leq v_{max} \\
& m_{dry} \leq m(t_f) \\
& r(0) = r_0,\ v(0) = v_0,\ m(0) = m_{wet} \\
& r(t_f) = r_N,\ v(t_f) = v_N
\end{aligned} \tag{2.40}$$

It is easy to see that this optimization problem is not convex, indeed the equation (2.27) defines for the lower bound of the thrust a non convex set on the optimization

variable. To overcome this problem, we need to exploit the lossless convexification theory.

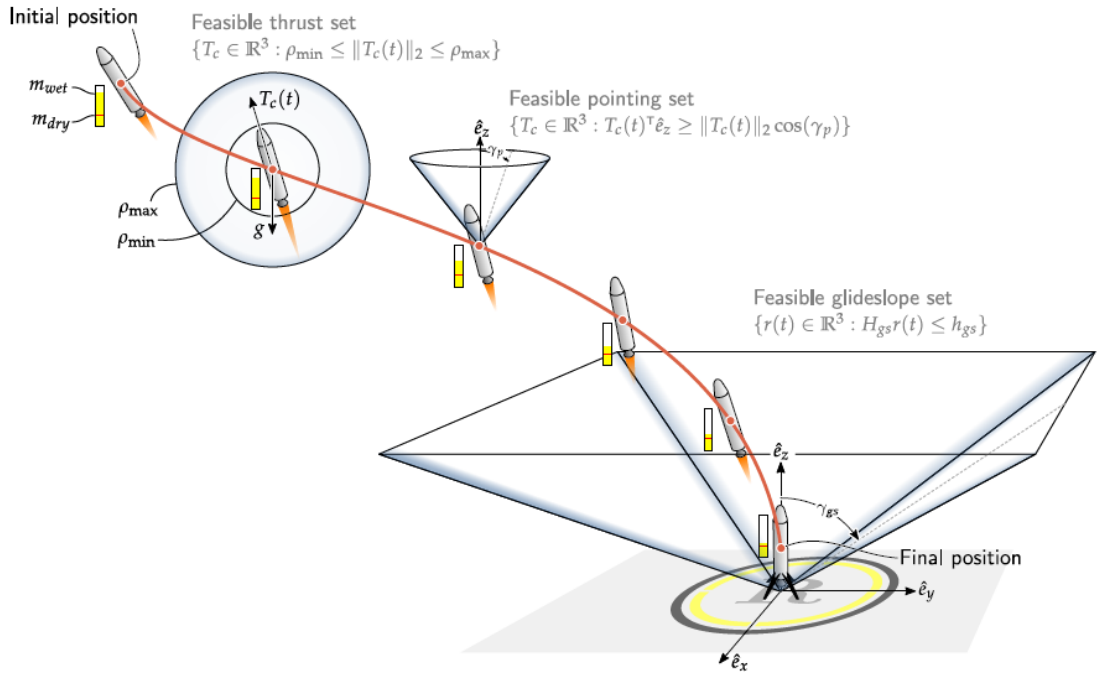


Figure 2.13: Graphical representation of the divert maneuver and his constraints [9].

2.3 Lossless convexification

The lossless convexification (LCvx) theory was developed by Açıkmeşe and Ploen in 2007 [7]. It deals with this specific kind of problem in order to convexify the equation (2.27). In order to overcome this problem, the LCvx theory introduces an augmented slack variable that leads to a new augmented convex problem. It can be proved that the optimal solution of the augmented problem coincides with the optimal solution of the original non convex problem.

Initially, it could deal only with the equation 2.27, but over the time the theory has been improved, more constraints were added to the optimization problem in order to address more general cases. Currently, it is still under development.

2.3.1 Convex relaxation of the input lower bound

The LCvx theory deals with the equation (2.27), more specifically with the lower bound of the thrust magnitude. Indeed, this constraint leads to a non convex feasible set (left image on the Figure 2.14). Thus, the theory introduces a new slack variable $\sigma \in \mathbb{R}$ such that

$$\|T_c(t)\|_2 \leq \sigma(t) \tag{2.41}$$

Thanks to this augmented optimization variable, the new feasible set (right image on Figure 2.14) results to be convex.

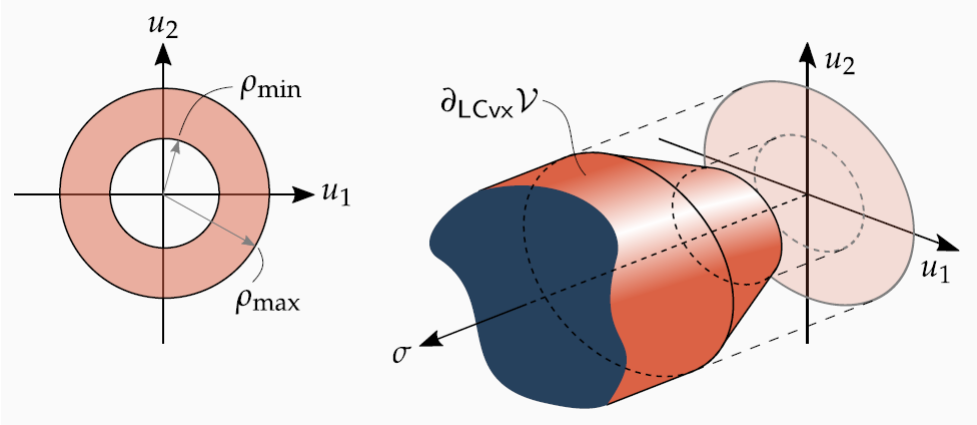


Figure 2.14: Feasible set due to the lower and upper bounds on the thrust vector $T_c = \begin{bmatrix} u_1 \\ u_2 \end{bmatrix} \in \mathbb{R}^2$ in the original problem on the left and in the augmented problem on the right [9].

The new slack variable replaces $\|T_c(t)\|_2$ in the cost function and in the constraints of the original optimization problem (2.40):

$$\begin{aligned}
\min_{\sigma,T_c,t_f} \quad & \int_0^{t_f} \sigma(t)dt \\
\text{subject to} \quad & \dot{r}(t) = v(t) \\
& \dot{v}(t) = g + \frac{T_c(t)}{m(t)} - \omega^{\times}\omega^{\times}r(t) - 2\omega^{\times}v(t) \\
& \dot{m}(t) = -\alpha\sigma(t) \\
& \rho_{min} \leq \sigma(t) \leq \rho_{max} \\
& \|T_c(t)\|_2 \leq \sigma(t) \\
& T_c(t)^T\hat{e}_z \geq \sigma(t)cos(\gamma_p) \\
& H_{gs}r(t) \leq h_{gs} \\
& \|v(t)\|_2 \leq v_{max} \\
& m_{dry} \leq m(t_f) \\
& r(0) = r_0,\ v(0) = v_0,\ m(0) = m_{wet} \\
& r(t_f) = r_N,\ v(t_f) = v_N
\end{aligned} \tag{2.42}$$

Since Problem (2.40) is a relaxation of Problem (2.42), a solution of the original problem is always feasible for the relaxed problem, but we cannot state the inverse. Pontryagin's maximum principle can be used to prove that the optimal solution of the Problem (2.42) $\{\sigma^*, T_c^*, t_f^*\}$ coincides with the optimal solution of the Problem (2.40) $\{T_c^*, t_f^*\}$ [7]. This is equivalent to say that at the optimal solution of the original problem (2.40) LCvx inequality (2.41) becomes the equality $\|T_c^*(t)\|_2 = \sigma^*(t)$.

Moreover, the obtained control action assumes a so called Bang Bang profile, where $\|T_c^*(t)\| = \rho_{min}$ or $\|T_c^*(t)\| = \rho_{max}$ for $t \in [0, t_f^*]$.

2.3.2 Dynamic linearization

Once the main source of non convexity has been treated, still remains to deal with the non linear dynamic, more specifically with the division $\frac{T_c(t)}{m(t)}$. We can make a change of variables to address this problem

$$\xi(t) = \frac{\sigma(t)}{m(t)}, \quad u(t) = \frac{T_c(t)}{m(t)}, \quad z(t) = ln(m(t)) \tag{2.43}$$

where $\xi \in \mathbb{R}$, $u \in \mathbb{R}^3$ and $z \in \mathbb{R}$.
The modified dynamic results to be

$$\begin{aligned} \dot{r}(t) &= v(t) \\ \dot{v}(t) &= g + u(t) - \omega^{\times}\omega^{\times}r(t) - 2\omega^{\times}v(t) \\ \frac{\dot{m}(t)}{m(t)} &= -\alpha\xi(t) \Rightarrow \dot{z} = -\alpha\xi(t) \end{aligned} \tag{2.44}$$

Then, we need to modify the cost function, since $\alpha > 0$ maximize $m(t_f)$ is equivalent to minimize

$$\int_0^{t_f} \xi(t)dt \tag{2.45}$$

It turns out that the new variables linearize all the constraints except the constraint on the upper bound of ξ, indeed the new inequality becomes

$$\rho_{min}e^{-z(t)} \leq \xi \leq \rho_{max}e^{-z(t)} \tag{2.46}$$

Moreover, we need to deal also with the lower bound, that is convex, but it doesn't fill into an SOCP, since it is an exponential cone. We decide to proceed linearizing both constraints by means of the Taylor series approximation

$$\begin{aligned} &\mu_{min}(t)[1 - \delta z(t) + \frac{1}{2}\delta z(t)^2] \leq \xi(t) \\ &\xi(t) \leq \mu_{max}(t)[1 - \delta z(t)] \end{aligned} \tag{2.47}$$

where

$$\begin{aligned} \mu_{min}(t) &= \rho_{min}e^{-z_0(t)} \\ \mu_{max}(t) &= \rho_{max}e^{-z_0(t)} \\ \delta z(t) &= z(t) - z_0(t) \\ z_0(t) &= ln(m_{wet} - \alpha\rho_{max}t) \end{aligned} \tag{2.48}$$

with $z_0(t)$ as lower bound of $z(t)$

$$z_0(t) \leq z(t) \leq ln(m_{wet} - \alpha\rho_{min}t) \tag{2.49}$$

Eventually we can state the final convex linearized optimization problem

$$
\begin{aligned}
\min_{\xi,u,t_f} \quad & \int_0^{t_f} \xi(t)dt \\
\text{subject to} \quad & \dot{r}(t) = v(t) \\
& \dot{v}(t) = g + u(t) - \omega^{\times}\omega^{\times}r(t) - 2\omega^{\times}v(t) \\
& \dot{z}(t) = -\alpha\xi(t) \\
& \mu_{min}(t)[1 - \delta z(t) + \frac{1}{2}\delta z(t)^2] \leq \xi(t) \\
& \xi(t) \leq \mu_{max}(t)[1 - \delta z(t)] \\
& \|u(t)\|_2 \leq \xi(t) \\
& u(t)^T \hat{e}_z \geq \xi(t) cos(\gamma_p) \\
& H_{gs} r(t) \leq h_{gs} \\
& \|v(t)\|_2 \leq v_{max} \\
& ln(m_{dry}) \leq z(t_f) \\
& z_0(t) \leq z(t) \leq ln(m_{wet} - \alpha\rho_{min}t) \\
& r(0) = r_0,\ v(0) = v_0,\ z(0) = ln(m_{wet}) \\
& r(t_f) = r_N,\ v(t_f) = v_N
\end{aligned}
\tag{2.50}
$$

2.4 Robustness approach to guarantee sub-optimal solutions

The solution provided by the literature does not focus on guarantee a reliable control action for real applications, rather it is a general approach to this kind of problem in order to show an implementation of the lcvx theory.

Therefore, the next step in this study in order to assure reliability of the software in real applications it is to provide an alternative control action to the unfeasible optimization problems, to assure the best sub optimal solution in such cases it is impossible to reach the target. To this aim, we need first to add a relaxation on the final constraint

$$r_z(t_f) = h_N \quad \text{instead of} \quad r(t_f) = r_N \tag{2.51}$$

and at the same time add an additional term on the cost function in order to minimize the final distance to the target

$$\left\| \begin{bmatrix} r_x(t_f) \\ r_y(t_f) \end{bmatrix} \right\|_2 \tag{2.52}$$

Finally also a slight modification on the glideslope constraint is needed

$$H_{gs} \left(r(t) - \begin{bmatrix} r_x(t_f) \\ r_y(t_f) \\ 0 \end{bmatrix} \right) \leq h_{gs} \quad \text{instead of} \quad H_{gs} r(t) \leq h_{gs} \tag{2.53}$$

The resulting optimization problem turn out to be

$$
\begin{aligned}
\min_{\xi,u,t_f} \quad & \int_0^{t_f} \xi(t)dt + \lambda \left\| \begin{bmatrix} r_x(t_f) \\ r_y(t_f) \end{bmatrix} \right\|_2 \\
\text{subject to} \quad & \dot{r}(t) = v(t) \\
& \dot{v}(t) = g + u(t) - \omega^{\times}\omega^{\times} r(t) - 2\omega^{\times} v(t) \\
& \dot{z}(t) = -\alpha \xi(t) \\
& \mu_{min}(t)[1 - \delta z(t) + \frac{1}{2}\delta z(t)^2] \leq \xi(t) \\
& \xi(t) \leq \mu_{max}(t)[1 - \delta z(t)] \\
& \|u(t)\|_2 \leq \xi(t) \\
& u(t)^T \hat{e}_z \geq \xi(t) cos(\gamma_p) \\
& H_{gs} \left(r(t) - \begin{bmatrix} r_x(t_f) \\ r_y(t_f) \\ 0 \end{bmatrix} \right) \leq h_{gs} \\
& \|v(t)\|_2 \leq v_{max} \\
& ln(m_{dry}) \leq z(t_f) \\
& z_0(t) \leq z(t) \leq ln(m_{wet} - \alpha \rho_{min} t) \\
& r(0) = r_0, \ v(0) = v_0, \ z(0) = ln(m_{wet}) \\
& r_z(t_f) = h_N, \ v(t_f) = v_N
\end{aligned}
\tag{2.54}
$$

where $\lambda \in \mathbb{R}$ is a weight parameter used to balance the influence of the two terms in the cost function.

2.5 Discretization

The optimization problems, as stated in (2.50),(2.54), are impossible to be solved, indeed their optimization variables are time continuous variables $\xi(t), u(t),\ t \in [0, t_f]$. This means, that the number of optimization variables to be implemented in the algorithm are actually infinite. This because, every variable in any time instant must be considered as an individual optimization variable of the algorithm. Therefore, it is needed to discretize the problem to reduce the optimization variables to a finite number. The problem obtained in this way is not exactly the same of his continuous counterpart, but it is a good approximation, that becomes more and more accurate as the sample time is reduced.

The time interval $[0, t_f]$ has been divided into equidistant time intervals of the duration Δt, such that $t_k = k\Delta t,\ k = 0, \ \dots\ , N$, with $N = \left\lceil \frac{t_f}{\Delta t} \right\rceil$.

In this study, the zero-order-hold (ZOH) method has been used as simple and reliable way to discetize the dynamics (2.44). It leads to $N+1$ discrete optimization variables for each state variable and N for each input variable. Then there will be:

- $r \in \mathbb{R}^3 \Rightarrow 3(N+1)$ variables
- $v \in \mathbb{R}^3 \Rightarrow 3(N+1)$ variables
- $z \in \mathbb{R} \Rightarrow N+1$ variables
- $u \in \mathbb{R}^3 \Rightarrow 3N$ variables
- $\xi \in \mathbb{R} \Rightarrow N$ variables

Therefore, the whole discretized optimization problem counts $11N+7$ optimization variables.

Alternatively to the ZOH, other more complex methods could be used to reduce this number. For example, we could exploit the Chebyshev polynomials as basis functions instead of the piecewise constant basis functions used in this case.

The dynamics of the spacecraft, described by the equations (2.44), can be reformulated in state space form as

$$\dot{x}(t) = A_c x(t) + B_c \tilde{u}(t) + E_c w(t) \tag{2.55}$$

where $x(t) = [r(t), v(t), z(t)]^T \in \mathbb{R}^7$ is the state vector, $\tilde{u}(t) = [u(t), \xi(t)]^T \in \mathbb{R}^4$ is the input vector, $w = g \in \mathbb{R}^3$ is the additive exogenous disturbance and the state space matrices are

$$A = \begin{bmatrix} 0_3 & I_3 & 0 \\ -\omega^\times\omega^\times & -2\omega^\times & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0_3 & 0 \\ I_3 & 0 \\ 0 & -\alpha \end{bmatrix}, \quad E = \begin{bmatrix} 0_3 \\ I_3 \\ 0 \end{bmatrix} \tag{2.56}$$

Thus, in order to represent the discrete time dynamics as

$$x(k) = Ax(k-1) + B\tilde{u}(k-1) + Ew, \quad k = 1, \ \dots \ , N \tag{2.57}$$

we need to compute the discrete time state space matrices as

$$\begin{aligned} A &= e^{A_c \Delta t} \\ B &= \int_0^{\Delta t} e^{A_c(\Delta t - s)} B_c \ ds \\ E &= E_c \end{aligned} \tag{2.58}$$

Therefore, the optimization problem (2.50) discretized becomes

$$\begin{aligned} \min_{\xi, u, t_f} \quad & \sum_{k=0}^{N} \xi(k) \Delta t \\ \text{subject to} \quad & x(k) = Ax(k-1) + B\tilde{u}(k-1) + Ew, \quad k = 1, \ \dots \ , N \\ & \mu_{min}(k)[1 - \delta z(k) + \frac{1}{2}\delta z(k)^2] \leq \xi(k), \quad k = 0, \ \dots \ , N-1 \\ & \xi(k) \leq \mu_{max}(k)[1 - \delta z(k)], \quad k = 0, \ \dots \ , N-1 \\ & \|u(k)\|_2 \leq \xi(k), \quad k = 0, \ \dots \ , N-1 \\ & u(k)^T \hat{e}_z \geq \xi(k) cos(\gamma_p), \quad k = 0, \ \dots \ , N-1 \\ & H_{gs} r(k) \leq h_{gs}, \quad k = 0, \ \dots \ , N \\ & \|v(k)\|_2 \leq v_{max}, \quad k = 0, \ \dots \ , N \\ & ln(m_{dry}) \leq z(N) \\ & z_0(k) \leq z(k) \leq ln(m_{wet} - \alpha \rho_{min} t_k), \quad k = 0, \ \dots \ , N \\ & r(0) = r_0, \ v(0) = v_0, \ z(0) = ln(m_{wet}) \\ & r(N) = r_N, \ v(N) = v_N \end{aligned} \tag{2.59}$$

Whereas, the optimization problem (2.54) discretized becomes

$$
\begin{aligned}
\min_{\xi,u,t_f} \quad & \sum_{k=0}^{N} \xi(k)\Delta t + \lambda \left\| \begin{bmatrix} r_x(N) \\ r_y(N) \end{bmatrix} \right\|_2 \\
\text{subject to} \quad & x(k) = Ax(k-1) + B\tilde{u}(k-1) + Ew, \quad k = 1, \ \dots \ , N \\
& \mu_{min}(k)[1 - \delta z(k) + \frac{1}{2}\delta z(k)^2] \leq \xi(k), \quad k = 0, \ \dots \ , N-1 \\
& \xi(k) \leq \mu_{max}(k)[1 - \delta z(k)], \quad k = 0, \ \dots \ , N-1 \\
& \|u(k)\|_2 \leq \xi(k), \quad k = 0, \ \dots \ , N-1 \\
& u(k)^T \hat{e}_z \geq \xi(k) cos(\gamma_p), \quad k = 0, \ \dots \ , N-1 \\
& H_{gs} \left(r(k) - \begin{bmatrix} r_x(N) \\ r_y(N) \\ 0 \end{bmatrix} \right) \leq h_{gs}, \quad k = 0, \ \dots \ , N \\
& \|v(k)\|_2 \leq v_{max}, \quad k = 0, \ \dots \ , N \\
& ln(m_{dry}) \leq z(N) \\
& z_0(k) \leq z(k) \leq ln(m_{wet} - \alpha \rho_{min} t_k), \quad k = 0, \ \dots \ , N \\
& r(0) = r_0, \ v(0) = v_0, \ z(0) = ln(m_{wet}) \\
& r_z(N) = h_N, \ v(N) = v_N
\end{aligned}
\tag{2.60}
$$

Chapter 3

Algorithm description

In this chapter, the algorithm used to solve numerically the optimization problems for pinpoint landing stated in the previous chapter is described. From the literature, has been taken as reference the already implemented code presented in [9].

The algorithm presents more levels of abstraction. The core of the algorithm remains the implementation of the SOCP. Once chosen a suitable programming language, we can exploit an already existing solver to compute the solution. They are algorithms studied specifically to solve convex optimization problems with the most efficient techniques. Unfortunately, every solver has his own interface, which therefore would require to change again the shape of the optimization problem, to be compliant with the predefined shape required by the solver. For this reason, come in handy to exploit a software called parser, that take care of this reshaping. The outer layer of the algorithm is dedicated to the computation of the optimal time of flight. Indeed, this optimization variable needs to be fixed a priori. A search algorithm evaluates and compares the solutions of the optimization problem for different values of time of flight, in order to find the best solution.

Finally, the optimal control action is computed from the result of the described algorithm, as well an accurate simulation of the dynamics of the spacecraft during the divert maneuver.

