TÉCNICO LISBOA

Robust Funnel Synthesis Algorithms for the 6-DOF Powered Descent Guidance Control Problem

Guilherme João Verdasca Videira

Thesis to obtain the Master of Science Degree in

Mechanical Engineering

Supervisors: Prof. João Sousa
Prof. Tamas Keviczky

Examination Committee

Chairperson: Prof. Duarte Pedro Mata de Oliveira Valério
Supervisor: Prof. João Miguel da Costa Sousa
Member of the Committee: Prof. Rui Miguel de Moura Antunes e Valejo Coelho

June 2024

”Exploration knows no bounds. In space, we find our greatest challenges, our most profound mysteries, and our greatest hope.”

— *Carl Sagan*

Declaration

I declare that this document is an original work of my own authorship and that it fulfills all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

Acknowledgments

As an important phase of my academic journey is coming to an end, it is paramount to present my acknowledgements to the people that played a crucial role in my life.

First of all, I want to thank my parents for being by my side for the last 24 years. You were a huge support and you gave me way more than I could ever ask for. I want to share this academic achievement with you, because, even though I was stubborn and tiresome sometimes, you had the patience to guide me towards the right path. I became the man that I am today because of you, I hope you feel that I was worth all the headaches and sleepless nights.

To both my sisters, thank you for being who you are and teaching me so many valuable lessons. We had our share of fights and disagreements, but we are brothers for life and that will never change. I felt your support along every step of the way, and I love you a lot. I am happy that I will have to put up with you throughout many years of my life.

Then, thank you to the most important person that I have got to know in these university years, my girlfriend Mónica. When I first met you, I would have never guessed how much we would grow together. You are the kindest and most cheerful person that I know, and you always made me smile when I was with you. Even when I was stressed with this thesis project, spending time with you would light up my day and remind me of the most important things in life. We have so many great memories together and I am sure we will continue to enjoy valuable moments with one another.

To my six grandparents, even though two are no longer with us, a huge thank you. You have done so many sacrifices throughout my life to raise me and to help me become the man I am today. You have always believed in my potential and I know that you get genuinely happy with my achievements. You have taught me so many lessons, and you will always have a special place in my heart.

A word of appreciation for all the friends that I have made throughout this academic journey. The best years of my life were shared with you, and you were responsible for making this experience much more enjoyable. I hope that, in the future, I can spend more time with most of you.

Finally, I would like to thank Professor João Sousa, from Instituto Superior Técnico, and Professor Tamas Keviczky, from the Technical University of Delft, for supervising this thesis project, providing insightful orientation and guiding me throughout this challenge.

Resumo

À medida que a humanidade entra numa nova era de exploração espacial, onde o potencial de foguetões reutilizáveis tem sido demonstrado por empresas privadas, o mundo académico ganhou um novo interesse em algoritmos de aterragem autónoma. A parte final do procedimento de aterragem é denominada de Powered Descent Guidance (PDG), onde, usando os mecanismos de controlo disponíveis, o veículo precisa de fazer correções de posição e orientação para aterrar de maneira segura e suave. Ao analisar a literatura publicada sobre este tópico, pode encontrar-se o recurso a estratégias de convexificação para calcular, de maneira rápida e precisa, trajetórias de referência. Para além disso, os chamados métodos de síntese de conjuntos invariantes têm sido aplicados ao problema de controlo de PDG, com o objectivo de acrescentar garantias de robustez, ao optimizar um controlador que permite corrigir desvios da trajetória de referência. Contudo, tais estratégias não são capazes de mostrar uma boa performance quando factores de perturbação inesperados são acrescentados ao modelo dinâmico. Por essa razão, nesta tese, duas abordagens diferentes, ambas baseadas na síntese de conjuntos invariantes, são propostas para diminuir este problema. A primeira é um algoritmo de controlo preditivo onde os parâmetros do conjunto invariante são usados para formular uma função de custo e uma restrição terminal. A vantagem de incluir os referidos parâmetros é demonstrada, ao fazer uma comparação com o típico problema de controlo preditivo. A segunda abordagem é um algoritmo que permite o recálculo rápido, em tempo real, da trajetória de referência e do conjunto invariante.