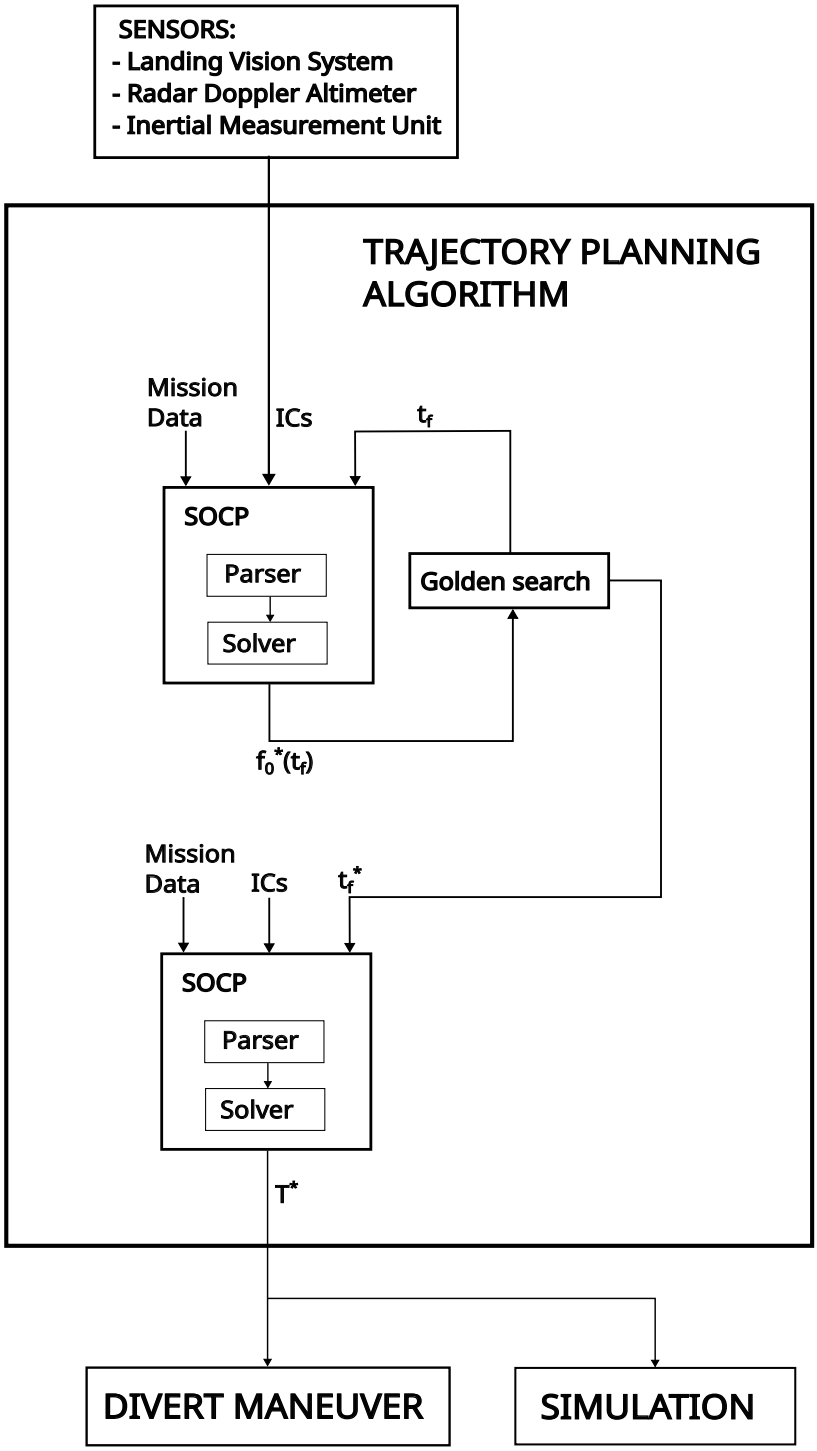


Figure 3.1: Scheme of the trajectory planning algorithm for the divert maneuver.

3.1 Implementation of the optimization problem

First of all, it is necessary to choose a suitable programming language for the algorithm. Here, it has been chosen Matlab, that provides a convenient environment where develop and debug the code, as well a set of already implemented solvers and parsers. Moreover, from the Matlab environment we can later automatically translate the code into other programming languages more suitable for implementation on real devices, e.g. C language.

3.1.1 Solver

The discretized optimization problems (2.59)(2.60) can be solved in the Matlab framework exploiting a set of algorithms called solvers, that have been already implemented in this environment. They are algorithms tailored to deal with optimization problems. There are different products among which choose, that have different capabilities, reliability and performances. Among the open source solvers distributed for the Matlab environment need to be mentioned SDPT3, SeDuMi and ECOS.

In this study, I have chosen SDPT3 solver, since although it is not the fastest solver between those mentioned, it results to be the most reliable during the simulations, with a computational time comparable to the others. In detail, SDPT3 version 4.0, as reported in the article [10], is an algorithm able to solve primal dual semidefinite-quadratic-linear conic problems, whose constraints cone is a product of semidefinite cones, second-order cones, non negative orthants and Euclidean spaces, thus including also the problem under study, that is an SOCP. It exploits an infeasible primal-dual predictor-corrector path-following method, with either the HKM or the NT search direction. This method belongs to the set of algorithms called Interior Point Method (IPM), used to efficiently solve numerical constrained optimization problems, with in this case additional features like predictor-corrector path following and different search directions. A detailed explanation of IPM is beyond the scope of this thesis work. For further details about IPM refer to [11] [12].

3.1.2 Parser

To solve the problem directly by means of a previously mentioned solver, the optimization problem must be rewritten to comply with the input structure of the solver. In the case of SDPT3 the problem must assume the structure of a

semidefinite program (SDP)

$$\begin{aligned} \min_{x} \quad & c^T x \\ \text{subject to} \quad & A(x) = b \\ & F(x) \succeq 0 \end{aligned} \tag{3.1}$$

Thus, to use the solver directly we need to provide it with the vectors and matrices c, A, b, F, in such a manner that the problem (3.1) results to be equivalent to the problems (2.59)(2.60).

Fortunately, there are some tools called parser that perform this operation for us, we just need to write the optimization problem respecting some coding rules and let the parser provides to the solver the data suitably cast into the desired form. These tools allow us to code the optimization problem in a readable way. Unfortunately this abstraction level implies a slight slower program, thus the use of this kind of tools is justifiable during a preliminary design phase, whereas probably they should be avoided in the implementation phase of the software on a real device.

In the Matlab framework the most used parser is CVX, a detailed documentation can be found in [13]. CVX allow us to choose a desired solver between some open source solutions provided together with the CVX distribution itself, such as SDPT3 and SeDuMi, and others that can be separately downloaded and attached to the parser, both free solutions like ECOS and commercial ones. Then we need to describe the optimization variables, as well their dimensions, the cost function and the constraints, exactly as they are written in the problems (2.59)(2.60). At the end, we will obtain the value of the optimal cost function and the corresponding values of the optimal variables, together with some specific details regarding the computation of the solution during every iteration.

3.2 Golden search

We observe that the problems (2.59)(2.60) cannot be solved without fixing a priori the value of t_f. Indeed, if t_f was considered as a variable of the problem, it would be a source of non convexity that cannot be eliminated. Therefore we need to find the optimal time of flight t_f^* outside of the optimization problem.

A search algorithm for t_f^* has been implemented. The basic idea behind it consists in comparing iteratively the optimal solution f_0^* of the optimization problem for different values of $t_f = \{t_f^1, t_f^2, \ \dots \ , t_f^N\} \in [t_{min}, t_{max}]$, searching for the value t_f^* that minimize f_0^*. The sequence of t_f has to be chosen in order to guarantee the convergence of the solution and a final accuracy compliant with a required tolerance.

The values t_{min}, t_{max} are respectively the minimum and maximum feasible value of the time of flight.
The minimum feasible time of flight is computed as the time taken by the spacecraft to reach the ground with the maximum command effort and without the fuel weight

$$t_{min} = \frac{m_{dry} * \|v_0\|_2}{\rho_{max}} \tag{3.2}$$

Instead the maximum feasible time of flight is computed as the time taken by the spacecraft to deplete all the fuel with the minimum command effort

$$t_{max} = \frac{m_{wet} - m_{dry}}{\alpha \rho_{min}} \tag{3.3}$$

The golden search function has been chosen as search algorithm. It is used to find the minimum of an unimodal function $f(x) : \mathbb{R} \to \mathbb{R}$ between two extreme points x_{min}, x_{max}.
During every step are defined 4 points a, b, c, d, as represented in Figure 3.2, such that

$$\begin{aligned}
&\frac{I_{TOT}}{I_C} = \frac{I_C}{I_c} = \phi, \quad I_C = |b-c| > I_c = |c-a|, \quad I_{TOT} = I_C + I_c = |b-a| \\
&\frac{I_{TOT}}{I_D} = \frac{I_D}{I_d} = \phi, \quad I_D = |d-a| > I_d = |b-d|, \quad I_{TOT} = I_D + I_d = |b-a| \\
&\phi = \frac{1+\sqrt{5}}{2} \quad \text{golden ratio}
\end{aligned} \tag{3.4}$$

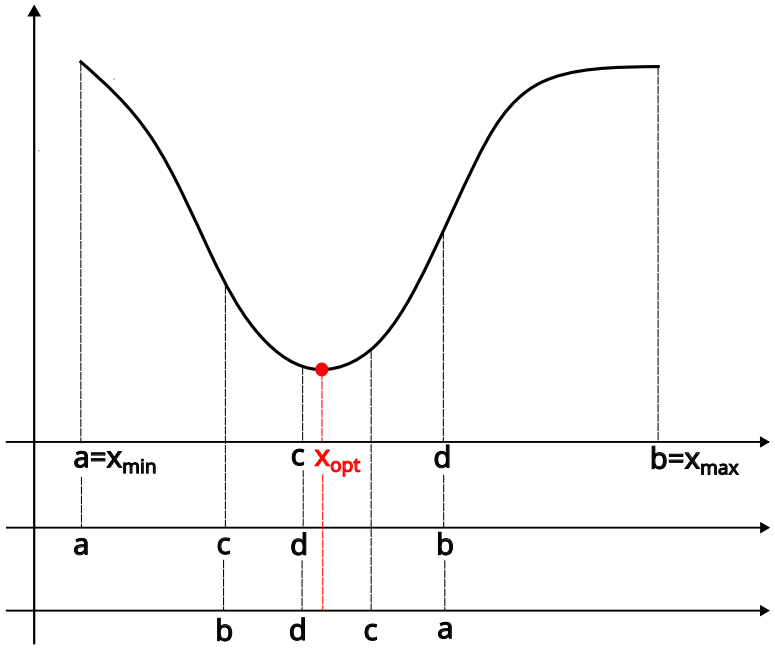


Figure 3.2: First 3 iterations of the golden search, in an example case.

The algorithm begins assigning $a = x_{min},\ b = x_{max}$ and computing c, d according to the rule (3.4).

Then, during every iteration $f(c)$ and $f(d)$ are compared and depending on whether or not $f(c) < f(d)$ new 4 points a, b, c, d are defined. The points a, b, c are reassigned to narrow the interval I_{TOT} around the solution, whereas d is computed again every iteration, accordingly to the rule (3.4).

The number of the iterations is computed a priori, in order to obtain a final accuracy of the solution with a required tolerance tol. We observe that at the n-th iteration the residual uncertainty interval, within we can find the solution, is $I_n = \frac{I_{n-1}}{\phi} = \frac{I_0}{\phi^{n-1}}$, where $I_0 = |x_{max} - x_{min}|$. Thus, to obtain at the end $I_N \leq tol$, the number of iterations N has to be at least

$$N = \left\lceil \frac{ln(\frac{x_{max}-x_{min}}{tol})}{ln(\phi)} + 1 \right\rceil \tag{3.5}$$

The algorithm implementation is based on [14].

```
function [x_sol,y_sol] = golden(f,a,b,tol)
    phi = (1+sqrt(5))/2;
    n = ceil(log((b-a)/tol)/log(phi)+1);
    rho = phi-1;
    d = rho*b+(1-rho)*a;
    yd = f(d);
    for i = 1:n-1
        c = rho*a+(1-rho)*b;
        yc = f(c);
        if yc<=yd
            b = d;
            d = c;
            yd = yc;
            x_sol = c;
            y_sol = yc;
        else
            a = b;
            b = c;
            x_sol = d;
            y_sol = yd;
        end
    end
end
```

In this case of study, $f(\cdot) = f_0^*(t_f)$ is the solution of the optimization problem depending on the time of flight, $a = t_{min}$ is the minimum feasible time of flight, $b = t_{max}$ is the maximum feasible time of flight and tol is the tolerance on the accuracy of the solution. Finally, $x_{sol} = t_f^*$ is the optimal time of flight, $y_{sol} = f_0^*(t_f^*)$ is the solution of the optimization problem with the optimal time of flight.

3.3 Unfeasible optimization problem

The golden search usually ends with an optimal time of flight t_f^* and the respective solution of the optimization problem. But, sometimes could happens that for every call of the the golden search the optimization problem (2.59) to be solved results to be unfeasible, in these cases no exact solution is found. The divert maneuver is impossible to be accomplished with the specified requirements. In these cases, we need to provide a sub optimal solution for the maneuver, to guarantee a safe touch down of the lander. As described in section 2.4, we are no longer capable to reach the target, then we settle for a final position as near as possible to the desired landing site. Hence, we replace the optimization problem (2.59) with the problem (2.60) and we run again the previously described algorithm, as represented in Figure 3.3. In this way, we obtain a sub optimal solution of our original optimization problem, that is a compromise between the necessity to safely land the spacecraft on the Mars surface, to minimize the fuel consumption and to minimize the final distance to the target.

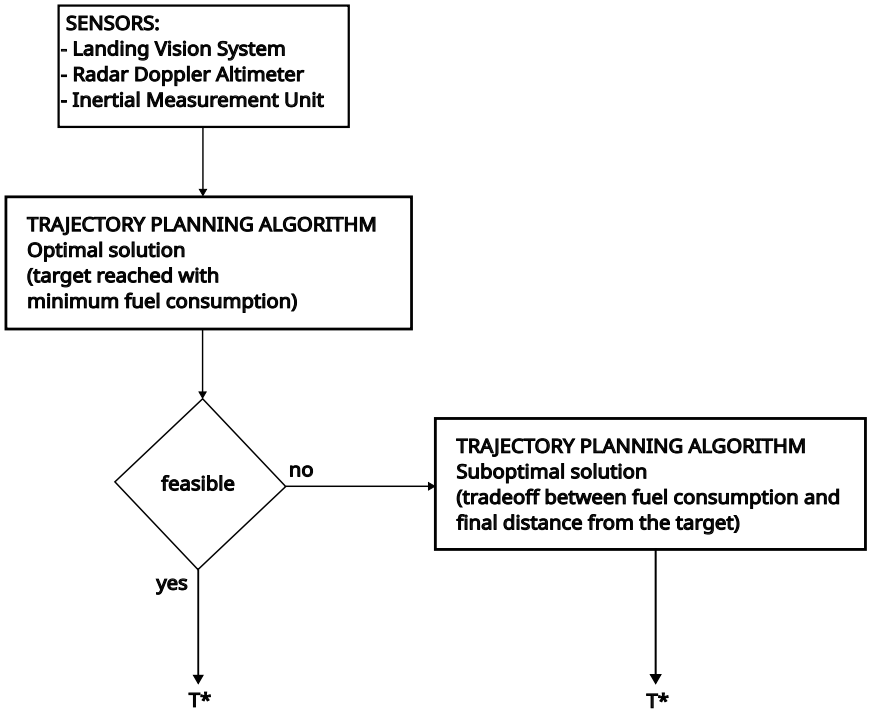


Figure 3.3: Scheme of the trajectory planning algorithm, with resolution of unfeasible optimization problems.

3.4 Solution and simulations

Once the optimization problem is solved, the optimal control sequence can be computed as

$$T_c^*(k) = m^*(k)u^*(k) = e^{z^*(k)}u^*(k), \ k = 0, \ \dots \ , N-1 \tag{3.6}$$

where $m^*(k), u^*(k), z^*(k)$ are the optimal variable solutions of the optimization problem.

Thus, we can retrieve the continuous control action by means of the ZOH method, keeping constant the value computed at the instant t_k until t_{k+1}

$$T_c^*(t) = T_c^*(k), \quad t_k \leq t < t_{k+1} \tag{3.7}$$

This is sufficient for thrust control purposes.

Instead, if we want to simulate accurately the behaviour of the dynamics variables $r(t), v(t), z(t)$, $t = [0, t_f^*]$ we cannot simply apply the ZOH method to the results of the optimization problem $r^*(k), v^*(k), z^*(k)$, $k = 0, \ \dots \ N$. We need a more accurate simulation of their behaviour, as represented in Figure (3.4).

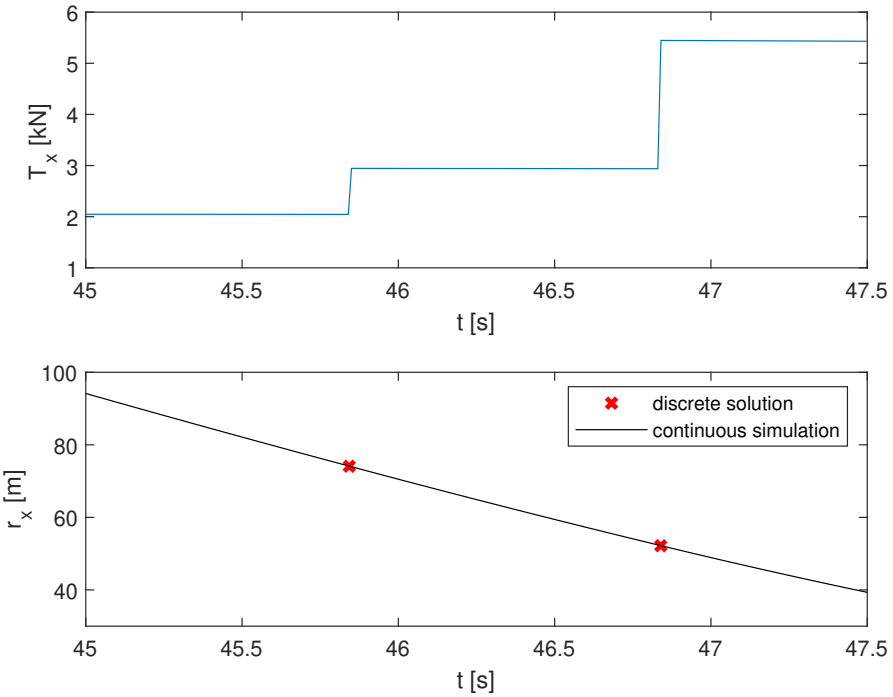


Figure 3.4: Simulation of the dynamics of $r_x(t)$ between 2 discrete solutions of the optimization problem, in an example case.

To this aim, we numerically integrate the differential equations of the dynamics (2.23)(2.26) with the rk4 method (the Runge-Kutta method).

Given an initial value $x(t_0) = x_0$ and a non linear differential equation $\frac{dx}{dt} = f(t, x),\ t \in [0, t_f]$ the rk4 method computes the discrete approximation of the unknown function $x(t_n)$ over the time instants $t_n = n\Delta t_{sim},\ n = 0,\ \dots\ , N_{sim}$, with $N_{sim} = \left\lceil \frac{t_f}{\Delta t_{sim}} \right\rceil$. Where $\Delta t_{sim} << \Delta t$ and $N_{sim} >> N$, to achieve a more accurate resolution than the optimization problem's result.

This method computes iteratively every value x_{n+1} from the previous value x_n plus an incremental term Δx. Where the incremental term Δx is composed by the time interval Δt_{sim} times a weighted slope of the function. Indeed, k_1 is the slope at the beginning of the interval, k_2 is the slope in the midpoint using k_1 to approximate x_n in the midpoint, k_3 is again the slope in the midpoint that use k_2 to approximate x_n and finally k_4 is the slope at the end of the interval using k_3 to approximate x_n at the end of the interval.

$$
\begin{aligned}
x_{n+1} &= x_n + \Delta x \\
t_{n+1} &= t_n + \Delta t_{sim} \\
\Delta x &= \frac{k_1 + 2k_2 + 2k_3 + k_4}{6} \Delta t_{sim} \\
k_1 &= f(t_n, x_n) \\
k_2 &= f(t_n + \frac{\Delta t_{sim}}{2}, x_n + \frac{k_1}{2} \Delta t_{sim}) \\
k_3 &= f(t_n + \frac{\Delta t_{sim}}{2}, x_n + \frac{k_2}{2} \Delta t_{sim}) \\
k_4 &= f(t_n + \Delta t_{sim}, x_n + k_3 \Delta t_{sim})
\end{aligned}
\tag{3.8}
$$

Finally, we observe that if both the optimization problem and the simulation have been solved correctly, the solution of the optimization problem will coincide with the simulation results in the discretized time instants $t_k = k\Delta t,\ k = 0,\ \dots\ , N$, as shown in Figure 3.4.

Chapter 4

Optimization of the execution time

In this chapter, the parameter's tuning of the optimal trajectory planning algorithm presented in the previous chapter is described.

To compute the optimal solution numerically, we need to solve several times a large optimization problem with thousands of optimization variables. Hence, the algorithm results to be computationally heavy. Since it is used in real-time online applications, a trade-off between the accuracy of the solution and the computational performance is required [1].

The parameters that most influence these factors are the sample time of the discretized optimization problem and the tolerance of the golden search. Indeed, the sample time is related to the dimension of the optimization problem, whereas the tolerance of the golden search suggests how many times the optimization problem must be repeated.

---

[1] In the Mars landing sequence the trajectory optimization algorithm is devoted to the planning of the so-called divert maneuver aimed to recover a large horizontal error in a limited altitude range. This operation may be executed once the navigation solution, based on Landing Vision System (LVS), Radar Doppler Altimeter (RDA) and Inertial Measurement Unit (IMU) has found the relative position of the lander with respect to the landing target. On the basis of the available assumptions from JPL, the navigation takes not less than 10 s to achieve this objective starting from an altitude close to 4000 m. In addition, it is needed to account about 5 s of additional time in advance to the execution of the divert maneuver, to allow the spacecraft achieving the initial divert attitude. Considering 15 s of total time and a conservative vertical velocity under parachute of 120 m/s, this correspond to an altitude loss of 1800 m. The resulting altitude coming out from this identified difference, 2200 m, is close to the altitude in which the divert maneuver must initiate to permit the recovery of horizontal errors up to 3 km. From this raw computation it comes immediately clear the paramount importance of minimizing the execution time of the trajectory planning algorithm.