Palavras-chave: Síntese de Conjuntos Invariantes, Controlador em Anel Fechado, Controlo de Aterragem, Robustez, Trajetória de Referência, Recálculo em Tempo Real, Controlo Preditivo

Abstract

As humankind enters a new era of space exploration, where the potential of reusable launch vehicles has been showcased by SpaceX and Blue Origin, the academic world has renewed its interest in autonomous landing algorithms. The terminal and most important phase of the landing sequence is commonly referred to as Powered Descent Guidance (PDG), where, using the available control mechanisms, the spacecraft needs to make final position and orientation corrections so it can land safely and smoothly. Throughout the literature published on this topic, convexification strategies have been proposed to compute, in an accurate and computationally efficient way, a reference landing trajectory. Furthermore, so-called funnel synthesis approaches have been applied to the PDG optimal control problem in order to provide additional robustness guarantees, by associating a control invariant set and a feedback controller to the reference trajectory and correcting deviations. However, such strategies fail to have an acceptable performance when unexpected additive random uncertainty and disturbance factors are considered in the dynamical model. For that reason, in this thesis project, two different approaches, both based on the funnel synthesis framework, are proposed to mitigate such issue. The first one is a Model Predictive Control (MPC) algorithm where the funnel parameters are leveraged to formulate a cost function and a terminal constraint. The advantage of including the funnel parameters in the optimization problem is shown, by comparing the corresponding performance with a standard MPC controller. The second approach is an algorithm that enables fast real-time reference trajectory and funnel recomputation.

Keywords: Funnel Synthesis, Feedback Controller, Powered Descent Guidance, Robustness, Reference Trajectory, Real-Time Recomputation, Model Predictive Control

Contents

List of Tables

List of Figures

Nomenclature

Greek symbols

α	Decay rate
η	State difference variable
γ	Lipschitz constant
λ	$\mathcal{S}$-procedure multiplier variable
ν	$\mathcal{S}$-procedure multiplier variable, where $\nu = \lambda^{-1}$
ω	Angular velocity vector
ϕ	Nonlinear channel function
ψ	Yaw angle
τ	Torque vector
Θ	Euler angle vector
θ	Pitch angle
ε	Ellipsoid
φ	Roll angle
ξ	Control input difference variable
ζ	Funnel contraction factor

Roman symbols

$\mathcal{F}$	Quadratic funnel
A	Aerodynamic force vector
C	Direction cosine matrix
F	Thrust force vector
K	Time-varying feedback controller gain

m Mass variable

Q Positive-definite matrix that defines the funnel parameters

r Position vector

u Control input vector

V Lyapunov function

v Linear velocity vector

w Disturbance vector

x State vector

Subscripts

$\mathcal{B}$ Relative to the body-fixed frame

$\mathcal{I}$ Relative to the landing site fixed frame

Superscripts

T Transpose

Chapter 1

Introduction

The first chapter of the following thesis document serves as an initial step to better understand the problem tackled throughout this thesis project. A brief introduction to the topic is conducted in Section 1.1, while the key motivation behind the choice to focus on this subject is explicit in Section 1.2. The main objectives to be achieved at the end of the current thesis project are defined in Section 1.3. Finally, the structure of the thesis document is outlined in Section 1.4.

1.1 Topic Overview

With the increased number of autonomous space missions bound for the Moon and Mars on the horizon, the focus of space exploration agencies and companies has shifted, in the most recent years, towards developing guidance and control algorithms that enable safe and precise landings. The pursuit of autonomous and precise landing capabilities can lead to great benefits for the future of the industry, such as decreased spacecraft cost and increased success rate of space missions, and would be a milestone in the journey to establish permanent human outposts beyond Earth [1].

One important part of the precision landing procedure is the Powered Descent Guidance (PDG) phase, which is the terminal segment of the entry, descent and landing sequence. The goal is to transition a vehicle from its initial state to a designated landing site using the available control systems, which often include a gimbaled rocket engine and a mechanism to apply additional torque. At the beginning of this stage, the vehicle has already largely decelerated from orbital speeds and, therefore, descends with smaller velocity values [2]. At the end of the PDG phase, the vehicle should land in a predefined location, with zero or near to zero velocity and in the upright position.

For this purpose, an optimal control problem is formulated in order to find the minimum fuel trajectory that takes the space vehicle from the initial to the final state. This optimization problem requires the definition of the spacecraft dynamical model and the constraints that it is subject to. The main challenge associated with the reference trajectory computation procedure is related with the complexity and nonlinearity of the given dynamical model, as well as the nonconvexity of some of the constraints [3]. Finding the optimal solution of nonconvex nonlinear optimization problems efficiently is still an open research

topic, and the algorithms proposed for that purpose are usually viewed as a trade-off between accuracy and runtime complexity [4]. Therefore, in a scenario where computational power is limited and a quick trajectory readjustment may be required to avoid hazards, a fast approach should be prioritized.

Across the years, from the advent of space exploration to today, many scientists and academic researchers have worked towards the improvement of trajectory planning algorithms and landing control methods, as well as the mitigation of the aforementioned issues. From the polynomial guidance laws used during the Apollo missions to the modern state-of-the-art convexification strategies, the field of autonomous landing algorithms has undergone a remarkable advance. Nevertheless, there is an area in which such control approaches still need to improve: its robustness and the ability to handle uncertainty and unforeseen disturbances. This is precisely the topic which the present thesis project focuses on.

1.2 Motivation

The field of autonomous landing algorithms is witnessing a renewed interest. Such enthusiasm is driven not only by the preparation of space missions to the Moon and Mars, but also by the successful demonstrations carried by private space companies on the potential of reusable launch vehicles. As explained above, achieving precise autonomous landing capabilities, besides being a step towards increased safety and reliability, has one major advantage: the decrease of space exploration cost. The main reason why the space industry has required such large investments across the years is related with the cost of designing and building rockets, which, until recent years, were single-use structures [5]. However, a fully reusable launch vehicle model, which inherently requires autonomous landing algorithms for its recovery, would help the space industry lower the costs in this area. Lower costs could, consequently, become a driving factor for further investment towards space exploration.

Nonetheless, accomplishing such task is not straightforward. Determining a landing trajectory and controlling the spacecraft along the descent procedure has its challenges. First of all, the dynamical model is nonlinear and highly complex, especially due to the coupling between the translational and rotational motions of the vehicle. The algorithm must take into account the 6 Degrees-of-Freedom (DOF) of the spacecraft, while considering a large set of state and input constraints. Additionally, each landing zone has its own characteristics and may require distinct mission specifications. For that reason, the guidance and control algorithms must be capable of guaranteeing a safe and accurate landing for a wide range of scenarios.

Besides that, there is another factor to be considered, one of vital importance for the success of space missions: the ability to land softly and safely even in the presence of external disturbances. It is certainly guaranteed that the vehicle, when performing the landing procedure, faces uncertain factors, not taken into account when the trajectory generation problem is formulated [1]. These factors can include model-plant mismatch, faulty sensor readings or unpredictable weather conditions, whether it is wind or atmospheric drag. Such unforeseen perturbations can make the vehicle deviate from the reference trajectory, and, consequently, the corresponding predefined control actions become suboptimal. In some cases, where the deviation becomes larger, they can even lead to infeasible scenarios. Hence, it

is important to make sure that the state-of-the-art PDG algorithms are able to adjust their control actions in the face of uncertainty and disturbances, not compromising the safety and feasibility of the landing procedure.