4.1 Process step

As described in section 2.5, the sample time Δt divides the time interval $[0, t_f]$ in N equidistant time instants. Hence, the optimization variables of the discretized optimization problems (2.59)(2.60) are

- $r(k) \in \mathbb{R}^3, \quad k = 0, \ \dots \ , N$
- $v(k) \in \mathbb{R}^3, \quad k = 0, \ \dots \ , N$
- $z(k) \in \mathbb{R}, \quad k = 0, \ \dots \ , N$
- $u(k) \in \mathbb{R}^3, \quad k = 0, \ \dots \ , N-1$
- $\xi(k) \in \mathbb{R}, \quad k = 0, \ \dots \ , N-1$

Then, the sample time Δt and the time of flight t_f define the overall number of the optimization variables and thus the dimension of the optimization problem.

$$N = \left\lceil \frac{t_f}{\Delta t} \right\rceil \Rightarrow \ 11N + 7 \quad \text{optimization variables} \tag{4.1}$$

Whereas, the time of flight t_f is a parameter defined inside the golden search, the sample time Δt is a free parameter that can be tuned.

Discretize the original optimization problem (2.50) let us solve it numerically, but lead also to an approximation of the optimal solution. This approximation is as much accurate as Δt is small.

Thus, decreasing Δt will lead to an heavier but more accurate optimization problem, whereas increasing Δt let us solve the optimization problem more quickly but with worst results.

To tune Δt, it is necessary to run several times the optimal trajectory planning algorithm, keeping constant all the parameters and the initial conditions, except the sample time Δt. Then, to compare the computational performance of the algorithm I measured the computational time t_c taken to find the solution. Whereas, to compare the accuracy of the solution I considered the computed fuel consumption at the end of the divert maneuver $m_{fc}(N)$.

The values of Δt for the comparison are taken around 1 s, that is the default value used in [9].

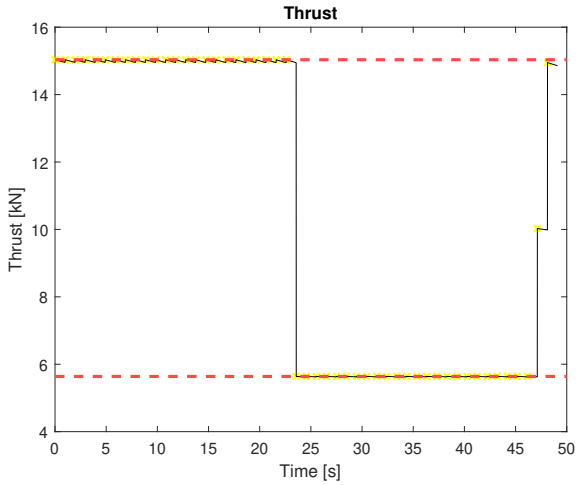
	t_c [s]	$m_{fc}(N)$ [kg]
$\Delta t = 0.5$ s	150.91	232.98
$\Delta t = 1$ s	78.6	233.02
$\Delta t = 2$ s	41.2	233.13
$\Delta t = 4$ s	24.32	233.69

Table 4.1: Comparison results of the optimal trajectory planning algorithm with different Δt values, in an example case.

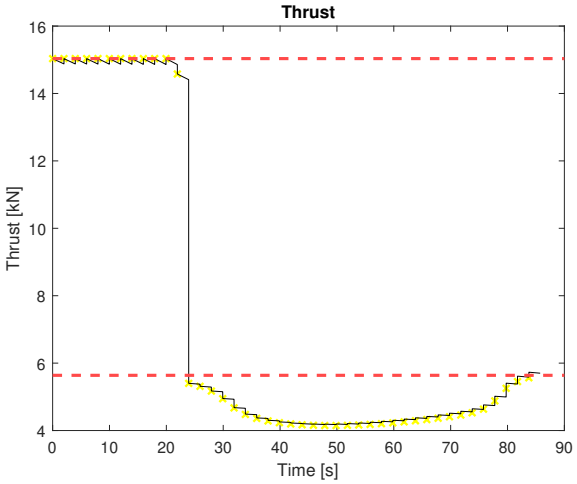
As we can see from the results summarized in table 4.1, the choice of the sample time Δt has a great influence on the computational time t_c, with an almost linear relationship. Whereas, the computed fuel consumption $m_{fc}(N)$ at the end of the maneuver almost does not change at all varying Δt.

Hence, it seems that we can increase significantly the value of Δt without losing accuracy of the solution. Unfortunately, other problems arise when we increase too much the value of Δt.

We note that the LCvx theory guarantees the resolution of the original optimization problem (2.40) only solving the modified continuous optimization problem (2.50), not his discretized approximation (2.59). The most of the time, the approximation is still fine to solve correctly the original optimization problem. But sometimes, when there are some critical initial conditions, could happen that the discretized optimization problem computes an unfeasible solution for the original optimization problem, as represented in Figure 4.1. In these cases, a thicker discretization is needed.



(a) $\Delta t = 1$ s



(b) $\Delta t = 2$ s

Figure 4.1: Solutions of the same optimization problem, with different values of Δt. The optimization problem has a critical initial velocity $\|v_0\|_2 = 161$ m/s. Lower and upper bounds on the thrust magnitude are represented with red dashed lines.

Moreover, we observe that a bigger sample time Δt requires a faster controller, to track precisely the discontinuous thrust profile.

In conclusion, the sample time Δt has to be reduced for the reasons just mentioned. For this thesis work it has been chosen $\Delta t = 1$ s, as satisfying trade-off between speed and reliability of the algorithm.

4.2 Golden search tolerance

As described in section 3.2, the number of calls of the optimization problem during the golden search is related to the required tolerance *tol* of the solution and the extreme values of feasible time of flight t_{min}, t_{max}.

$$N = \left\lceil \frac{ln(\frac{t_{max}-t_{min}}{tol})}{ln(\phi)} + 1 \right\rceil \tag{4.2}$$

where ϕ is the golden ratio.

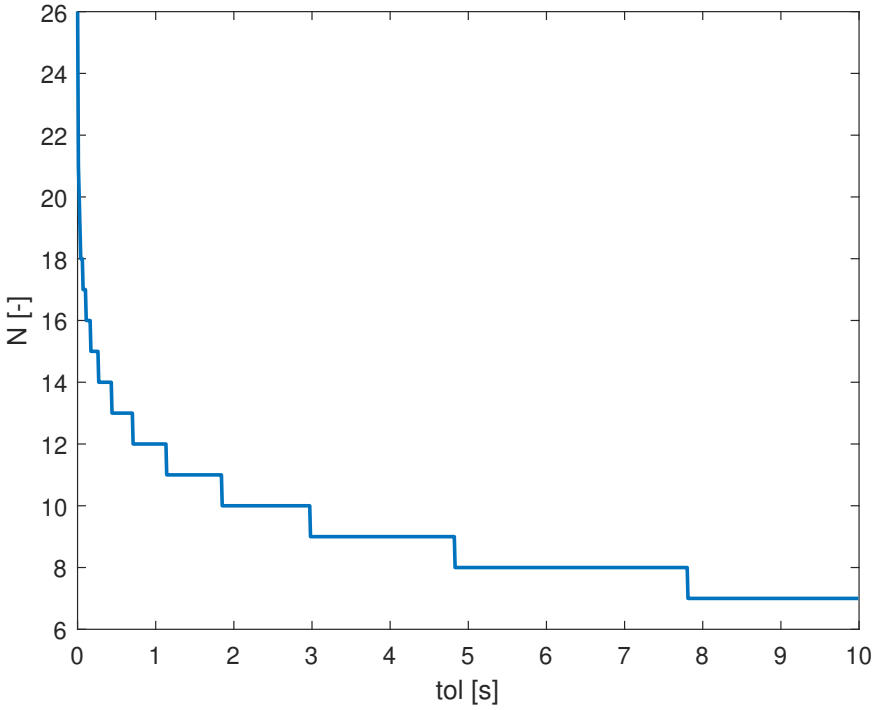


Figure 4.2: Non linear relationship between N and *tol*, in a case example with $t_{max} = 150$ s, $t_{min} = 10$ s.

Since every optimization problem needs a not negligible computational time to be solved, we need to take the number of calls as low as possible, guaranteeing at the same time enough accuracy of the solution. Hence, a trade-off between accuracy of the solution and computational performances is needed also here. Whereas, the values of t_{max}, t_{min} are computed by (3.2),(3.3) and cannot be modified, the tolerance *tol* is a free parameter that can be tuned.

Similarly as done in the previous section, to tune *tol* it is necessary to run several times the optimal trajectory planning algorithm, keeping constant all the parameters and the initial conditions, except the tolerance *tol*. As before, the computational performance of the algorithm is compared measuring the computational time t_c taken to find the solution. Whereas, the accuracy of the solution is compared considering the computed fuel consumption at the end of the divert maneuver $m_{fc}(N)$.

The values of *tol* for the comparison are taken starting from 0.001 s, that is the default value used in [9].

	t_c [s]	$m_{fc}(N)$ [kg]
$tol = 0.001$ s	161.73	233.01
$tol = 1$ s	80.98	233.01
$tol = 2$ s	68.15	233.04
$tol = 3$ s	62.99	233.04
$tol = 4$ s	60.74	233.04

Table 4.2: Comparison results of the optimal trajectory planning algorithm with different *tol* values, in an example case.

From the table 4.2, we can observe that increasing the value of *tol* we can decrease the computational time t_c, keeping almost constant the fuel consumption $m_{fc}(N)$. Anyhow, the relationship between *tol* and t_c is not linear, for bigger values of *tol* the improvements on t_c are smaller.

To maintain a reasonable tolerance on the golden search solution and decrease the computational time as long as there is a significant variation on the computational performance, it has been chosen $tol = 3$ s.

As a final remark, we observe that in case of an unfeasible problem the algorithm performs again the golden search looking for the best sub optimal solution and therefore using twice the computational time.

Chapter 5

Case studies of meaningful profiles

In this chapter, several landing maneuvers are presented and analysed, in order to test the results of the trajectory planning algorithm for pinpoint landing, described in the previous chapters.

First, single maneuver tests are performed and their results are analysed, such as the trajectory, the attitude profile, the fuel consumption and the optimal thrust command profile. Both feasible and unfeasible cases are considered, with respectively optimal and sub optimal solutions.

Then, a Monte Carlo experiment is used to analyse the general behaviour of the algorithm. The reliability of the algorithm is tested sounding out a wide variety of cases, within nominal working conditions.

5.1 Initialization of the parameters

The parameters to be defined in the algorithm can be divided in mission parameters and initial conditions. Mission parameters have to be defined at priori and concern the desired landing site and the spacecraft features. Whereas, the initial conditions of the divert maneuver are measured by the sensors on the spacecraft, such as the landing vision system, the radar doppler altimeter and the inertial measurement unit. They concern the position and velocity of the lander at the beginning of the maneuver.

Below are defined the mission parameters, chosen to be as similar as possible to a real case. Whereas, since the initial conditions are specific to each maneuver, they will be reported later along with every case.

Landing site

We need to know the position and the orientation of the desired landing site, to define the reference frame R_{ENU} and the relative positions and velocities.

Moreover, we also need the latitude of the landing site θ to compute the Mars angular velocity ω in R_{ENU}, as described in equation (2.25), to correctly set up the dynamic of the optimization problem.

- $\theta = 30° \quad \Rightarrow \quad \omega = [0.6139,0,0.3544]^T 10^{-4}$ rad/s

Final conditions

The divert maneuver, topic of this thesis, it is not the last phase in the landing procedure. Whereas, it allows to recover large horizontal errors at the end of the descending phase, it does not dealt with the touch down on the Mars surface. That is a delicate maneuver, which needs to be completed accurately in a following separate phase.

Therefore, the final conditions of the divert maneuver are chosen in such a way to prepare the lander for the touch down. The final position of the spacecraft after the divert maneuver will be an elevated point above the desired landing site, with a non zero final velocity directed downwards. Indeed, we need to leave enough space to softly slow down the lander during his approach to the Mars surface.

- $h_N = 300$ m
- $v_N = [0,0,-25]^T$ m/s

Spacecraft features

The parameters of the spacecraft have been assumed on the basis of the following considerations:

1. A dry mass of the landing platform similar to the expected one in the Rosalind Franklin mission
2. A conservative level of the specific impulse and of the thrust level of each engine
3. A cant angle sufficient to provide contributions to the roll control and to fine lateral translation control

Thus, the chosen values are

- $m_{dry} = 1100$ kg
- $m_{wet} = 1500$ kg
- $Isp = 205$ s
- $n_{eng} = 6$
- $\phi = 20°$
- $\alpha = 5.293310^{-4}$ s/m
- $T_{max} = 2.5$ kN

Constraints

Finally, we need to define the values of the constraints used in the optimization problem. The maximum and minimum bounds on the thrust norm $\rho_{max},\ \rho_{min}$ are computed by means of equation (2.29). The glideslope angle γ_{gs} in a generic case can be chosen near to 90°, just to avoid terrain's collisions. The pointing angle γ_p has to be compliant with the radar requirements. The maximum velocity v_{max} needs to be a reasonable value, to avoid damages to the spacecraft structure.

- $\rho_{min} = 4.2286$ kN
- $\rho_{max} = 11.2763$ kN
- $\gamma_{gs} = 86°$
- $\gamma_p = 45°$
- $v_{max} = 800$ km/h

5.2 Examples of single cases

Singular maneuver tests are presented and discussed below, with the aim to analyse in detail the results of the algorithm, in the case of a feasible and an unfeasible optimization problem. Where, the feasibility of the optimization problem depends on the mission parameters and the initial conditions of the divert maneuver. Too demanding requests will lead to an unfeasible optimization problem and therefore a sub optimal solution, as described in section 3.3.

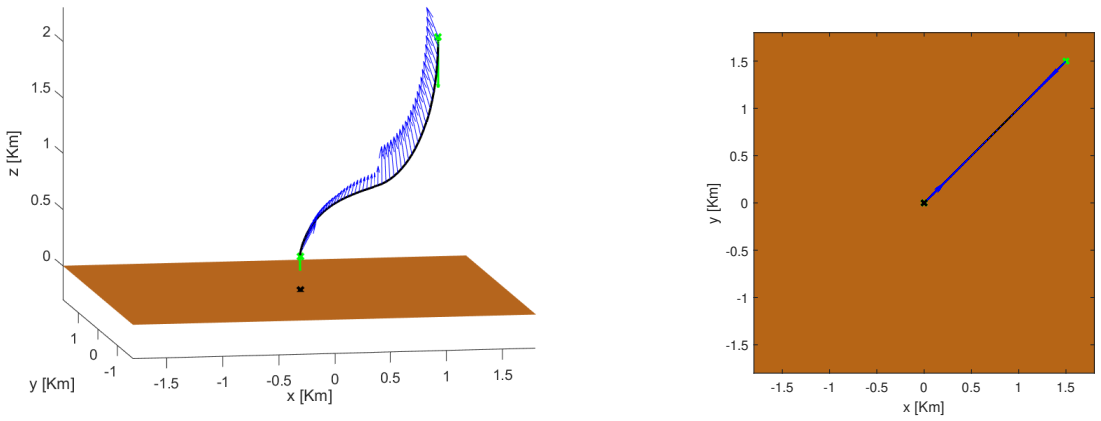
The algorithm is set up in accordance with the evaluations on the trade-off between accuracy and performance, done in the previous chapter. The mission parameters used are those described in the previous section. Whereas, the initial conditions are chosen depending on the case under study.

5.2.1 Feasible optimization problem

To analyse the solution of the trajectory planning algorithm with a feasible optimization problem (2.59), we need to select not too demanding initial conditions. The following values are chosen accordingly to the experience and the knowledge in the field, to be similar to real working conditions.

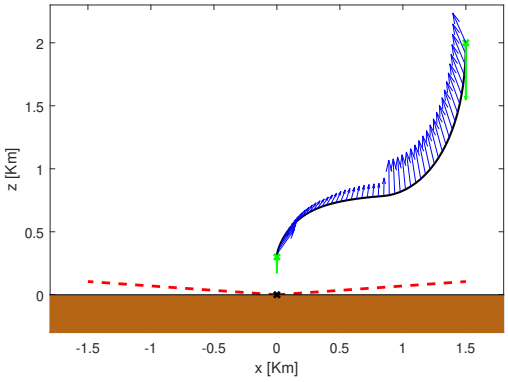
$$r_0 = \begin{bmatrix} 1500 \\ 1500 \\ 2000 \end{bmatrix} \text{m}, \qquad v_0 = \begin{bmatrix} 0 \\ 0 \\ -90 \end{bmatrix} \text{m/s} \tag{5.1}$$

Below, in Figure 5.1 are reported the plots of the variables of interest for the computed divert maneuver.

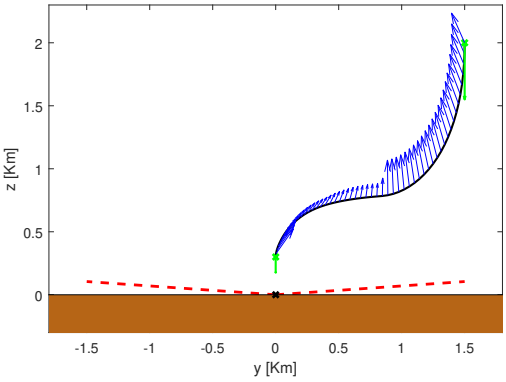


(a) Trajectory on the xyz space

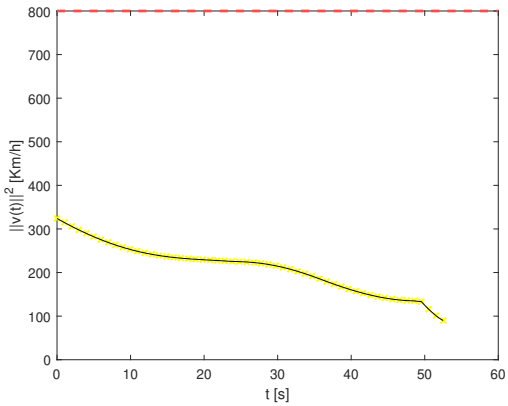
(b) Trajectory on the xy plane



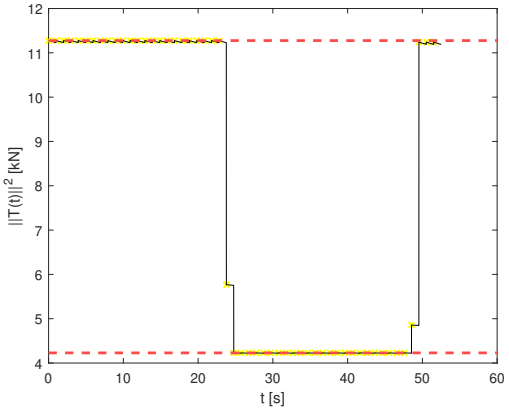
(c) Trajectory on the xz plane



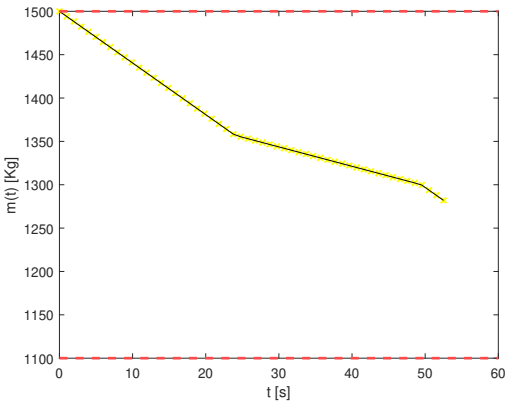
(d) Trajectory on the yz plane



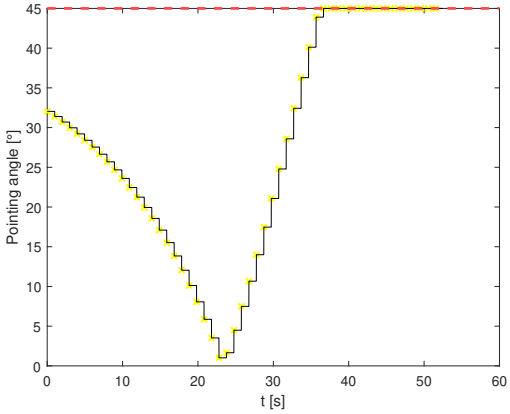
(e) Velocity norm



(f) Thrust norm



(g) Mass of the spacecraft



(h) Pointing angle

Figure 5.1: Simulation and computed results of the trajectory planning algorithm for pinpoint landing, in a case of a feasible optimization problem. In yellow the computed discrete solution, in black the simulation, in blue the thrust vector profile superimposed on the trajectory, in green the initial and final position and velocity, in red the constraints.

In Figure 5.1a,5.1b,5.1c,5.1d,5.1e, we can observe the computed trajectory for the divert maneuver, along with the velocity profile.

Since the initial velocity has not any horizontal components, the whole trajectory evolves along a straight line on the xy plane. Whereas, it assumes a S shape along the xz and yz planes. This because, the whole maneuver can be divided in the three main phases, well defined in the thrust profile Figure 5.1f.

At the beginning, we can observe a powerful braking of the spacecraft, that decreases the descent speed and at the same time accelerates the lander toward the target landing site. Then, in a second phase the thrust is reduced to the minimum to save fuel and exploit inertia to move toward the target. While at the same time, the thrust vector inverts its orientation on the other site, to be ready for the third phase. Finally, in the last phase there is a final powerful braking, that decrease the descending speed until the desired final velocity and slow down the horizontal motion until the spacecraft is above the target.

Figure 5.1f shows the profile of the thrust norm. We find the expected Bang Bang profile, foreseen by the LCvx theory, where the thrust norm assumes only maximum or minimum values. Small differences from the expected behaviour are to be taken into account, due to the dynamic linearization and the discretization of the original problem.

In Figure 5.1g, we can observe the spacecraft mass during the maneuver, that obviously monotonically decreases over the time due to the fuel consumption. Following the previously described three phases of the maneuver, since the fuel consumption is strictly related to the thrust effort.

Finally, we can observe the attitude profile in Figure 5.1h. To control purposes, it is important to observe that the attitude profile begins and ends with a non zero pointing angle. Therefore, an initial rotation of the lander is needed to bring the spacecraft in the right position to begin the divert maneuver and a final maneuver it is needed to set upright the lander before the touch down.

5.2.2 Unfeasible optimization problem

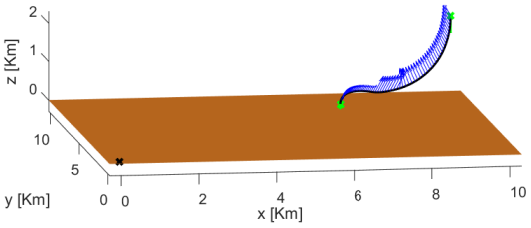
On the other side, it could happen that the spacecraft starts the divert maneuver with one or more critical initial conditions and/or critical mission parameters (e.g. initial position too far from the target, too fast initial velocity, too tight pointing angle range). In these cases, the optimization problem to be solved results to be unfeasible and therefore a sub optimal solution is needed to assure the spacecraft to reach safely the Mars surface. The computed sub optimal solution is a compromise between the final distance from the target and the fuel consumption.

Among all the possible combination of critical conditions that could lead to an unfeasible optimization problem, here we test a case with an initial position too far

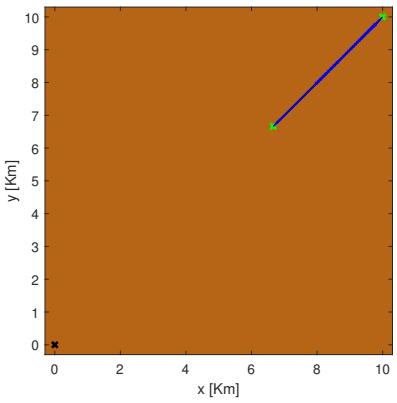
from the target to be able to reach it. The chosen initial conditions values are

$$r_0 = \begin{bmatrix} 10000 \\ 10000 \\ 2000 \end{bmatrix} \text{m} \quad v_0 = \begin{bmatrix} 0 \\ 0 \\ -90 \end{bmatrix} \text{m/s} \tag{5.2}$$

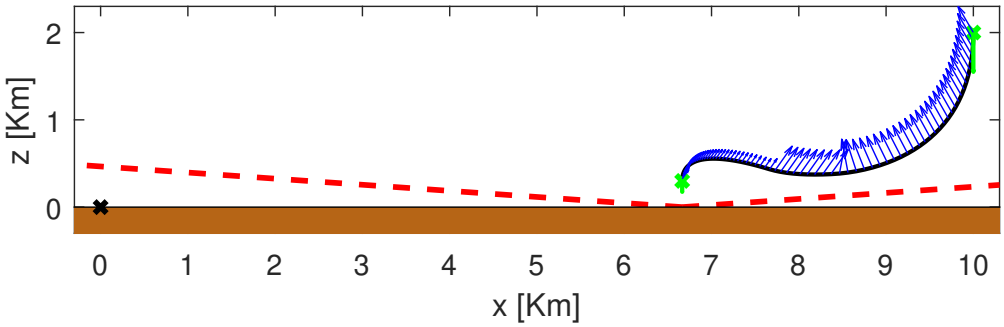
The computed sub optimal solution highly depends on the value of λ, that weight up the two terms of final distance from the target and fuel consumption, in the cost function of the optimization problem (2.60). Below are reported the plots of the variables of interest for the computed divert maneuver, for two different values of λ.



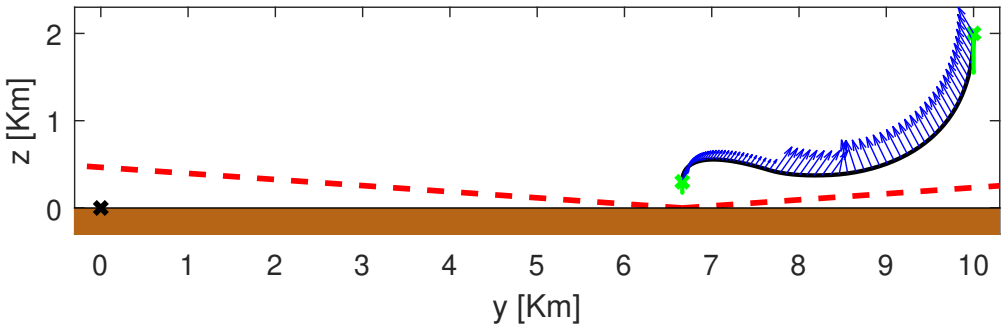
(a) Trajectory on the xyz space



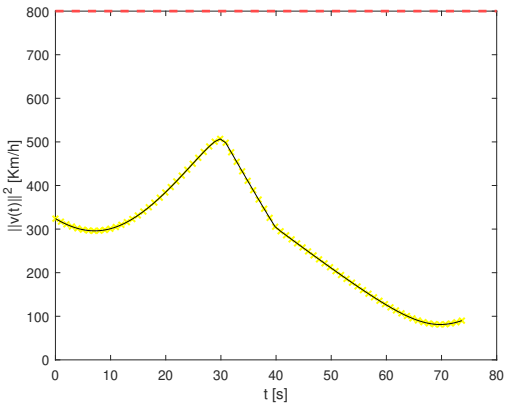
(b) Trajectory on the xy plane



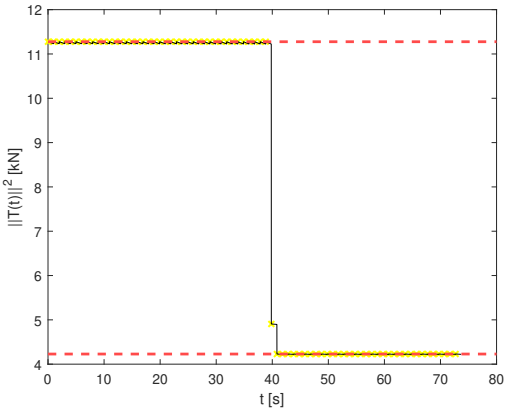
(c) Trajectory on the xz plane



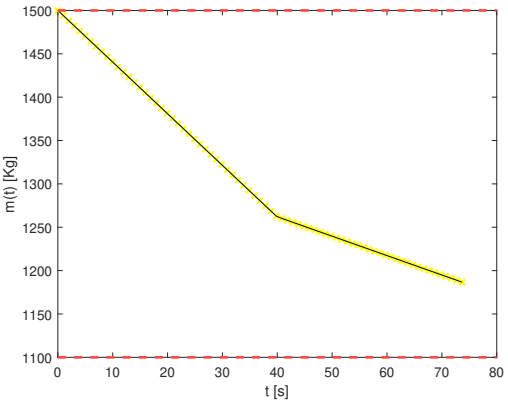
(d) Trajectory on the yz plane



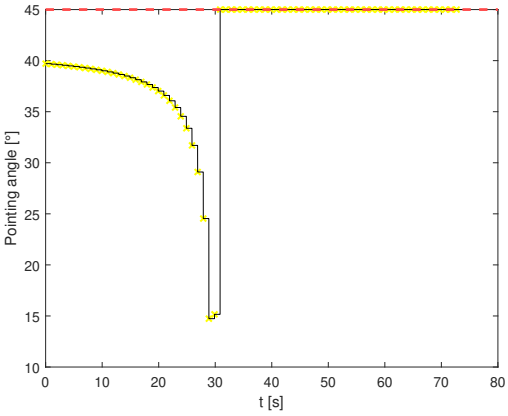
(e) Velocity norm



(f) Thrust norm

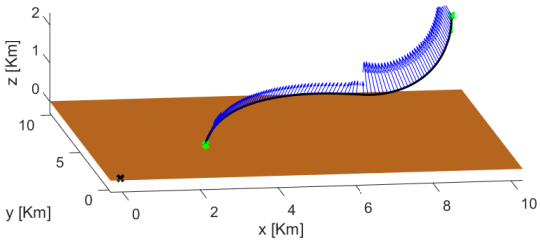


(g) Mass of the spacecraft

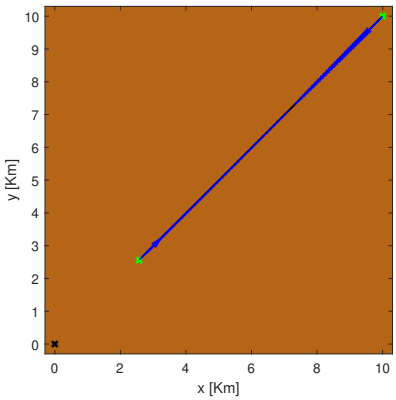


(h) Pointing angle

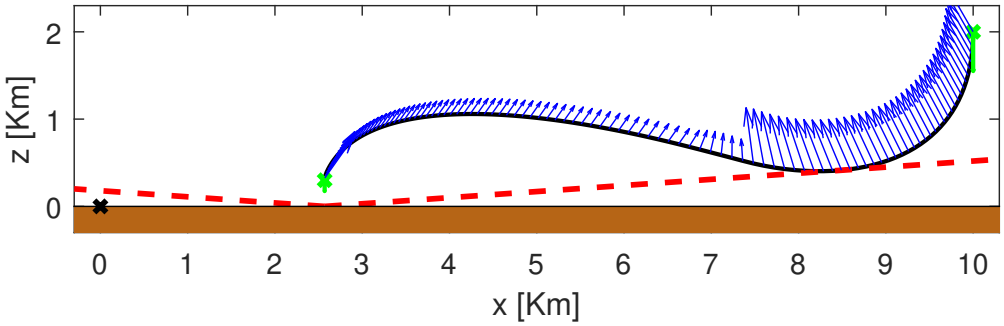
Figure 5.2: Simulation and computed results of the trajectory planning algorithm for pinpoint landing, in a case of an unfeasible optimization problem. In yellow the computed discrete solution, in black the simulation, in blue the thrust vector profile superimposed on the trajectory, in green the initial and final position and velocity, in red the constraints. The value of λ is set to 1.



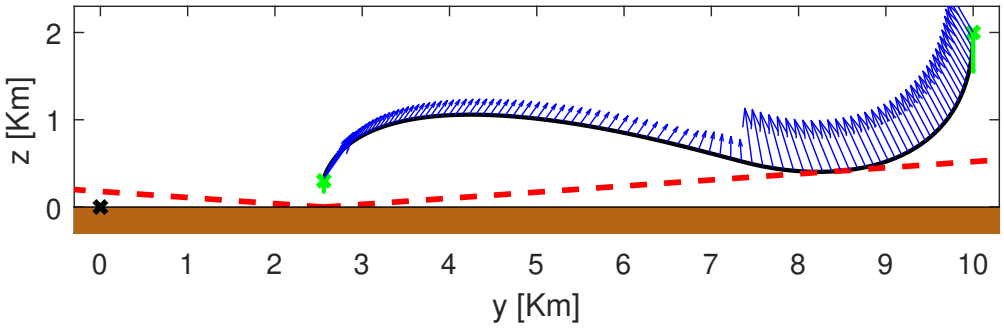
(a) Trajectory on the xyz space



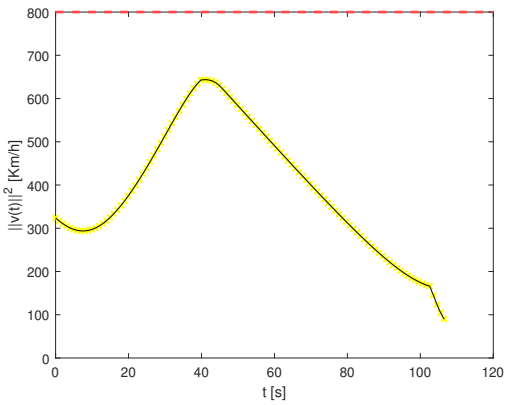
(b) Trajectory on the xy plane



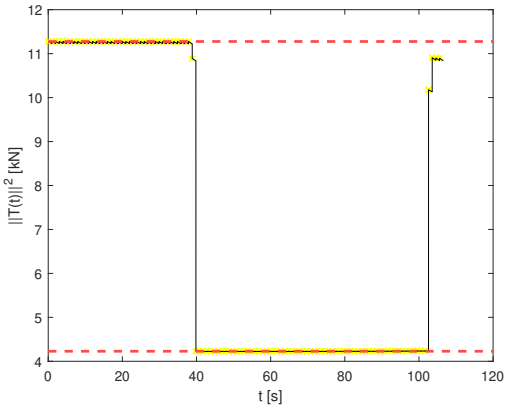
(c) Trajectory on the xz plane



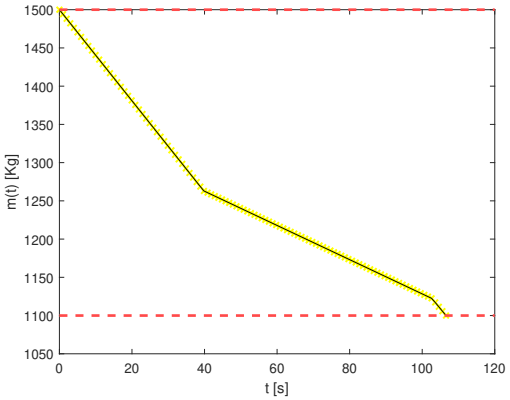
(d) Trajectory on the yz plane



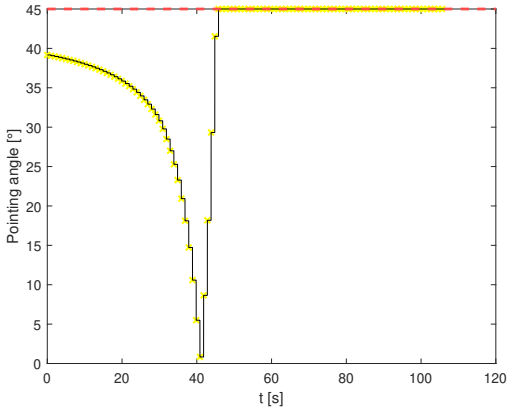
(e) Velocity norm



(f) Thrust norm



(g) Mass of the spacecraft



(h) Pointing angle

Figure 5.3: Simulation and computed results of the trajectory planning algorithm for pinpoint landing, in a case of an unfeasible optimization problem. In yellow the computed discrete solution, in black the simulation, in blue the thrust vector profile superimposed on the trajectory, in green the initial and final position and velocity, in red the constraints. The value of λ is set to 0.1.

The first case with $\lambda = 1$ in Figure 5.2 is compared with the second case with $\lambda = 0.1$ in Figure 5.3. In general, both cases have similar behaviour to the feasible case in Figure 5.1. Whereas, as we could imagine, they differ significantly in the mass and trajectory results. Bigger values of lambda lead to less fuel consumption at the end of the divert maneuver, but at the same time worst results to reach the target.

Another important difference regards the pointing angle and its rate. In contrast to the feasible case in Figure 5.1h, the results achieved in Figure 5.2, 5.3 have a very sharp rising edge respectively at time $t = 31$ s and $t = 41$ s. This sudden increase of pointing angle cannot be easily realized, due to the limitation on the attitude dynamics of the spacecraft.

In conclusion, in both cases the impossibility to reach the desired landing site leads the spacecraft to move close to the target without reaching it. Then, we need to choose the value of λ depending on the specific situation and our primary goals.

5.3 Monte Carlo simulations

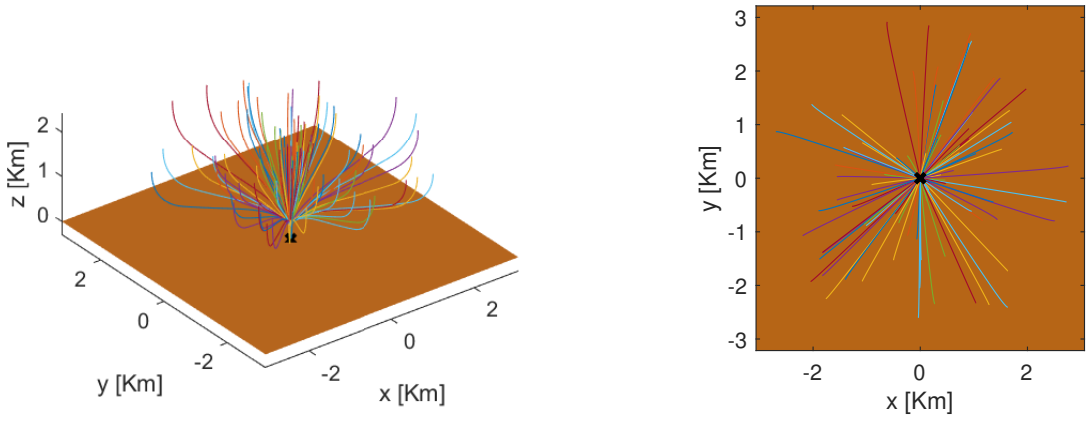
To study the overall behaviour of the algorithm, we observe the results of a numerous set of cases, over a defined range of reasonable working conditions. They are a random collection of samples of possible outcomes of the entry and descending phase.

In the guided entry phase, thanks to bank reversal maneuvers, we are able to reach the target within a precision of 3 km [3]. Then, in the descending phase a parachute slows down the descent until manageable speeds. Therefore, the nominal working conditions used in the Monte Carlo experiment are

$$\begin{aligned} r_0 &= \begin{bmatrix} r_{0x} \\ r_{0y} \\ r_{0z} \end{bmatrix}, \quad s.t. \quad \begin{aligned} &\sqrt{r_{0x}^2 + r_{0y}^2} \in [0{,}3000]\text{m} \\ &r_{0z} \in [1900{,}2100]\text{m} \end{aligned} \\ v_0 &= \begin{bmatrix} v_{0x} \\ v_{0y} \\ v_{0z} \end{bmatrix}, \quad s.t. \quad \begin{aligned} &v_{ox},\ v_{0y} \in [-5{,}5]\text{m/s} \\ &v_{0z} \in [-100, -80]\text{m/s} \end{aligned} \end{aligned} \tag{5.3}$$

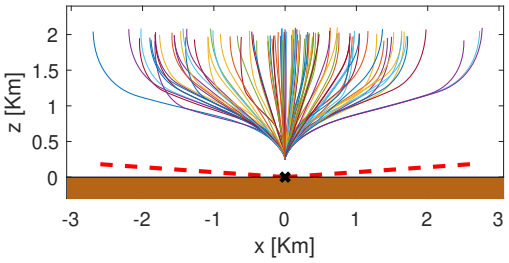
Hopefully, within these values we will obtain only feasible optimization problems and therefore optimal solutions.

Instead, as far as concern unfeasible optimization problems, since countless combination of critical parameters could lead to unfeasibility, a more detailed analysis of them will be proposed in the next chapter only for some specific cases of interest.

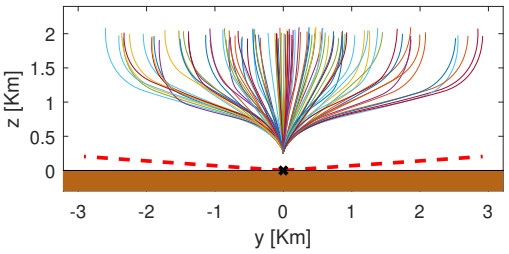


(a) Trajectory on the xyz space

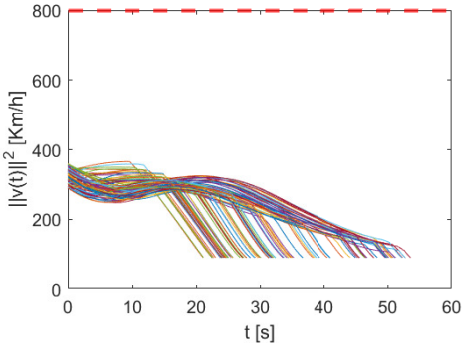
(b) Trajectory on the xy plane



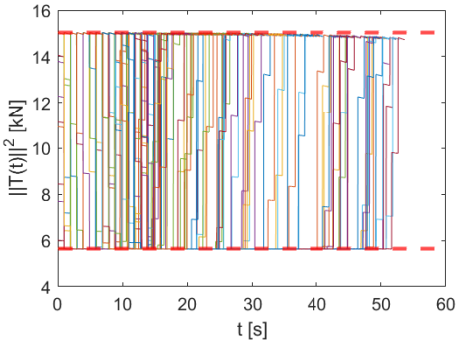
(c) Trajectory on the xz plane



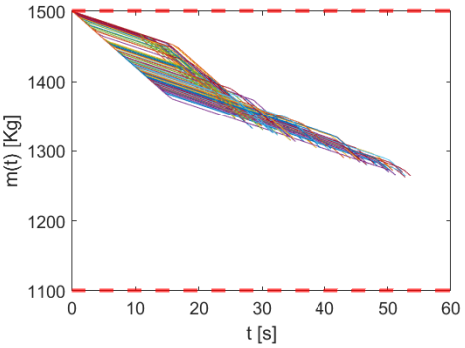
(d) Trajectory on the yz plane



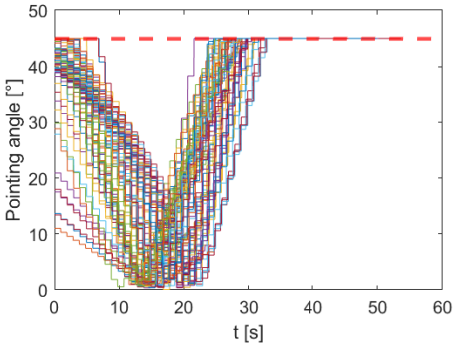
(e) Velocity norm



(f) Thrust norm



(g) Mass of the spacecraft



(h) Pointing angle

Figure 5.4: Simulation and computed results of the trajectory planning algorithm for pinpoint landing, for a Monte Carlo experiment (100 cases). Every simulation is represented with a different color, in red the constraints

As we can observe in the Figure 5.4, all the cases tested lead to feasible optimization problems and therefore to optimal solutions. Hence, the Monte Carlo experiment confirms the feasibility of the working condition under analysis (5.3). It proves the reliability of the algorithm under nominal working conditions.

Chapter 6

Simulations

In this chapter, we simulate the behavior of a complete Guidance Navigation and Control (GNC) algorithm for pinpoint landings, coupled with the developed trajectory planning algorithm, that provides in this framework the guidance function for the divert maneuver.

The simulator computes the spacecraft dynamics, given the thrust command provided by the GNC. The GNC, in turn, is fed by the sensor measurements provided by the simulator, adding random disturbances to the real values. The simulation includes the entire lading maneuver, from the separation of the lander from the backshell and parachute composite, to the touch down on the surface.

6.1 Overview of the simulation enviroment

The simulator, as well as the GNC, used in this simulation, are kindly provided by Thales Alenia Space. The algorithm developed in this thesis work is integrated in the provided GNC algorithm. It is used as guidance function for the divert maneuver.