Across the literature, many authors have worked on the landing trajectory generation problem, proposing convex optimization methods in order to minimize the computational burden required for that purpose. However, these do not take into account the disturbances certainly encountered when landing in real-world scenarios and, consequently, are not sufficiently robust [6]. The following thesis project looks to mitigate this limitation, and the corresponding specific objectives are outlined in the next Section.

1.3 Objectives

The main aim of the present thesis project is to study and improve a current state-of-the-art algorithm for the PDG control problem, by understanding how different formulations affect the robustness results and by tackling the corresponding limitations. Two novel approaches are proposed in order to, through fast real-time replanning of the control actions, make the system more robust and pave the way for a better controller performance.

The algorithm to be studied is based on a funnel synthesis approach. A funnel is a control invariant set, which implies that, if a system's initial condition is located inside its entry, it is mathematically guaranteed that there exists at least one control input that keeps the state inside the funnel for all future times [7]. The funnel synthesis approach allows to simultaneously compute the funnel parameters and an associated feedback controller in a single optimization problem.

Funnel synthesis is a control strategy that has proved to be real-world implementable due to its increased computational efficiency. The majority of the computational burden is placed on the offline phase, long before the vehicle initiates the landing procedure, and, in real-time, a feedback controller is capable of quickly determining the suitable control actions [8].

Additionally, funnel synthesis adds a robustness element to the controlled system, because, even when the spacecraft is largely deviated from the reference, the feedback controller is able to perform corrections and guarantee that a wide set of initial conditions eventually leads to the goal state [8]. Due to the presence of uncertainty and external disturbances, it is almost guaranteed that, when initiating the PDG phase, the vehicle's true initial condition will be different from the reference's one. For that reason, it is important to opt for a control strategy which mathematically guarantees that, even in those cases, the system converges to the goal state. Funnel synthesis fills the above requirement.

Despite that, there are benefits in using real-time information regarding trajectory evolution to adjust the control actions and, consequently, decrease the direct influence of the disturbances on the system. This detail has not been explored properly across the literature, and, therefore, the goal of the present thesis project is to work on this topic.

Three specific objectives are defined:

1. Implement the current state-of-the-art 6-DOF PDG control strategies and study if they are able to

effectively control a system where the direct influence of uncertainty and disturbances is considered within the simulation model;

2. Understand the impact of different funnel synthesis formulations on the controller's ability to reject external perturbations;

3. Propose two new different approaches to increase the robustness of the current state-of-the-art 6-DOF PDG strategies. The first one is based on a Model Predictive Control (MPC) framework, while the second considers real-time trajectory and funnel recomputation steps.

1.4 Thesis Outline

In Chapter 1, a brief introduction to the topic and the corresponding relevance for the scientific community, as well as the specific objectives to be accomplished at the end of this thesis project, were presented. In Chapter 2, a detailed look into the history of the development of autonomous landing algorithms is provided, starting with the Apollo missions and, then, diving deeper into the current state-of-the-art control methods. In Chapter 3, the spacecraft's dynamical model, the constraints it is subject to and all the mathematical formulations behind the control strategies used in this thesis project are explained in detail. In Chapter 4, the most important algorithm results are shown and briefly discussed. Finally, in Chapter 5, key conclusions are drawn according to the obtained results and future steps are proposed in order to further improve the algorithm.

Chapter 2

Related Work

The second chapter is an introduction for those who may not have extensive knowledge of the history behind the implementation of autonomous landing procedures. It aims to provide background information and get the reader acquainted with the algorithms developed along the years with the purpose of advancing precision landing capabilities. Essentially, the main goal is to understand the past work that has contributed to the advancement of the knowledge in this area, as well as further outline the practical significance of the conducted research.

Several published articles are reviewed throughout this chapter. From the Apollo missions' period to modern days, the evolution from polynomial guidance laws to fast and reliable 6 Degrees-of-Freedom (DOF) convexification strategies is discussed. Additionally, the advantages and limitations of the state-of-the-art Powered Descent Guidance (PDG) control methods are outlined, paving the way for the following thesis project.

2.1 Landing Algorithms in the Apollo Era

The journey through autonomous precision landing technologies starts with the Apollo missions, where the guidance system was designed to be able to land the spacecraft without ground assistance [9, 10]. It could, autonomously, process sensor data, perform the necessary computations and provide the crew with the right control actions to land safely. Apollo system was, therefore, one of the first to be equipped with full onboard capability to perform the full landing procedure.

Apollo guidance system assumes that the lunar module is a 3-DOF system, by only considering the translational motion of the vehicle when defining a landing trajectory [11]. The spacecraft's attitude is then controlled by a faster inner control loop, computing the necessary torque commands in a cascade architecture. The reality is that, even though developments in the PDG research field have been made throughout the years, by the beginning of the 21st century, many real missions still relied on landing algorithms derived from the Apollo guidance system [2, 12].

Guidance algorithms for descent and landing procedures are composed by two elements: trajectory generation and real-time control tracking. For the trajectory generation phase, Apollo used an iterative

numerical optimization scheme [13]. The reference trajectory was parameterized by only two fourth-order polynomials, which constitutes a major limitation, as this type of parameterization cannot represent a generic trajectory [14].

An extensive amount of research has been developed and published regarding the 3-DOF Apollo guidance system. The work of D. F. Lawden, published in 1963, is often considered a relevant first insight into the optimality of 3-DOF trajectories [15]. The author developed a series of analytical expressions that could provide the optimal solutions for 3-DOF rocket systems' control problems, whether it is the maximization of its range or computation of the optimal thrust profile. Nevertheless, at the time, the computing power was limited and numerically obtaining an accurate solution for such equations was a challenging task. The first tractable numerical algorithm that could efficiently solve the minimal fuel problem for the terminal landing phase was proposed in 1964 [16]. It resorted to the Pontryagin maximum principle to determine the optimal thrust inputs, but it only considered the rocket altitude and not a complete translational motion.

The Project Apollo ended in 1972, and, after that, research in the field of PDG came to a halt. Additionally, as NASA transitioned to the Space Shuttle program, the focus from the major space agencies changed from PDG to Powered Flight Guidance (PFG). Meanwhile, the work on 3-DOF rocket landing control problems became less extensive [17, 18].

2.2 A Renewed Interest

2.2.1 Polynomial Guidance Laws

At the start of the 21st century, with robotic planetary missions to the Moon and Mars on the horizon, research on methods for the PDG control problem gained a new enthusiasm. In 1997, Christopher D'Souza developed a 3-DOF landing polynomial guidance law which looks to, simultaneously, minimize the control efforts and the time to land [19]. It did not consider any constraints, but it provided an exact analytical solution to the two-point boundary-value problem. In 2007, the 3-DOF minimum fuel problem for the case of a pinpoint landing in Mars was first formulated [20]. First-order necessary conditions for minimum fuel solutions are derived, based on a Hamiltonian analysis, and interpreted. After that, other authors have studied analytical solutions to the first-order necessary conditions for the 3-DOF landing problem, along with efficient numerical methods [21–23].

However, the guidance laws discussed so far are based on polynomial analytical equations and, despite being solvable either through algebraic or numerical methods, do not consider constraints. For that reason, polynomial control laws can lead to trajectories which are physically infeasible, since there is a maximum thrust and a maximum speed that the rocket can achieve. Also, a vehicle whose navigation system is based on polynomial methods may require a higher propellant-to-mass ratio beforehand [24]. This happens because, if the spacecraft largely deviates from the landing site, it can eventually use more fuel that what it is available to realign itself.

2.2.2 Optimal Landing Control Problem

For that reason, in 2005, an alternative approach was proposed by B. Açikmese and S. Ploen [25]. For the first time, a standard optimization problem was formulated to find an optimal trajectory for the PDG landing phase and, then, the Lossless Convexification (LCvx) method was applied.