The GNC is composed by three separated pieces of software, with different purposes, that interact with each other, as represented in Figure 1.2.

- The navigation function makes sensor fusion on the measurements provided by the Inertial Measurements Unit (IMU), Radar Doppler Altimeter (RDA) and Landing Vision System (LVS), to estimate position, velocity, acceleration, attitude, angular rate and angular acceleration of the spacecraft throughout the landing phase.
- The guidance function computes the reference trajectory for the landing. Once the separation of the lander occurs, at the end of the descent phase, an initial tip-up rotation brings the spacecraft to the initial attitude required by the divert maneuver. Then, the divert maneuver slows down the vertical velocity and at the same time recovers the horizontal distance from the target. Finally, a sequence of procedures is carried out in order to guarantee a safe touch down of the lander on the surface. Firstly, a rotation aligns vertically the lander. Then, an intensive braking decreases significantly the remaining descent velocity. The thrust is smoothly decreased until only gravity is compensated. The spacecraft lands on the surface. The thrusters are switched-off at first contact with the martian terrain. This is done independently by the GNC control loop thanks to touch-down sensors placed in the landing legs with a redundant configuration allowing majority voting to exclude undesired switch-off in advance.
- The control function takes care of keeping the spacecraft on track during the landing. Indeed, although the computed trajectory results to be feasible, we should always deal with disturbances and model uncertainties.

The simulator provides an environment to test the performance and the robustness of the GNC algorithm. The spacecraft dynamics is computed accurately, given the initial conditions and the thrust command provided by the GNC in real time. The sensor measurements, used to run the GNC algorithm, are simulated adding random disturbances.

6.2 Example case

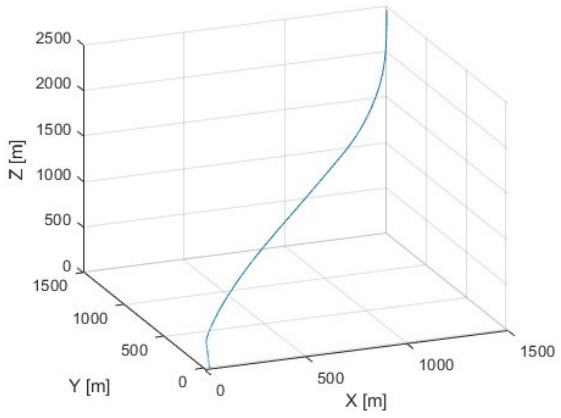
In this section, a simulation of the GNC behavior is carried out, in an example case. The lander chosen for the simulation is similar to the ExoMars one, but properly enlarged to be able to host a bigger propellant mass, in order to allow the divert maneuver execution. The mission parameters used are the same presented in section 5.1. Except for the rocket parameters, whose values are reported below:

- $T_{max} = 2.5$ kN
- $n_{eng} = 8$
- $\phi = 20°$

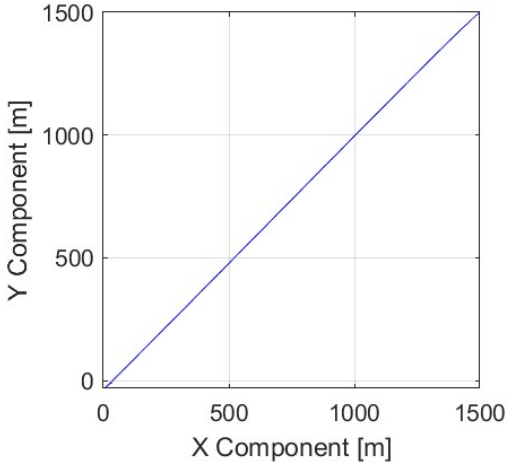
The initial conditions of the spacecraft are:

$$r_0 = \begin{bmatrix} 1500 \\ 1500 \\ 2000 \end{bmatrix} \text{m}, \qquad v_0 = \begin{bmatrix} 0 \\ 0 \\ -90 \end{bmatrix} \text{m/s} \tag{6.1}$$

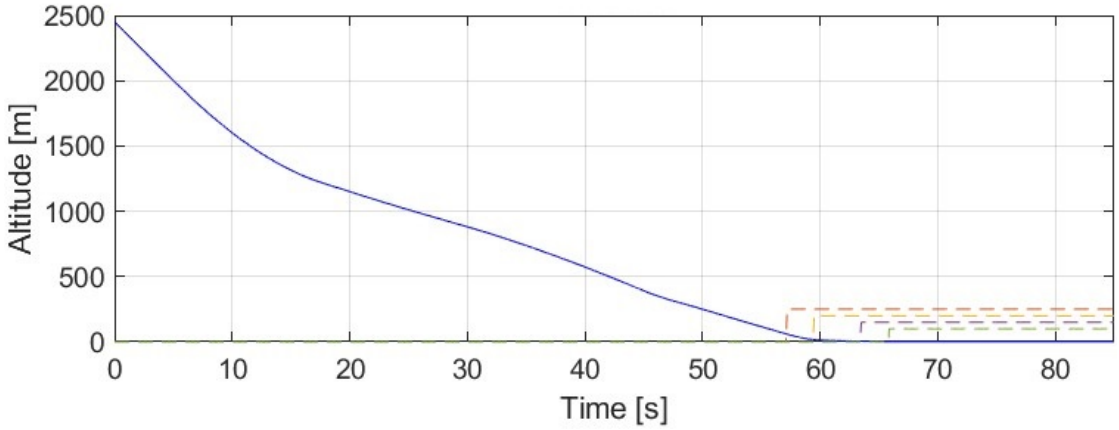
The results of the simulation are reported in the figures below:



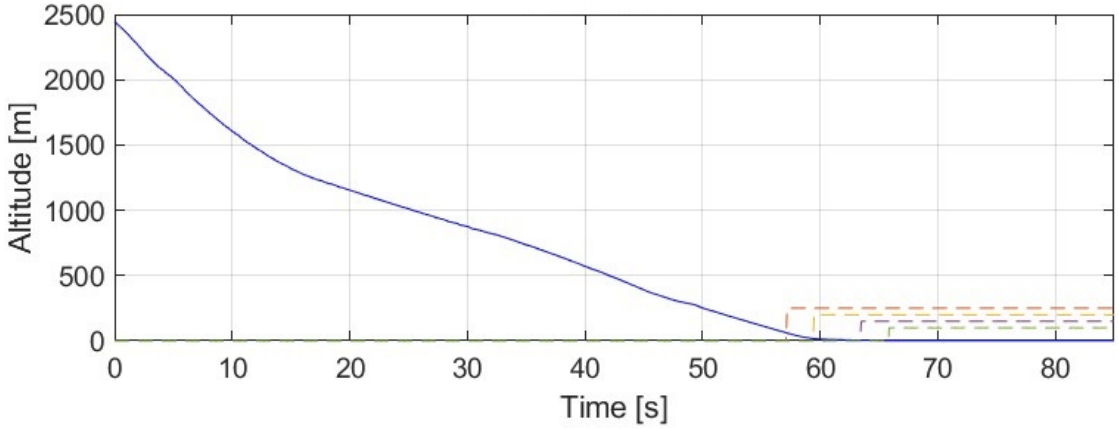
(a) Trajectory in the xyz plane



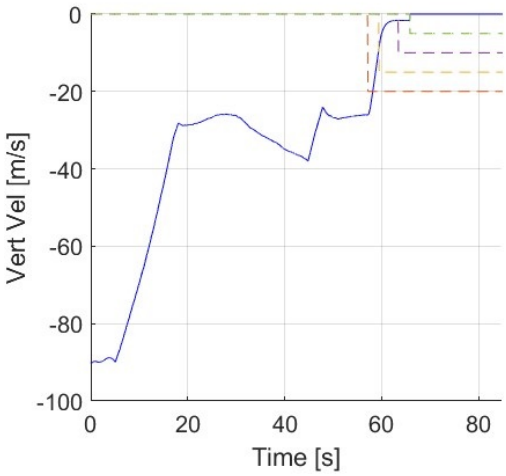
(b) Trajectory in the xy plane



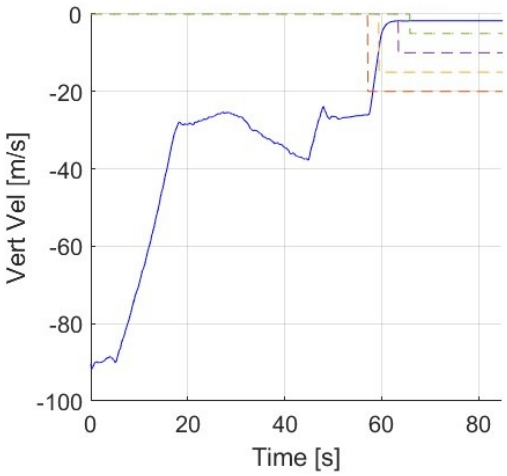
(c) Altitude



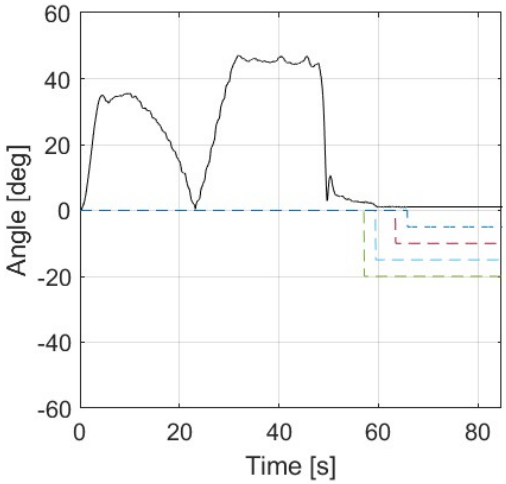
(d) Estimated altitude



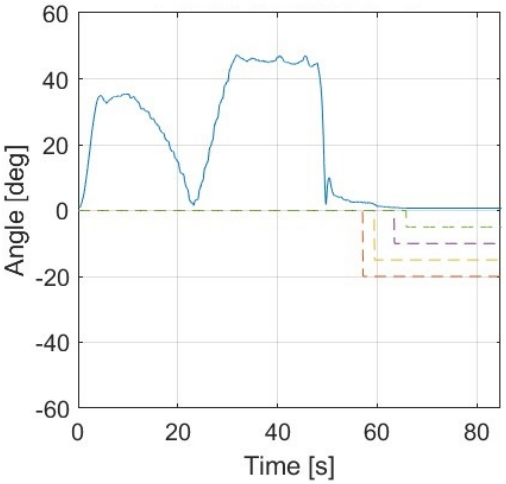
(e) Vertical velocity



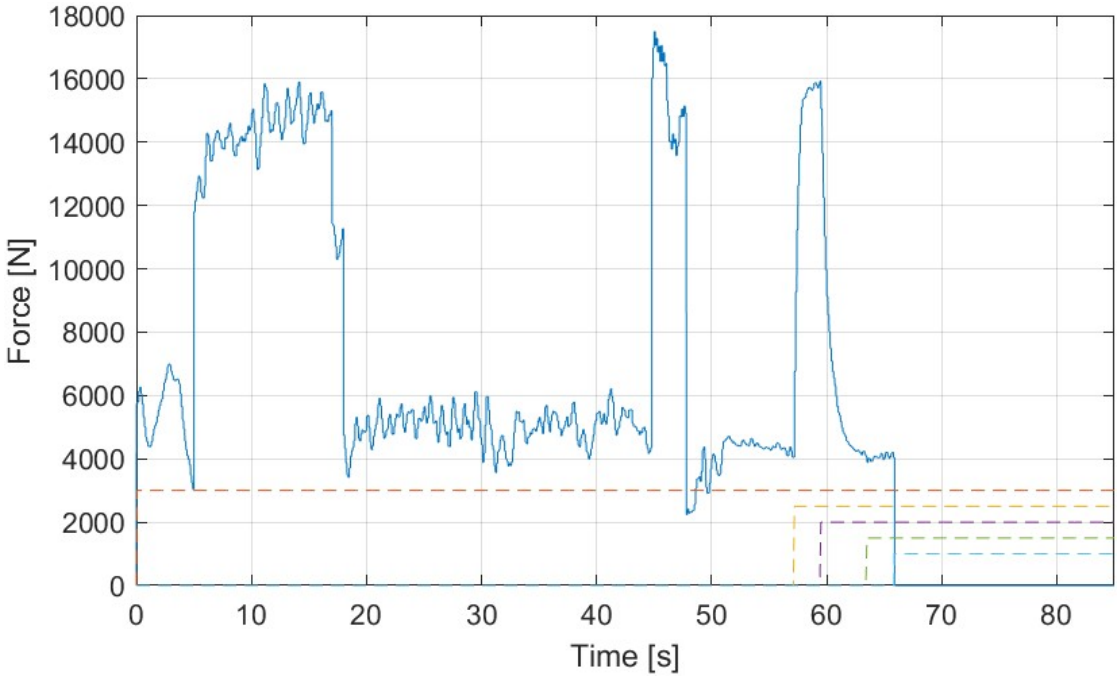
(f) Estimated vertical velocity



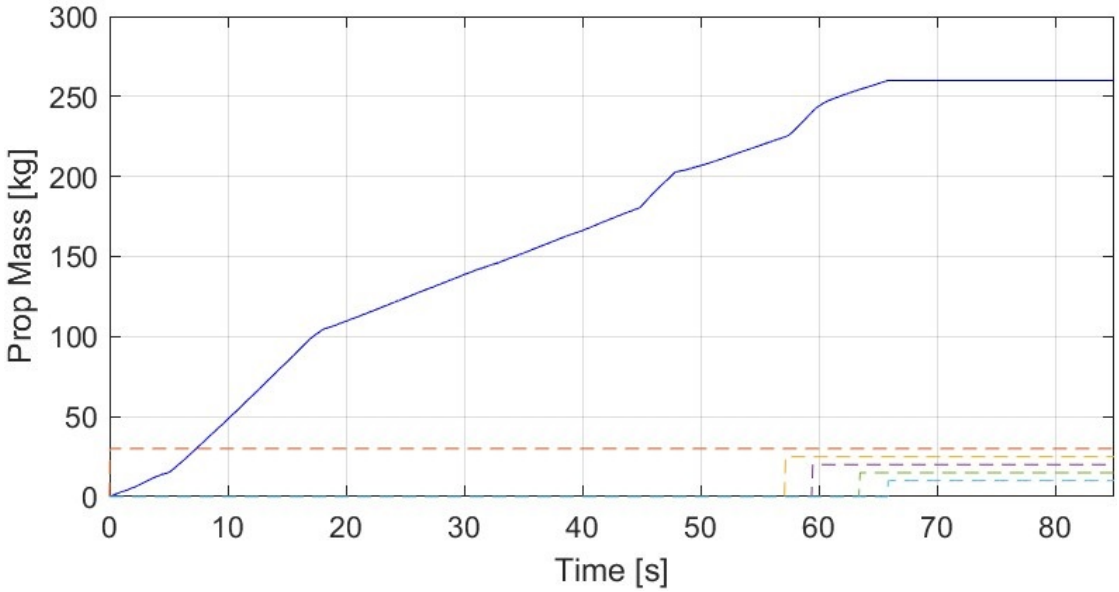
(g) Pointing angle



(h) Estimated pointing angle



(i) Thrust norm



(j) Fuel consumption

Figure 6.1: Results of the landing simulation, in an example case. Superimposed on the graphs are drawn in sequence the activation flags of the following events: intensive braking, final phase transition, ready for touch down, leg touch down.

As we can see in the simulation results presented in Figure 6.1, the spacecraft at the end of the descent phase is oriented almost vertically. Then, an initial rotation is necessary to reach 35° of pointing angle, required as initial condition of the divert maneuver. In the meanwhile, the spacecraft maintains the same descent velocity of -90 m/s travelling 450 m in 5 s.

Then, the divert maneuver begins, the descent velocity is slowed down to -24 m/s, and a final distance from the target of 35 m is achieved.

Later, a verticalization of the lander is carried out, to prepare the spacecraft for the touch down.

The spacecraft is at 300 m above the ground when it begins the touch down maneuvers. Firstly, an intense braking decreases significantly the vertical velocity up to -8 m/s. Then, a final transition smoothly decreases the thrust command to the minimum. At this point, the lander is ready for the touch down. When the leg sensors perceive the ground, the thrusters are shut down.

At the touch down, the lander reaches a final distance from the target of 35 m, with a vertical velocity of -1.6 m/s, a horizontal velocity of 0.8 m/s and a pointing angle of 1°.

Chapter 7

Parametric analysis

In this chapter, a parametric analysis of the trajectory planning algorithm is carried out. This study evaluates the obtained results, varying a selection of significant design parameters. With the aim of extrapolating a general behavior of the algorithm.

Firstly, we look for the feasibility boundaries of the original optimization problem. Then, where the optimization problem turns out to be unfeasible, we compute a sub optimal solution and evaluate its performance.

This analysis provides interesting tips for the spacecraft design, relating the performance of the propulsion system, the parachute, the radar doppler altimeter and the landing vision system, with the results of the trajectory planning algorithm.

7.1 Analysis for the driver parameters

A selection of relevant parameters have to be chosen to drive this analysis. Combinations of their values will be used to study the feasibility of the guidance problem, whereas all the other parameters will be kept constant. The parameters chosen are strictly related to the design of the lander.

- **Initial vertical velocity** $v(0) = [0,0,v_0]^T$
  The initial velocity depends on the performance of the parachute. From the available data, we can expect values of $\|v(0)\|_2$ within the range [90,120] m/s. To simplify the analysis, we assume zero horizontal components of the initial velocity.

- **Maximum pointing angle** γ_p
  The value of the maximum pointing angle is strictly related to the radar doppler altimeter (RDA) placement. So firstly, we need to explain the RDA configuration and its two possible placements on the bottom of the spacecraft. The RDA is a sensor, that measures altitude and velocity of the spacecraft, exploiting the slant and doppler measurements along 4 directions (one direction parallel to the symmetry axis and the other ones 20° skewed), as represented in Figure 7.1.

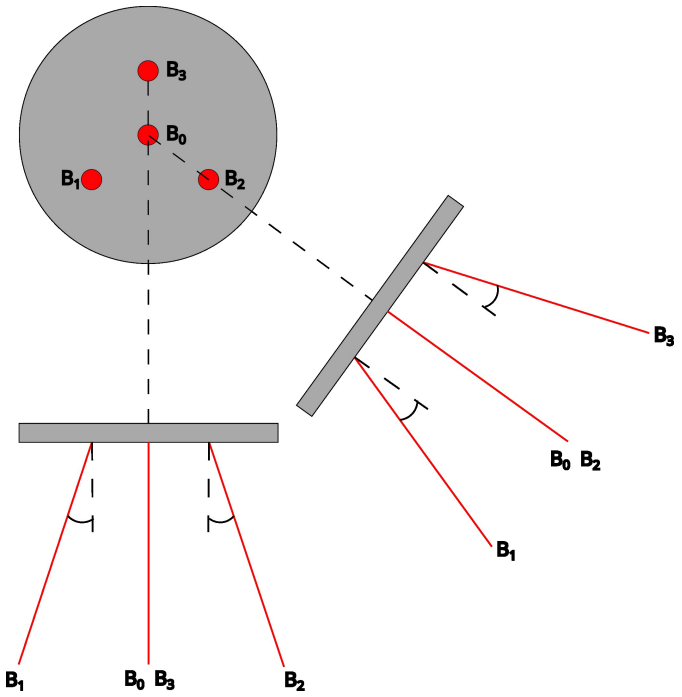
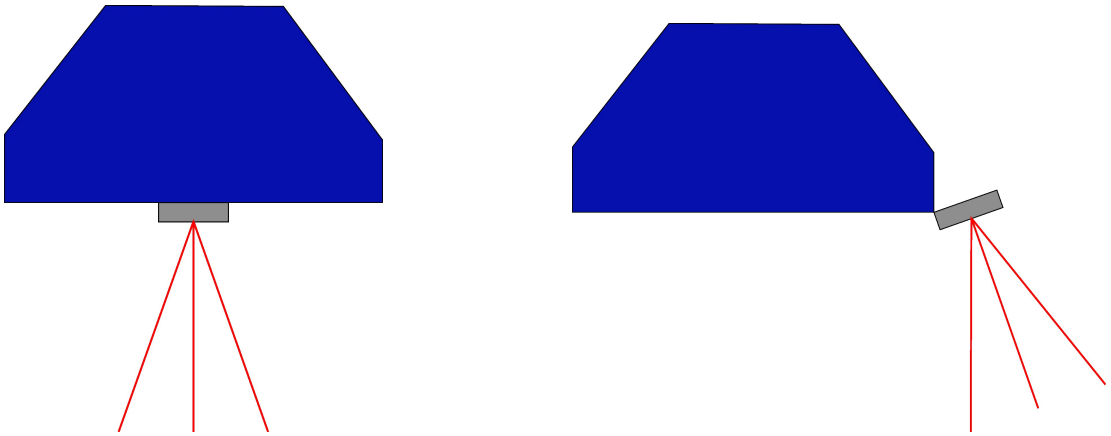


Figure 7.1: RDA configuration. In red the beams, from different views.

To work properly, the RDA needs reliable measurements from at least 3 of 4 beams. A beam no longer works when the distance to measure is too large. This happens when the inclination of the beam, with respect to the local vertical k_{ENU}, exceeds 60°.

Regarding the RDA location in the spacecraft, this study considers as applicable the ExoMars configuration, represented in Figure 7.2a. The RDA is mounted in the center of the spacecraft's bottom, to exploit its symmetrical geometry. In this way, the RDA behaves the same regardless the attitude of the lander. But, the maximum pointing angle cannot be larger than 40° for 4 beams still working and 48.1° for 3 beams still working.

An alternative placement is the one chosen in the NASA missions MSL and MSR, represented in Figure 7.2b. This configuration exploits an asymmetric geometry, the RDA is placed on a side of the spacecraft's bottom, 20° skewed in order to align an external beam to the body vertical vector k_{BRF}. This different approach brings improvements on the maximum pointing angle achievable, if the descent direction is aligned with the RDA. In this way, the maximum pointing angle becomes 35.4° for 4 beams still working and 57.8° for 3 beams still working. But, on the other hand, is required an additional rotation maneuver at the beginning of the divert maneuver and an attitude control during the descent.