The reality is that the optimization problem to generate a landing trajectory is inherently nonconvex, due to nonconvex control constraints. Solving such optimization problem is possible through a Nonlinear Programming (NLP) method. However, that may not be desirable, as a NLP method does not have convergence guarantees. A promising PDG control algorithm must be able to, with the limited computing power onboard the spacecraft, quickly find a solution in real-time. For that reason, a reformulation is required. In [25], the nonconvex constraints are convexified and a similar finite-dimensional convex optimization problem is formulated. It is shown that the new optimization problem leads to the same solution as the original one, which is essentially the working principle behind LCvx.

In 2006, a comparison between the newly proposed LCvx algorithm, a polynomial algorithm based on the Apollo guidance law and another polynomial algorithm based on arbitrary order polynomials has been made [26]. Despite being more complex, the solution generated by the convex optimization problem was found to have a better performance in terms of maneuverability. A strong advantage of the LCvx approach is that it leads to a Second-Order Cone Programming (SOCP) optimization problem [27]. SOCP is a specific subset of convex optimization problems, which has known convergence guarantees. It can be solved through primal-dual interior-point method algorithms, whose high efficiency allows them to run in real-time and onboard the spacecraft.

The strategy behind LCvx has been applied to a series of nonconvexities associated with the PDG control problem, such as a lower bound on the thrust magnitude [25], input pointing constraints [28, 29] and minimum error-landing optimization problems [30]. This approach has also led to the development of an algorithm named Guidance for Fuel Optimal Large Diverts (G-FOLD), which was implemented in live flight tests with the help of NASA and the old Masten Space Systems [31, 32].

Although LCvx proved itself to be an useful tool, it may not be available in certain problem structures, such as nonlinear kinematics or dynamics. For that reason, a different methodology to handle nonconvex optimal control problems, with aerospace applications, was proposed [33]. It was referred to as Successive Convexification (SCvx) and, instead of simply rewriting a nonconvex formulation into a convex formulation through the use of slack variables, SCvx approximates sources of nonconvexity through an iterative solution process where a sequence of convex SOCP subproblems are solved. This is possible due to linearization, use of trust regions and relaxations. The main advantage here is that SCvx can handle a much broader class of nonconvex optimization control problems, but it does so by introducing more computational demand and weakening the convergence guarantees [33]. Later, the SCvx principle was also used to include aerodynamic drag, mass-depletion dynamics and free-final-time constraints in the 3-DOF PDG control problem [34].

Lars Blackmore, the principal rocket landing engineer at SpaceX, has explained that the company uses these convex optimization strategies, especially Lossless Convexification, to compute trajectories from the vehicle's current location to the landing site [1]. These calculations are performed by the

onboard computer and have an emphasis on computationally efficiency, as failure to find a feasible solution on time could lead to a crash.

2.3 6-DOF Landing Algorithms

However, all the methods discussed so far have a significant limitation. Similarly to the Apollo guidance law, they assume that the spacecraft orientation is controlled by a faster inner control loop. For that reason, the translational and rotational motions of the spacecraft are treated as two decoupled motions. But, in a rigid body, designing trajectories in a 3-DOF sense may lead to infeasible results when applied to real scenarios, where the systems are 6-DOF. On top of that, to accomplish today's goals of collision avoidance and target replanning, there is a need to employ vision-based sensors in order to analyse the landing sites and guarantee that there are no terrain hazards [35]. These sensors lead to line-of-sight constraints, which inevitably couple the position and orientation of the space vehicle. Hence, recent research work on PDG guidance and control laws focuses on the more generalized 6-DOF system.