(a) ExoMars configuration. **(b)** MSL and MSR configuration.

Figure 7.2: Position and attitude of the Radar Doppler Altimeter in the spacecraft.

- **Altitude loss** $h = h_0 - h_N$
  The altitude loss h is the altitude difference between the start and the end of the divert maneuver. Whereas, the final altitude h_N is fixed at 300 m, the initial altitude h_0 can vary within an upper and a lower boundary value. For the purpose of this analysis, we need to estimate the upper boundary of h_0 and thus the upper boundary of h.

  The divert maneuver begins once the reference trajectory is computed. But, to compute the reference trajectory, we need to know the exact location of the spacecraft with respect to the desired landing site. The Landing Vision System (LVS) is in charge to measure the position of the lander. It compares constellations of landmarks in the camera and in the on-board maps, till a successful matching locates the spacecraft. This operation needs to be repeated at least twice to be validated. To work properly, the landing navigation algorithms need to know the altitude of the spacecraft. The altitude measurements are provided by the Radar Doppler Altimeter, and they are not available before reaching 4200 m.

  Once reached that altitude, the LVS needs 10 s to locate the spacecraft. Then, the guidance problem must be solved, the landing platform must be separated by the composite of parachute and backshell and the spacecraft must be oriented in the skewed attitude required at initiation of the divert maneuver. It is necessary to allocate about 7 s for these operations, and an extra 2 s as safety margin. Therefore, after reaching 4200 m a total of 19 s must elapse before starting the divert maneuver. Hence, considering a maximum descent velocity of 120 m/s, in this time interval the spacecraft travels 120 m/s * 19 s = 2280 m. Thus, the divert maneuver starts at most at $4200 - 2280 \simeq 1900$ m. Or rather, at most with an altitude loss of $h = h_0 - h_N = 1900 - 300 = 1600$ m.

- **Thrust and Mass** T2W=$\frac{T_{max} n_{eng} cos(\phi)}{m_{wet} g}$
  The thrust and the mass characterize the dexterity of the spacecraft during the divert maneuver. Since the dynamic of the spacecraft is affected simultaneously by both, we need to use a new metric, the Thrust to Weight (T2W), which takes both into consideration.

  T2W is defined as the ratio between a thrust and a mass reference value. As thrust reference, we take the maximum thrust value obtainable by all the rockets together $T_{max} n_{eng} cos(\phi)$. Whereas, as mass reference, the force of gravity of the spacecraft at the beginning of the maneuver $m_{wet} g$.

  To provide a reasonable amount of thrust throughout the maneuver, a lower boundary of T2W is set to 2 (with a maximum achievable thrust twice the force of gravity). Then, the following three cases are considered:

– The same spacecraft's features used for the tests in the previous chapter.

$$\begin{cases} T_{max} = 2500\text{N} \\ n_{eng} = 6 \\ \phi = 20^\circ \\ m_{wet} = 1500\text{kg} \end{cases} \Rightarrow \quad T2W = 2.53 \tag{7.1}$$

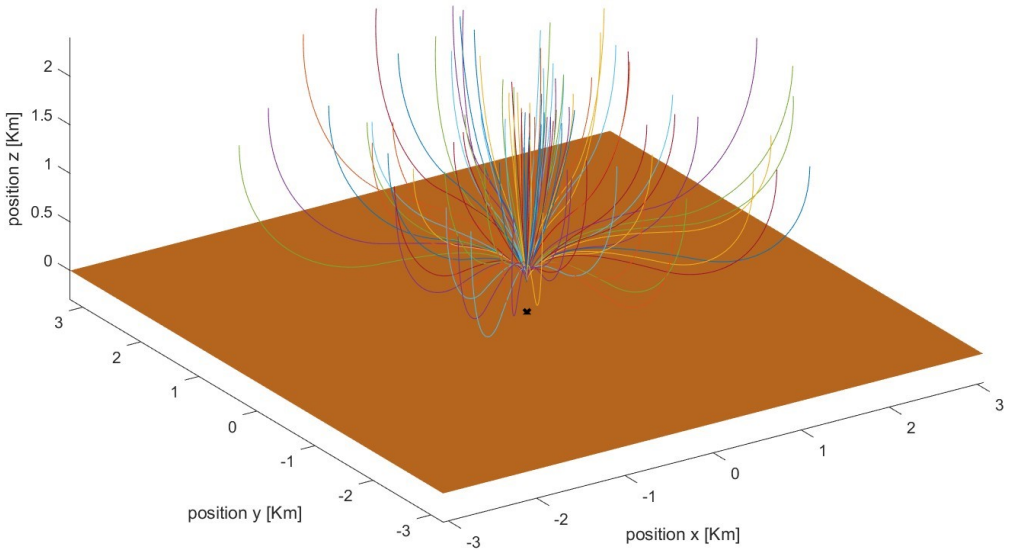
– The previous case with an augmented number of engines. The configuration of the thrusters has to remain symmetric, with an even number of engines (n=8), to provide lateral thrust stability.

$$\begin{cases} T_{max} = 2500\text{N} \\ n_{eng} = 8 \\ \phi = 20^\circ \\ m_{wet} = 1500\text{kg} \end{cases} \Rightarrow \quad T2W = 3.38 \tag{7.2}$$

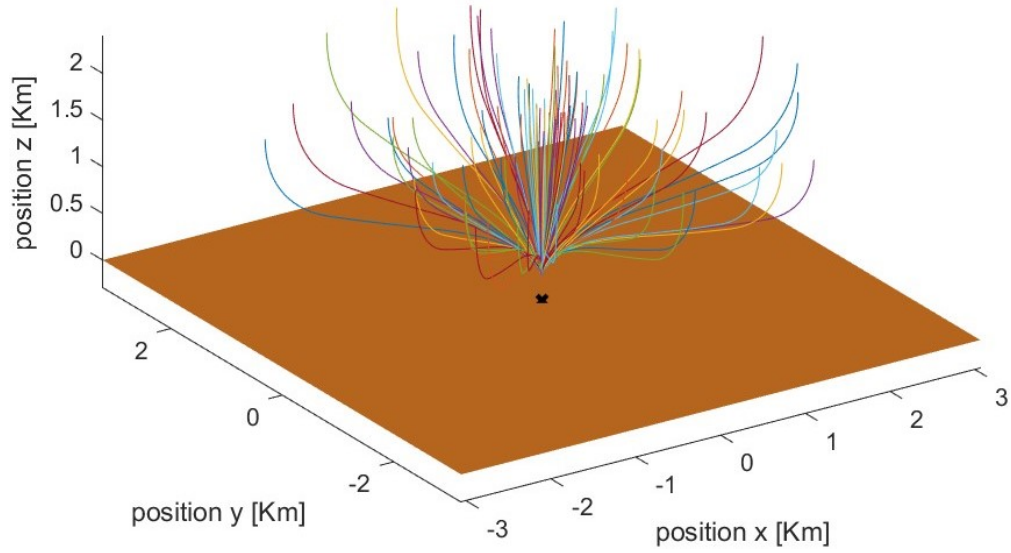
– The previous case with more powerful rockets, the same presented in the literature [9].

$$\begin{cases} T_{max} = 3100\text{N} \\ n_{eng} = 8 \\ \phi = 20^\circ \\ m_{wet} = 1500\text{kg} \end{cases} \Rightarrow \quad T2W = 4.18 \tag{7.3}$$

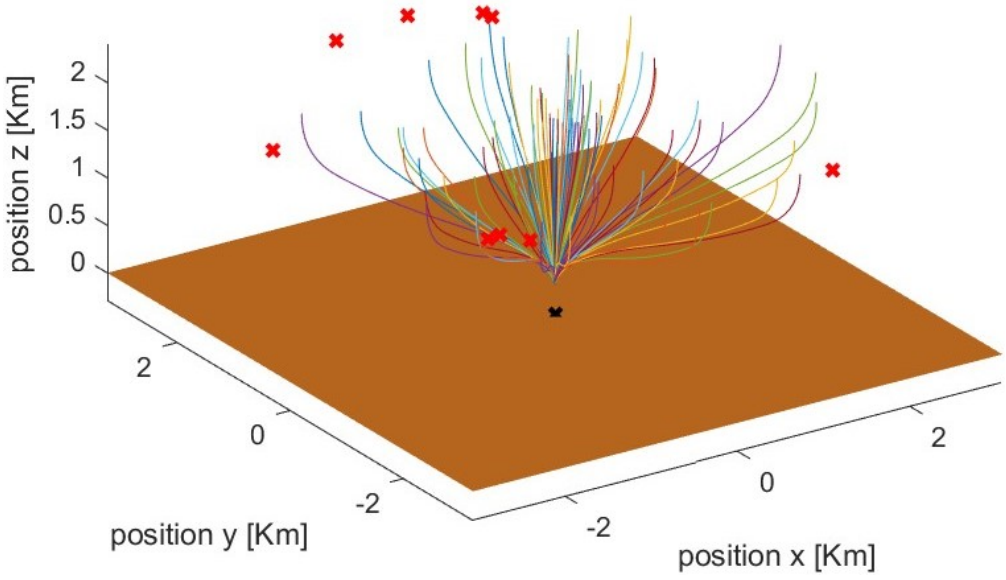
A more thorough analysis of these cases is conducted, before proceeding with the parametric analysis. Each case is tested with a Monte Carlo experiment, to prove the reliability of the algorithm over a reasonable range of initial conditions, the same used in section 5.3.



(a) T2W = 2.53.



(b) T2W = 3.38.



(c) T2W = 4.18.

Figure 7.3: Trajectories on the xyz space, computed by a Monte Carlo experiment (100 cases), for three different values of T2W. In red the initial positions leading to the unfeasibility of the optimization problem 2.59.

As we can see from the results in Figure 7.3a, 7.3b, the cases with T2W= {2.53,3.38} provide good results. A difference in the shape of the trajectories can be observed, it concerns mainly the initial phase. A lower T2W value means greater inertia in the movement and therefore a longer initial braking phase.

On the other hand, with T2W=4.18 as represented in Figure 7.3c, the trajectory planning algorithm cannot find a feasible solution of the optimization problem 2.59, for cases with an initial distance from the target d_0 greater than 2700 m. Initially, we could think of an impossibility to reach the target from such a distance. Instead, a deeper analysis reveals the existence of some feasible trajectories able to accomplish this task. The problem in these cases is related to the golden search, which fails to find a feasible time of flight. To explain the reason, we need to remember that the golden search, presented in section 3.2, is an iterative search algorithm that computes and compares the results of an optimization problem for a sequence of values of time of flight t_f within a range $[t_{min}, t_{max}]$. Unfortunately, the convergence of the result is guaranteed only if

both t_{min} (3.2), t_{max} (3.3) correspond to feasible optimization problems. But, since these values are initial guesses, usually do not fulfill such requirement. We call $\hat{t}_{min}, \hat{t}_{max}$ the minimum and maximum values of time of flight leading to a feasible optimization problem. Then, the values of t_{min}, t_{max} are chosen such that $[\hat{t}_{min}, \hat{t}_{max}] \subseteq [t_{min}, t_{max}]$. The wider is the feasible interval $[\hat{t}_{min}, \hat{t}_{max}]$, the more likely the golden search will converge to a feasible solution. Narrow intervals could be not found, leading to an unfeasible solution. Returning to the beginning of this digression, for T2W=4.18 and $d_0 \geq 2700$ m, the golden search no longer provides a feasible solution to the problem. Indeed, in these cases, $\hat{t}_{max}$ turns out to be significantly reduced, which leads to a narrow feasible interval $[\hat{t}_{min}, \hat{t}_{max}]$. This aspect can be explained considering the greater thrust available. Indeed, a bigger feasible time of flight $\hat{t}_{max}$ is reached with a slower movement on the xy plane. The thrust vector needs to be pointed along a more vertical direction, while moving toward the target. But, since in this case the thrust provided is bigger, reducing its component along xy means to increase further its component along z. An excessive thrust component on z increases too much the vertical velocity of the lander and prevents the spacecraft to reach the desired final vertical velocity.

Finally, we conclude that we could find a feasible solution of the optimization problem 2.59, but it would need a different search algorithm with an increased computational effort, that we cannot afford in a real time application. Thus, no case of T2W≥ 4 will be considered in this study.

For the sake of completeness, we report also the T2W value of the case presented in literature [9].

$$\begin{cases} T_{max} = 3100\text{N} \\ n_{eng} = 6 \\ \phi = 27^\circ \\ m_{wet} = 1905\text{kg} \end{cases} \quad \Rightarrow \quad T2W = 2.34 \tag{7.4}$$

The remaining parameters are the same used in the previous chapter, defined in section 5.1.

The initial position on the xy plane is taken at 3000 m far from the target, i.e. the worst case we could reasonably expect on the basis of the Guided Entry assumed performances.

$$\begin{bmatrix} r_{0x} \\ r_{0y} \end{bmatrix}, \quad s.t. \quad d_0 = \sqrt{r_{0x}^2 + r_{0y}^2} = 3000\text{m} \tag{7.5}$$

Finally, the glideslope constraint is applied on the final position of the divert maneuver and no longer on the landing site.

7.2 Impact of the configuration

The parametric analysis evaluates, separately for T2W= {2.53,3.37}, the value of the minimum altitude loss achieved by a feasible optimization problem, for the following values of initial velocity and maximum pointing angle:

- $v_0 = \{-90, -100, -110, -120\}$ m/s
- $\gamma_p = \{35°, 40°, 45°, 50°\}$

Then, the computed minimum altitude losses are compared with the maximum altitude loss compatible with the LVS, that is 1600 m. If the computed value is over this threshold, the trajectory planning algorithm will be forced in any case to begin at 1900 m, leading to an unfeasible optimization problem and to a sub optimal solution.

7.2.1 Algorithm description

The algorithm designed to find the minimum altitude loss compatible with a feasible optimization problem (2.59) is similar to the bisection method.

It starts with two extreme values of altitude $r_{0z\_min},\ r_{0z\_max}$. These values are initial guesses, r_{0z_min} has to be associated to an unfeasible optimization problem and r_{0z_max} to a feasible optimization problem. The values chosen are

$$\begin{aligned} r_{0z\_min} &= h_N = 300 \text{ m} \\ r_{0z\_max} &= 3000 \text{ m} \end{aligned} \tag{7.6}$$

Then, at each iteration, the middle point between $r_{0z\_min},\ r_{0z\_max}$ is computed as

$$r_{0z} = \frac{r_{0z\_min} + r_{0z\_max}}{2} \tag{7.7}$$

And it is used to compute again the feasibility of the optimization problem. If it turns out to be a feasible altitude loss $r_{0z\_max} = r_{0z}$, otherwise $r_{0z\_min} = r_{0z}$. Then, to achieve a precise result, this iteration step is repeated 10 times.

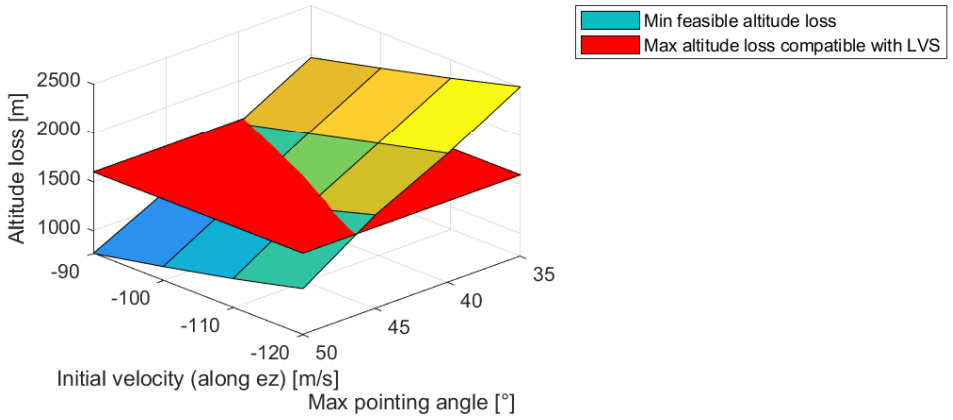
Finally, the whole procedure is repeated with a different combination of v_0 and γ_p, until all 16 combinations are tested.

7.2.2 Results

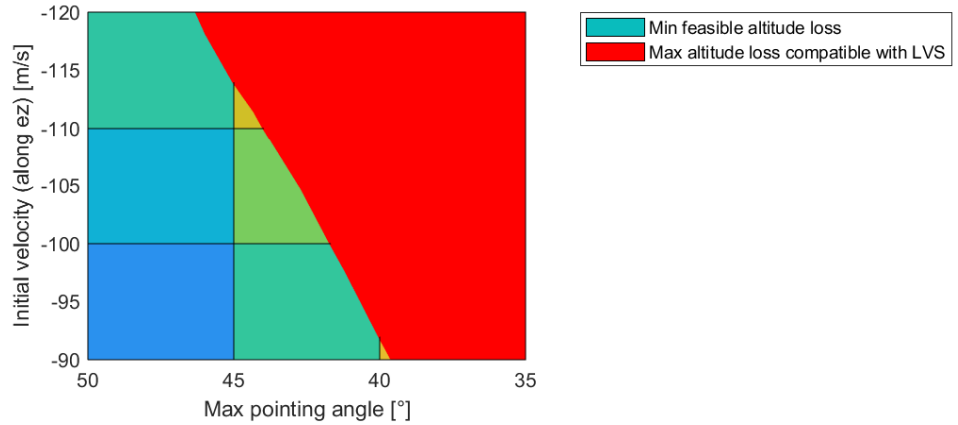
- **T2W=3.37**
  As we can observe in Figure 7.4, the best result is achieved by the smallest value of initial velocity $v_0 = -90$ m/s and the biggest value of maximum pointing angle $\gamma_p = 50°$. Worse results are obtained increasing the initial

velocity and decreasing the maximum pointing angle, with an almost linear relationship. The pairs of values that pass the test are:

$$(v_0, \gamma_p) = \{(-90{,}50), (-90{,}45), (-90{,}40), (-100{,}50), (-100{,}45), (-110{,}50), (-110{,}45), (-120{,}50)\}$$



(a) Results in the three dimensional space of the parameters.

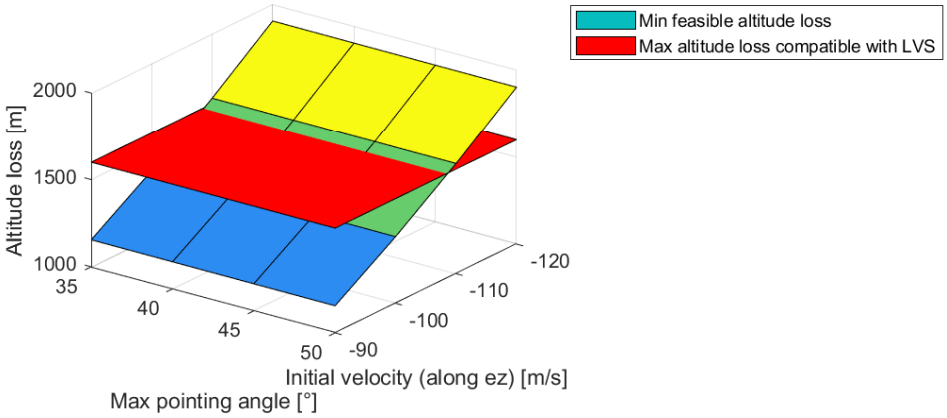


(b) Projection of the results on the $v_0 - \gamma_p$ plane, to highlight the combinations of initial velocity and maximum pointing angle compatible with the LVS.

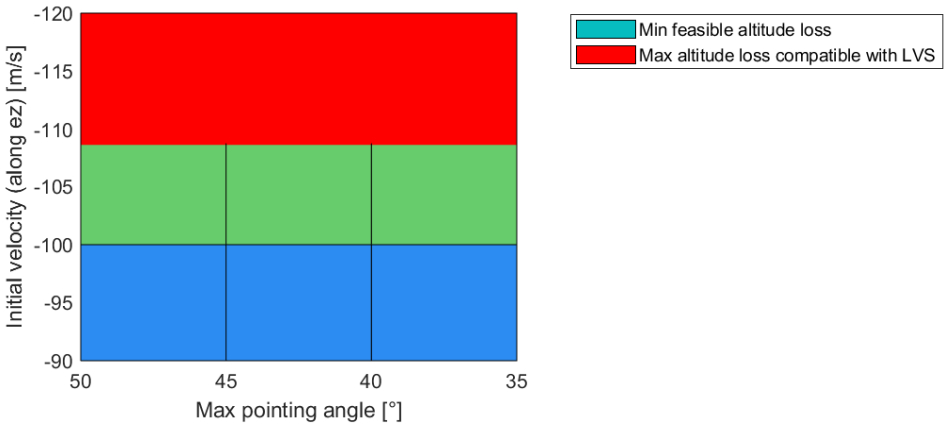
Figure 7.4: Minimum values of altitude loss obtainable as a result of a feasible optimization problem, for a set of initial velocity and maximum pointing angle values, with T2W=3.37. In red the maximum altitude loss compatible with the LVS.

- **T2W=2.53**
  As we can see in Figure 7.5, in this case the computed results are no longer related to the maximum pointing angle. There still remains an almost linear relationship between the altitude loss and the initial velocity. The pairs of values that pass the test are:

$$(v_0, \gamma_p) = \{(-90,50), (-90,45), (-90,40), (-90,35), (-100,50), (-100,45), (-100,40), (-100,35)\}$$



(a) Results in the three dimensional space of the parameters.



(b) Projection of the results on the $v_0 - \gamma_p$ plane, to highlight the combinations of initial velocity and maximum pointing angle compatible with the LVS.

Figure 7.5: Minimum values of altitude loss obtainable as results of a feasible optimization problem, for a set of initial velocity and maximum pointing angle values, with T2W=2.53. In red the maximum altitude loss compatible with the LVS.

7.2.3 Influence of the driver parameters on the results

We can give some observations about the relationship between the feasibility of the optimization problem and the driver parameters.

- **Initial velocity**
  The initial velocity mainly influences the dynamic of the spacecraft in the first phase of the divert maneuver, when the spacecraft carries out the initial braking.

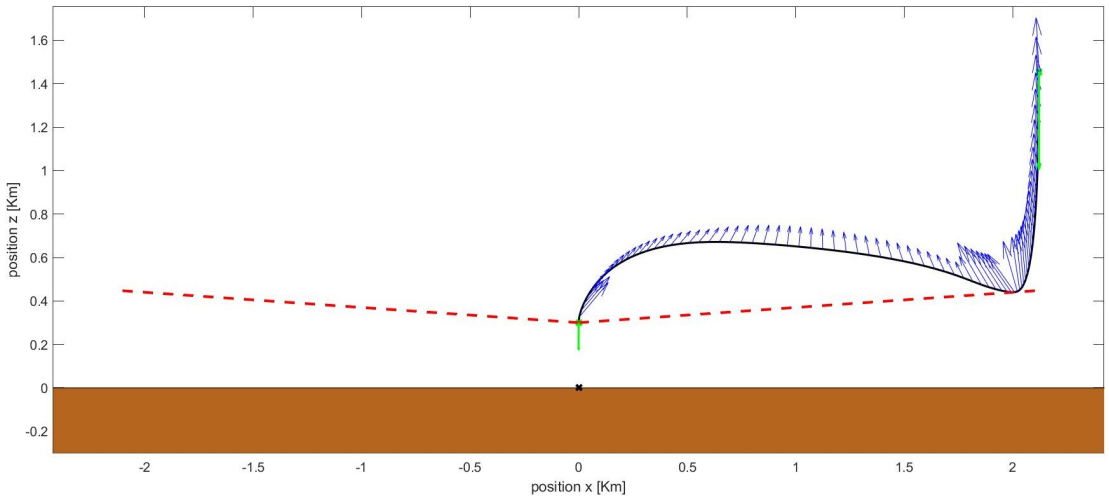
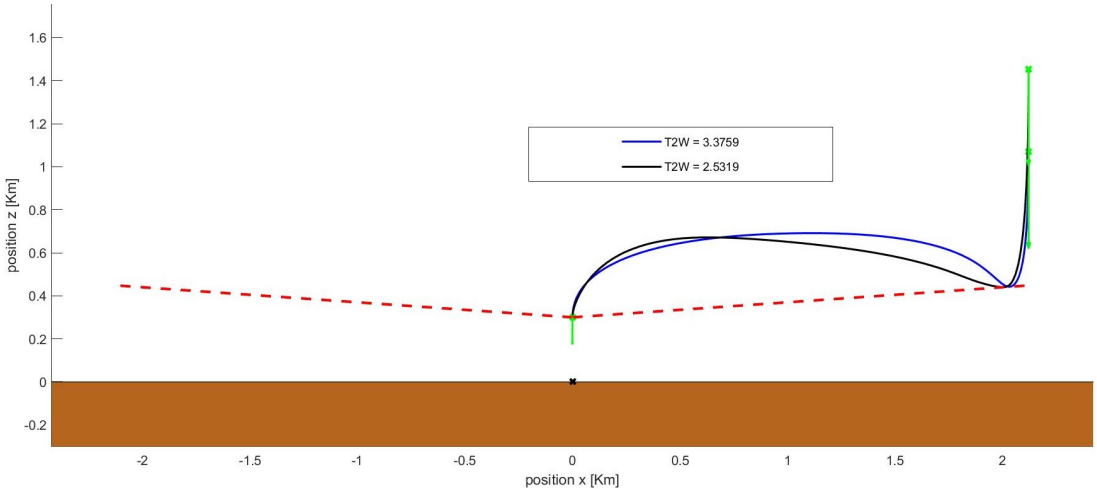
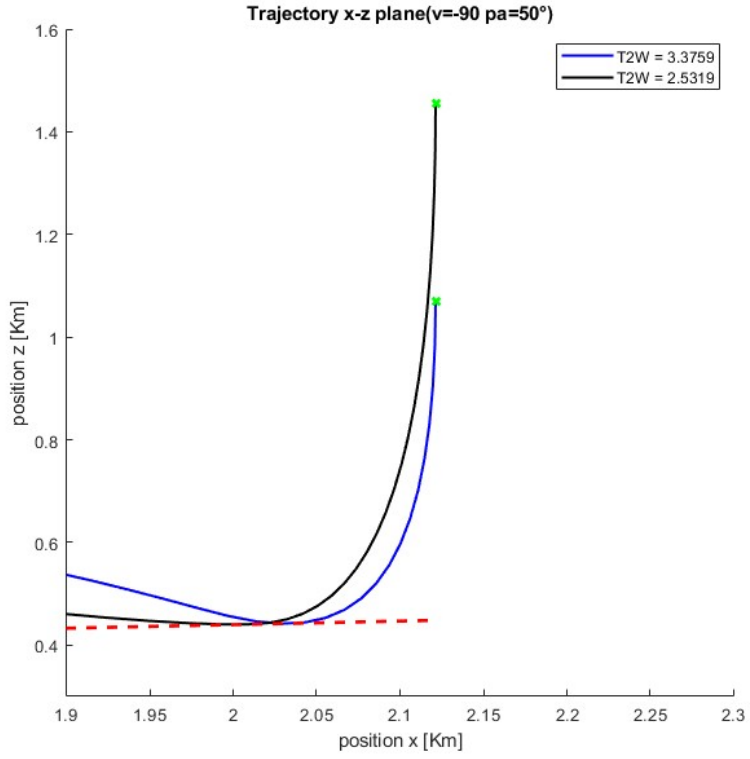


Figure 7.6: Divert maneuver trajectory on the xz plane, with T2W=2.53, $v_0 = -90$ m/s, $\gamma_p = 50°$ and minimum altitude loss. In black the simulation, in blue the thrust vector profile superimposed on the trajectory, in green the initial and final position and velocity, in red the glideslope constraint.

As we can see in the Figure 7.6, in a critical condition, where the optimization problem is about to become unfeasible, the initial braking is almost vertical and the spacecraft manages to stop the descent just before the forbidden area, delimited by the glideslope constraint. Thus, a smaller altitude loss would no longer provide enough space to slow down the lander in time, and therefore it would lead to an unfeasible problem. Then, cases with higher initial velocities need more space to slow down the lander and therefore greater altitude loss.



(a) Entire trajectories.



(b) Zoom on the start of the trajectories.

Figure 7.7: Divert maneuver trajectories comparison, between the two T2W cases, with $v_0 = -90$ m/s, $\gamma_p = 50°$ and minimum altitude loss. In black the simulation, in green the initial and final position and velocity, in red the glideslope constraint.

As we can see in Figure 7.7, also the value of T2W is relevant for the results obtained. Indeed, in the lower T2W case, the lander employs a bigger percentage of the available thrust to compensate the gravity force, reducing its capability to slow down the vehicle during the descent, leading to a bigger minimum altitude loss.

- **Maximum pointing angle**
  The influence of the maximum pointing angle on the feasibility of the optimization problem is strictly related to the T2W value. To understand why, firstly, we need to analyze the main forces applied to the spacecraft during the divert maneuver.

$$F_{TOT}(t) = T(t) + F_g(t) = T(t) - m(t)g_M\hat{e}_z \tag{7.8}$$

  Then, we need to focus on the last phase, when the lander is lighter $m \to m_{dry}$, and to compute the minimum value achievable on the z axis.

$$\begin{aligned} &\min\left(F_{TOT}(t_f)^T\hat{e}_z\right) = \\ &= \min\left(T(t_f)^T\hat{e}_z - m(t_f)g_M\right) \approx \rho_{min}cos(\gamma_p) - m_{dry}g_M \end{aligned} \tag{7.9}$$

  Therefore, for the two different cases of T2W

	ρ_{min} [N]	m_{dry} [kg]	γ_p [°]	$\min\left(F_{TOT}(t_f)^T\hat{e}_z\right)$ [N]
T2W = 2.53	14095.38	1100	35	-617.1
			40	-841.7
			45	-1090.9
			50	-1362.9
T2W = 3.38	18793.85	1100	35	537.5
			40	238.1
			45	-94.2
			50	-465.9

Table 7.1: Minimum values of the overall force along the z axis applied on the spacecraft at the end of the maneuver, for different values of T2W and γ_p.

In the table 7.1, we can notice the presence of two positive values in the last column. In these two cases, the overall force applied on the z axis at the end of the maneuver cannot reach negative values, i.e. the spacecraft can no longer accelerate downwards. This consideration has the meaningful implication to constraint the spacecraft on a monotonic decreasing trajectory, that limits

the guidance authority for planning the trajectory in the cases of low initial altitude.

We can observe the consequences of this implication in Figure 7.8, where the spacecraft with T2W=3.37 can no longer follow the black trajectory, where the spacecraft descends until the glideslope constraint to slow down the initial velocity of the lander, to increase later again its altitude and recover the horizontal error. In the T2W=3.37 case, this can no longer be done, because once increased again the altitude it would approach the last phase of the divert maneuver with an ascending speed that can no longer be redirected downward. Hence, the spacecraft is constrained to move along a monotonically decreasing trajectory and to achieve worse results.

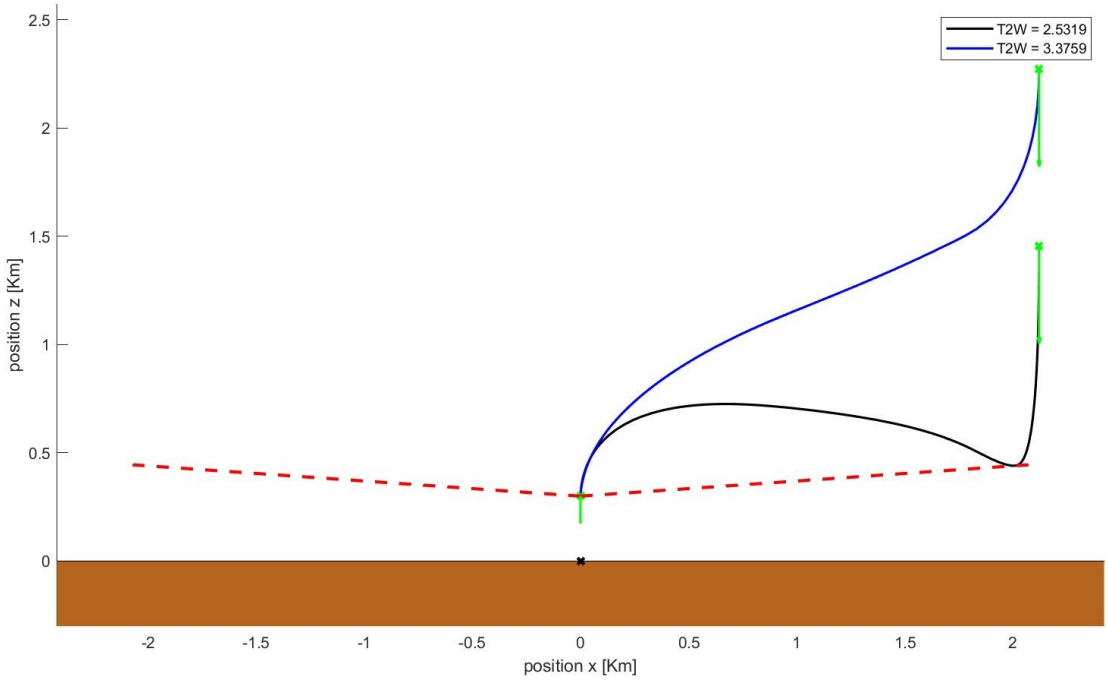
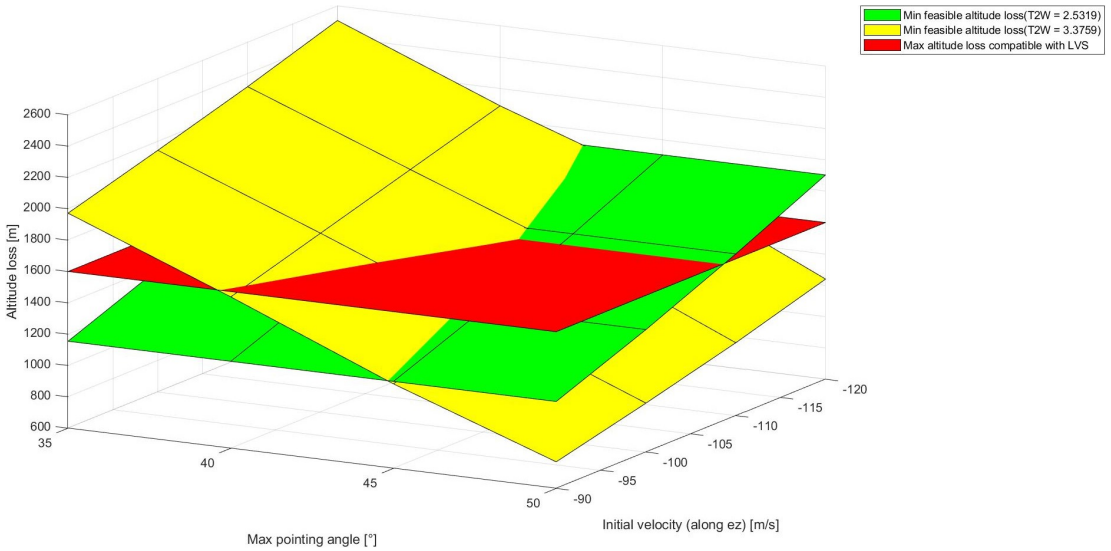


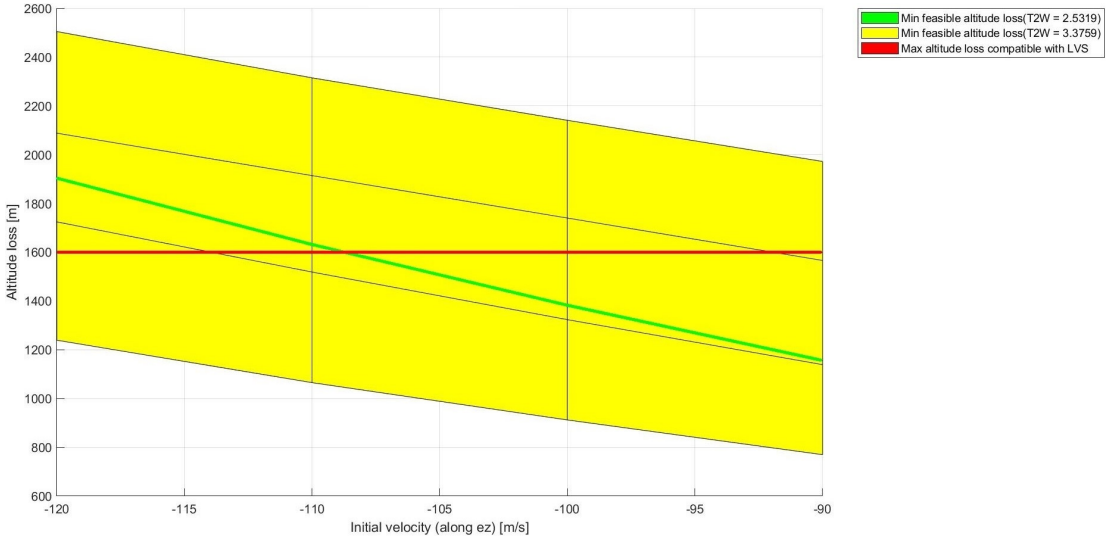
Figure 7.8: Divert maneuver trajectories comparison, between the two T2W cases, with $v_0 = -90$ m/s, $\gamma_p = 35°$ and minimum altitude loss. In black the simulation, in green the initial and final position and velocity, in red the glideslope constraint.

7.2.4 Comparison between different T2W cases

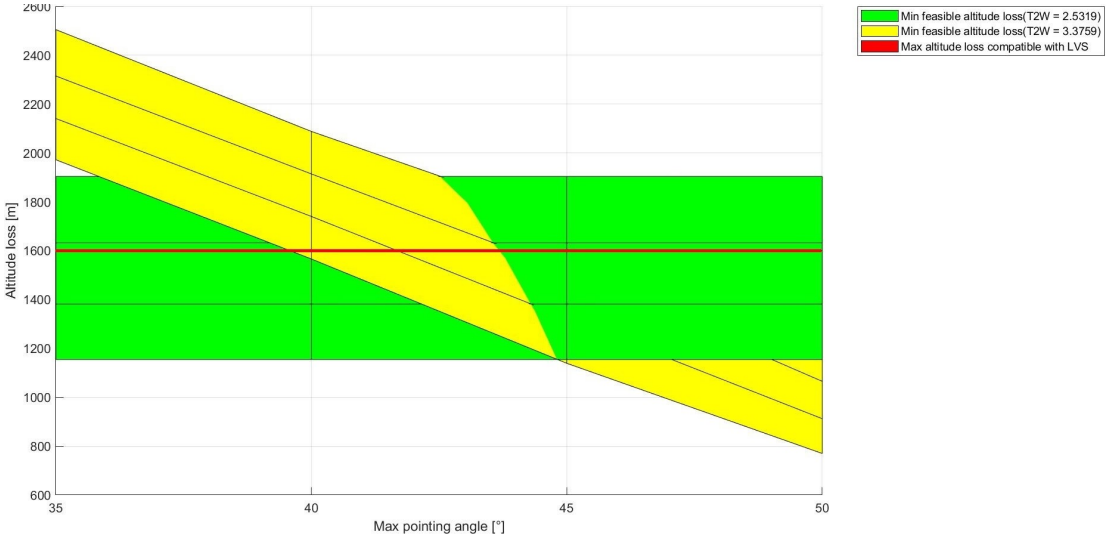
The results achieved in the two cases, with different T2W values, are compared in the Figure 7.9.



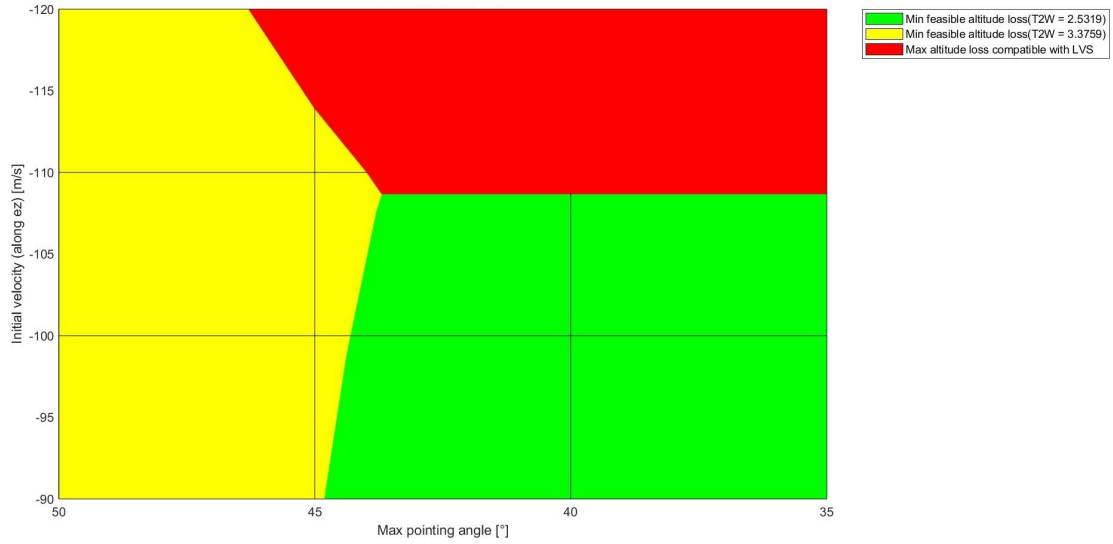
(a) Results in the three dimensional space of the parameters.



(b) Projection of the results on the $h - v_0$ plane.



(c) Projection of the results on the $h - \gamma_p$ plane.



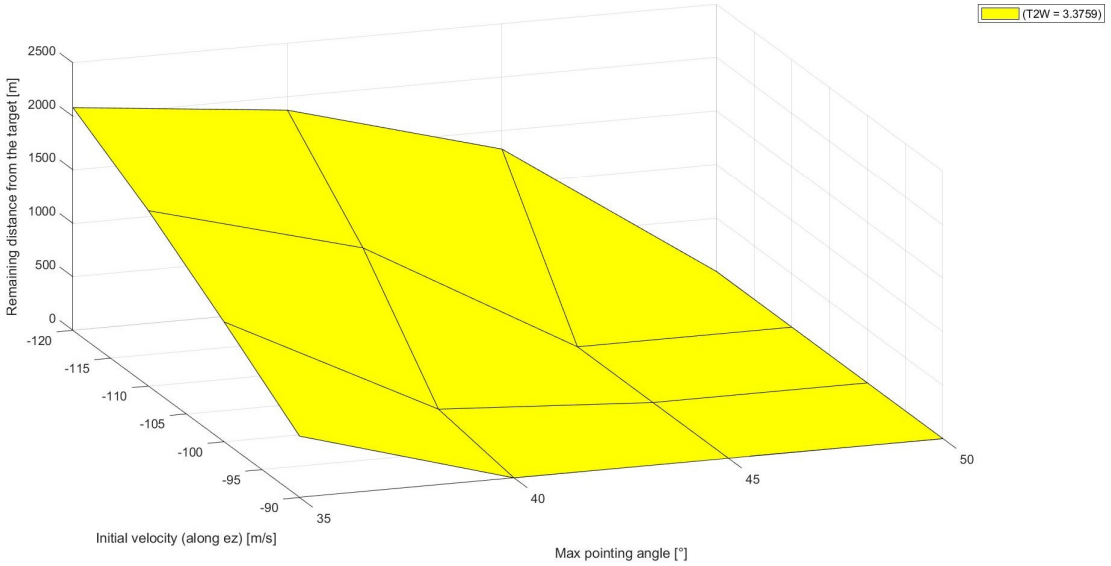
(d) Projection of the results on the $v_0 - \gamma_p$ plane, to highlight the best combinations of initial velocity and maximum pointing angle compatible with the LVS.

Figure 7.9: Comparison of the results obtained in the two T2W cases. In yellow and green the minimum values of altitude loss achievable as results of a feasible optimization problem, for a set of initial velocity and maximum pointing angle values. In red the maximum altitude loss compatible with the LVS.

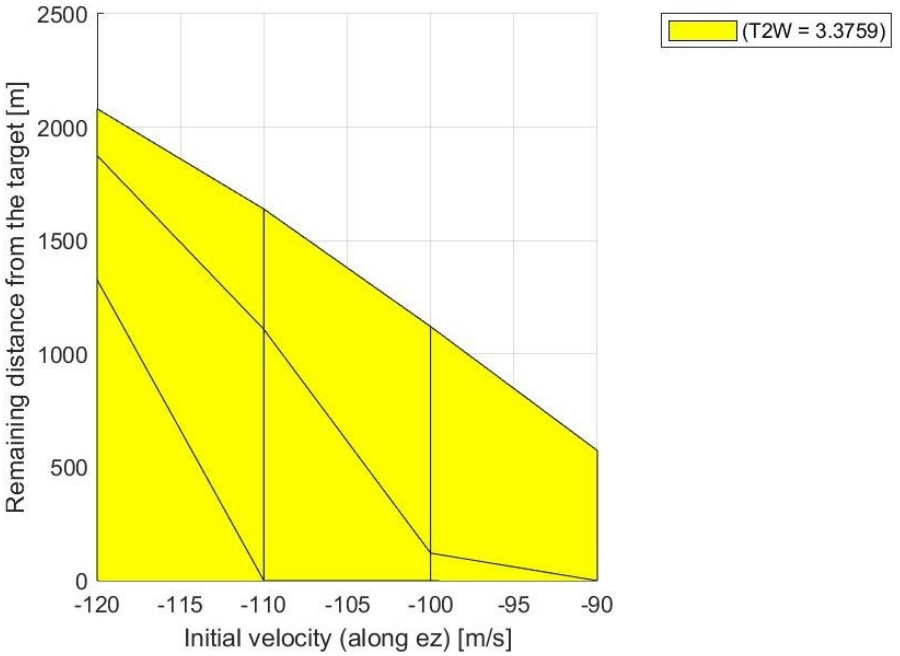
A comparison of the results obtained in the two T2W cases is represented in Figure 7.9d. Better outcomes are achieved for T2W = 2.53 with $\gamma_p = \{35°,40°\}$ and for T2W = 3.37 with $\gamma_p = \{45°,50°\}$. Whereas, in both cases the results get better with smaller initial velocities, as we can see in Figure 7.9b. These observations can be directly explained by the considerations previously made on the initial velocity and maximum pointing angle. Indeed, in almost all the cases the optimization problem becomes unfeasible due to the presence of the glideslope constraint. Therefore, the results obtained are mainly related to the initial velocity and the T2W value. With the exceptions of the cases with T2W=3.37 and $\gamma_p = \{35°,40°\}$, where the maximum pointing angle is too small and leads to a monotonically decreasing trajectory.

7.3 Identification of the performance limits

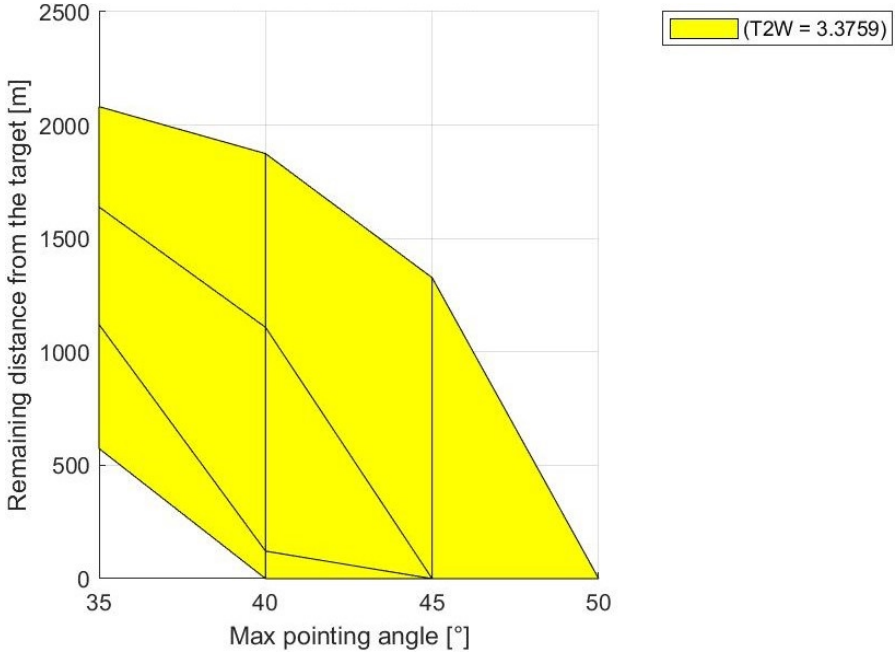
We now analyze those cases that are incompatible with the LVS requirement on the altitude loss. We run again the trajectory planning algorithm, with an altitude loss h of 1600 m, as required by the LVS. Obtaining sub optimal solutions of the problem. The performance of the results are compared, measuring the final distance from the target at the end of the divert maneuver.



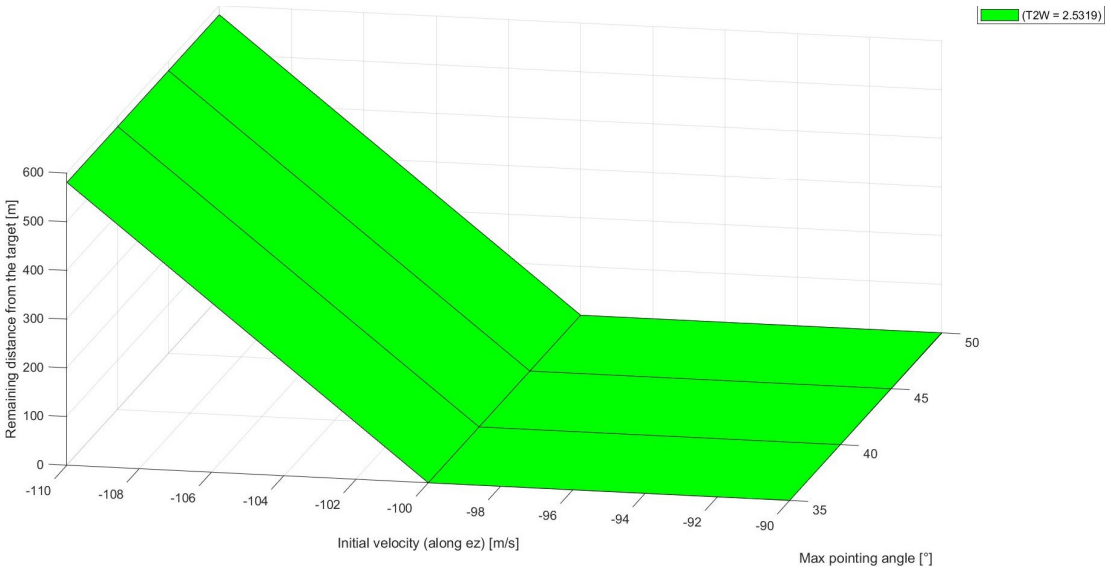
(a) Results in the three dimensional space of the parameters.



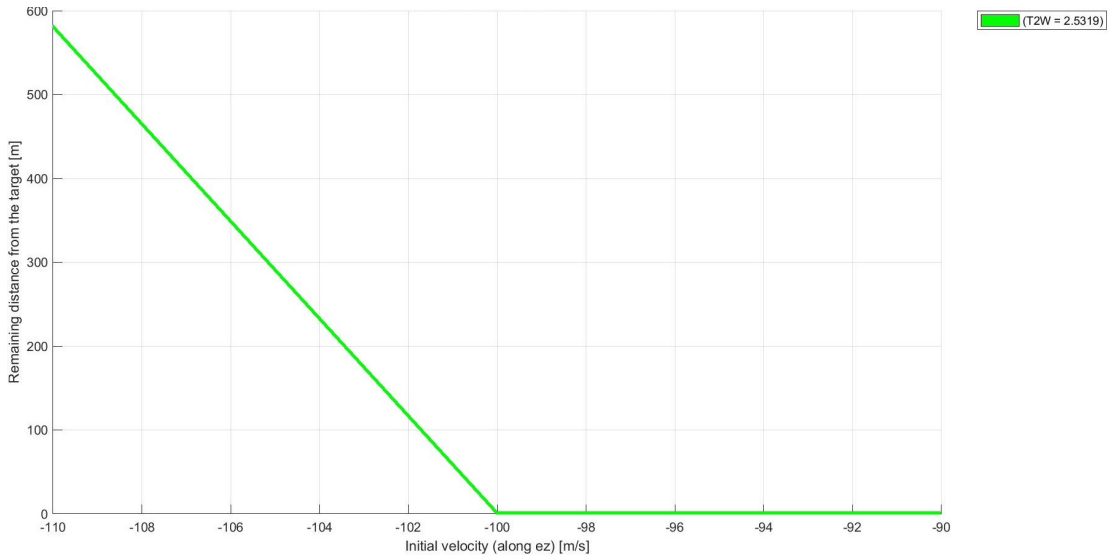
(b) Projection of the results on the $d - v_0$ plane.



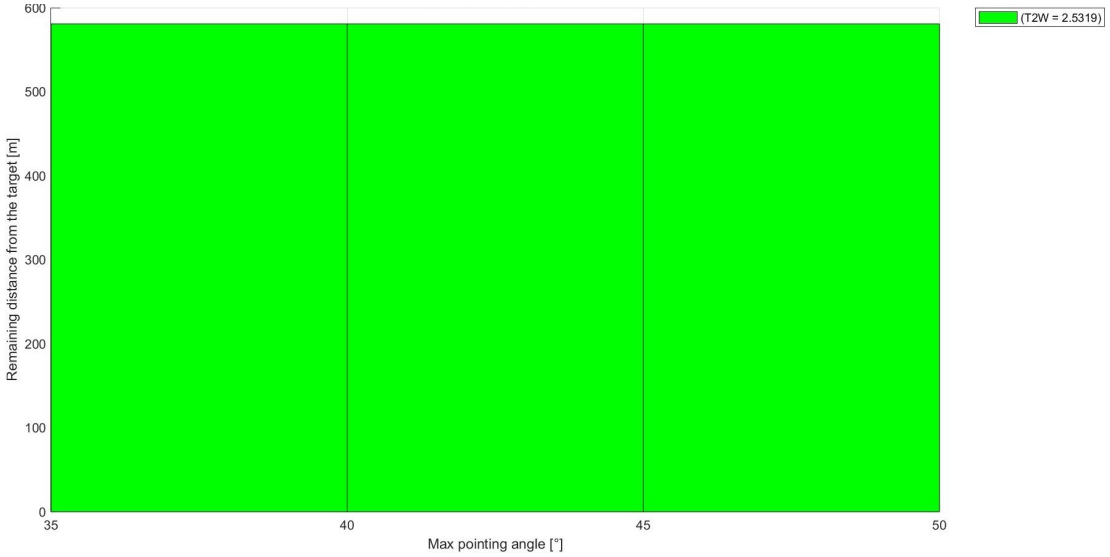
(c) Projection of the results on the $d - \gamma_p$ plane.



(d) Representation of the results in the three dimensional space of the parameters.



(e) Projection of the results on the $d - v_0$ plane.



(f) Projection of the results on the $d - \gamma_p$ plane.

Figure 7.10: Final distance from the target, for those cases that are incompatible with the LVS requirement on the altitude loss. Solutions of the optimization problem (2.60) with $\lambda = 1$.

As we can see in Figure 7.10, both T2W cases present the same result's pattern of the altitude loss parametric analysis. The same considerations done before can be applied also here. For T2W=2.53, the results are not related to the value of the maximum pointing angle. Whereas, for T2W=3.37, better results are achieved with smaller initial velocities and bigger maximum pointing angles.

In addition, it is worth to highlight the absence of any result in the case of T2W=2.53, with initial velocity $v_0 = -120$ m/s. Indeed, in this specific case, also the optimization problem for sub optimal solutions (2.60) results to be unfeasible. This aspect can be explained considering the space needed to completely arrest the descent of the spacecraft with an initial velocity of $v_0 = 120$ m/s and a reduced thrust available $T2W = 2.53$. The space required exceeds the altitude loss available $h = 1600$ m, even in the most favorable conditions, i.e. a completely vertical descent with the thrust always at the maximum. A simulation is proposed in Figure 7.11, where we can see that the final altitude, achieved at $v = 0$ m/s, is $h_N = 147.93$ m, that is less than 300 m as required.

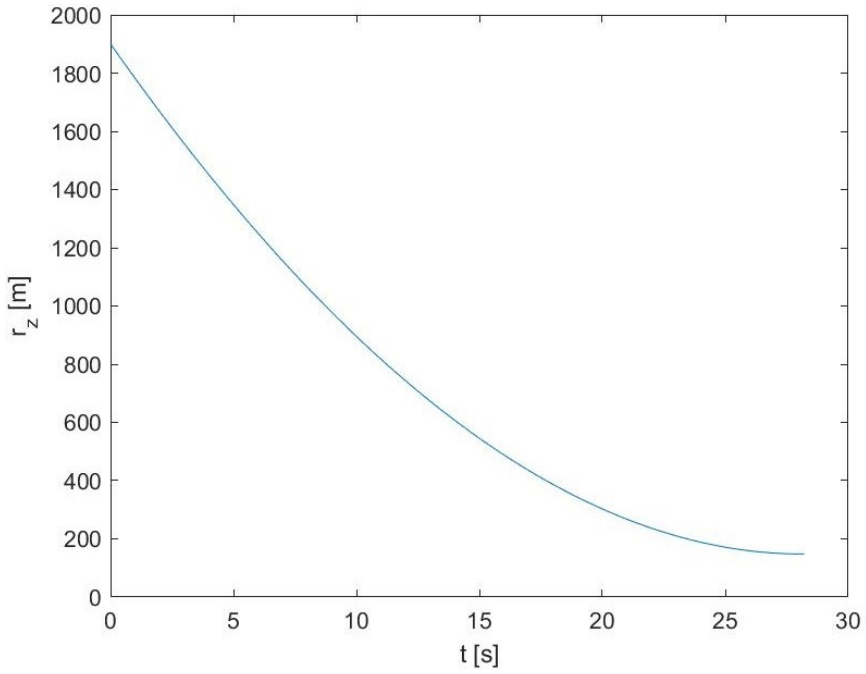


Figure 7.11: Altitude of the lander, in a vertical descent, during a deceleration of the motion with the maximum thrust available. In the case of T2W=2.53 and $v_0 = -120$ m/s.

Chapter 8

Conclusions

This thesis work aims to analyze and implement the guidance problem for a pinpoint landing on Mars, i.e. autonomously guide the spacecraft from his initial location toward the desired landing site, with an accuracy of less than 100 m. The developed algorithm will have to provide a reliable and accurate solution of the guidance problem in real time. Moreover, a sub optimal solution has to be computed in case of an unfeasible guidance problem, to ensure in any case a safe landing of the spacecraft.

Firstly, a survey of analytical and numerical methods, used or proposed over time to solve this type of problem, is carried out. We identify the Lossless convexification (LCvx) as a very promising method and choose it for the development of this thesis work. This choice is based on the fact, that LCvx provides a proven way to convexify the guidance problem, originally described as a non convex optimization problem. The obtained formulation assures at the same time the convergence to the optimal solution and an efficient computation of the result, essential for real time applications.

The first contribution of this thesis is to extend the optimization problem proposed in literature [9] to assure sub optimal, but feasible, solutions also in case of unfeasible optimization problems. For example, this could be required for landing sites too far or initial velocities too high, available fuel too low and so on. Thus, sub optimal solutions will not reach the desired landing site, but will assure a safe touch down of the spacecraft, optimizing at the same time the final distance from the target and the fuel consumption.

Then, a trajectory planning algorithm is designed to implement the mathematical formulation of the guidance problem, derived previously. The developed algorithm is written in Matlab code, which exploiting specific solvers, based on interior point methods (IPMs), efficiently and rapidly find an optimal solution of the problem.

The tests performed on some meaningful case studies prove the performance of the algorithm. Optimal solutions of feasible optimization problems reach the

desired landing site with an optimal fuel consumption. Sub optimal solutions of unfeasible optimization problems optimize at the same time fuel consumption and final distance from the target. A weight parameter can be tuned to choose which criterion to optimize over the other. A Monte Carlo experiment, with 100 cases, proves the robustness of the algorithm within nominal working conditions. Finally, a simulation of the entire landing maneuver proves the performance of the developed algorithm as part of a more structured algorithm, that manages guidance, navigation and control of the spacecraft throughout the landing.

At the end, a parametric analysis of the trajectory planning algorithm provides correlations between the behavior of the developed algorithm and a selection of significant design parameters of the lander, such as the configuration of the Radar Doppler Altimeter, the performance of the parachute and the requirements of the Landing Vision System. This last analysis suggests useful guidelines for the spacecraft design choices.

8.1 Future work

1. The Lossless convexification theory, used in this thesis work to formulate the guidance problem as a convex optimization problem, is a research field still under development. The LCvx cannot be applied to a generic non convex optimization problem. Indeed, only some specific classes of cost function and constraints can be used. For this reason, a significant assumption is done in the formulation of the guidance problem, i.e. we neglect the attitude dynamics, in favor of the translational dynamics. This choice can be motivated, considering the attitude dynamics as a direct result of the translation dynamics. Indeed, decoupling the guidance problem, the reference attitude can be derived by the pointing angle of the thrust vector. However, this assumption leads to the following drawbacks:

    - We could compute an unfeasible attitude dynamic. Indeed, limitations on the physical capability of the thrusters could constrain the maximum attitude rate achievable.
    - The attitude dynamics requires setting aside a percentage of the available thrust from the translation dynamics.
    - Since we cannot define the initial and final attitude of the spacecraft in the guidance problem, an initial rotation is necessary to bring the lander to the initial attitude required by the divert maneuver and a final rotation is necessary to verticalize again the spacecraft at the end of the divert maneuver. These rotation maneuvers subtract time, while the spacecraft is falling down, forcing to start the landing maneuvers higher.

Future works on the LCvx theory could enlarge the classes of constraints that can be used, letting to add the attitude dynamics to the guidance problem and solving the previously mentioned problems.

2. The discretization of the guidance problem, essential to solve numerically the optimization problem, leads to an approximation of the original problem. Large sample times could lead to approximations of the original problem that no longer fulfill the requirements of the LCvx theory. In these cases, a feasible solution of the approximated problem is computed, which however results to be unfeasible for the original problem. The unfeasibility derives from the violation of the thrust constraints.

   Future works could identify a maximum admissible sampling time, or implement an exception procedure in case of unfeasibility of the original optimization problem. This activity will also account the interaction between guidance and control in terms of acceptable roughness of the reference profile and associated mitigation actions.

Bibliography

[1] Lars Blackmore. «Autonomous precision landing of space rockets». In: *Frontiers of Engineering: Reports on Leading-Edge Engineering from the 2016 Symposium*. Vol. 46. The Bridge Washington, DC. 2016, pp. 15–20 (cit. on p. 3).

[2] Bradley A Steinfeldt, Michael J Grant, Daniel A Matz, Robert D Braun, and Gregg H Barton. «Guidance, navigation, and control system performance trades for Mars pinpoint landing». In: *Journal of Spacecraft and Rockets* 47.1 (2010), pp. 188–198 (cit. on pp. 4, 7, 10).

[3] Enrico Canuto, Jose Ospina, and Marcello Buoncore. «Model-based guidance and control for atmospheric guided entry». In: *AIAA Guidance, Navigation, and Control Conference*. 2012, p. 4840 (cit. on pp. 5, 67).

[4] Scott Ploen, Bechet Acikmese, and Aron Wolf. «A comparison of powered descent guidance laws for mars pinpoint landing». In: *AIAA/AAS astrodynamics specialist conference and exhibit*. 2006, p. 6676 (cit. on p. 7).

[5] Ingo Gerth and Erwin Mooij. «Guidance for autonomous precision landing on atmosphereless bodies». In: *AIAA Guidance, navigation, and control conference*. 2014, p. 0088 (cit. on p. 7).

[6] Stephen Boyd, Stephen P Boyd, and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004 (cit. on pp. 12, 14, 19).

[7] Behcet Acikmese and Scott R Ploen. «Convex programming approach to powered descent guidance for mars landing». In: *Journal of Guidance, Control, and Dynamics* 30.5 (2007), pp. 1353–1366 (cit. on pp. 25, 26, 31, 32).

[8] Behçet Açıkmeşe, John M Carson, and Lars Blackmore. «Lossless convexification of nonconvex control bound and pointing constraints of the soft landing optimal control problem». In: *IEEE Transactions on Control Systems Technology* 21.6 (2013), pp. 2104–2113 (cit. on p. 27).

[9] Danylo Malyuta, Taylor P Reynolds, Michael Szmuk, Thomas Lew, Riccardo Bonalli, Marco Pavone, and Behcet Acikmese. «Convex optimization for trajectory generation». In: *arXiv preprint arXiv:2106.09125* (2021) (cit. on pp. 27, 30, 31, 41, 52, 56, 81, 84, 101).

[10] Kim-Chuan Toh, Michael J Todd, and Reha H Tütüncü. «On the implementation and usage of SDPT3–a Matlab software package for semidefinite-quadratic-linear programming, version 4.0». In: *Handbook on semidefinite, conic and polynomial optimization*. Springer, 2011, pp. 715–754 (cit. on p. 43).

[11] Farid Alizadeh, Jean-Pierre A Haeberly, and Michael L Overton. «Primal-dual interior-point methods for semidefinite programming: convergence rates, stability and numerical results». In: *SIAM Journal on Optimization* 8.3 (1998), pp. 746–768 (cit. on p. 43).

[12] Sanjay Mehrotra. «On the implementation of a primal-dual interior point method». In: *SIAM Journal on optimization* 2.4 (1992), pp. 575–601 (cit. on p. 43).

[13] Inc. CVX Research. *CVX: Matlab Software for Disciplined Convex Programming, version 2.0*. `http://cvxr.com/cvx`. Aug. 2012 (cit. on p. 44).

[14] Mykel J Kochenderfer and Tim A Wheeler. *Algorithms for optimization*. Mit Press, 2019 (cit. on p. 47).