One of the first scientific articles to develop a control strategy for the 6-DOF PDG problem was published in 2012 [36]. The authors described the translational and rotational dynamics of a rigid body in terms of dual quaternion vectors and used it to derive, through a Lyapunov analysis, stabilizing control laws. A few years later, the same authors developed a Model Predictive Control (MPC) algorithm to tackle the precision landing problem where line-of-sight constraints are imposed [37]. A piecewise-affine model was used to describe the coupled translational and rotational dynamics of the spacecraft. A quadratic cost function on the control actions was imposed to minimize the fuel consumption, and the different constraints are formulated in a convex setting. In the end, the algorithm was able to land the spacecraft accurately and respect the line-of-sight constraints, but, unless when the prediction horizon was relatively short, the computation time proved to be too large to be implementable in real-time onboard the spacecraft.

M. Szmuk and B. Açikmese were responsible for working over the years on the application of the SCvx method to the 6-DOF PDG control problem. The first article published on the matter was an extension of the previous work done on the SCvx topic to now include the coupling between translational and rotational motions [38]. Later, this approach was further developed to minimize not only the fuel consumption but also the time of flight [39]. The authors showed again that Successive Convexification is able to eliminate nonconvexities of a complex nonlinear and nonconxex optimization problem such as the 6-DOF PDG, and the corresponding solution is capable of satisfying the original dynamics. They also demonstrated that, for the typical iterative procedure of SCvx, the initial guess for the reference trajectory has a small influence on the solution.

Later, state-triggered constraints were introduced into the formulation of the 6-DOF PDG optimization problem [40]. State-triggered constraints are constraints that are only enforced if the state vector meets a certain criteria. In the case above, a state-triggered constraint is introduced to express the need to maintain a clear line of sight to the landing site, but only when the spacecraft altitude is within a certain interval. The authors were able to incorporate such constraints in the optimization problem

without resorting to mixed-integer programming approaches. So, for that reason, after a Successive Convexification procedure, the solution is obtained through a convex optimization programming method and allows the algorithm to be real-time implementable. This approach was then extended to handle compound state-triggered constraints, where the trigger and constraint conditions are vector-valued, instead of scalar-valued, which are then compounded through Boolean logic operators [41]. For example, in the case of a Or-Trigger and an And-Constraint functions, all the elements of the constraint condition are enforced when at least one element of the trigger condition is satisfied.

A complete summary of the SCvx algorithm for the 6-DOF PDG control problem, with line-of-sight constraints and whose dynamics is based on dual quaternion vectors, is presented on [42]. This paper also highlights its real-time implementability as it was shown that the algorithm would only take, on average, few deciseconds to find a solution. A final extension on this subject was then published, which included a free-ignition time modification that allows the algorithm to determine the optimal engine ignition time and a tractable aerodynamics formulation that models both lift and drag [3]. Two subsequent articles focused on developing real-time-capable customized solvers for efficient onboard implementation [43, 44].

In fact, the work developed through [44] was a first step towards computational efficiency and real-time implementability of the complete dual quaternion-based SCvx PDG algorithm, justified by the fact that it was selected as a candidate for NASA's Safe and Precise Landing - Integrated Capabilities Evolution (SPLICE) project. It was later flight-tested on the Blue Origin's New Shepard suborbital rocket onboard the Descent and Landing Computer [45, 46]. This was the first time that the Successive Convexification approach was used in a real space mission, and represented a significant advancement in the field of computational guidance and control.

However, this implementation focused on demonstrating the capability of the dual quaternion-based SCvx PDG algorithm in a terrestrial flight configuration. Space missions to celestial bodies usually require stricter missions specifications to guarantee a safe landing. For that reason, in 2022, improvements were made so that it could meet the requirements for a lunar landing scenario [47]. The constraints were actually designed specifically for the Blue Origin's Blue Moon lander.

On the other hand, as an ESA-sponsored activity, a different algorithm designed to tackle specifically the control challenges of reusable rockets recovery procedures was presented [48]. The authors start by developing a complete flight mechanics model of a reusable launch vehicle and, from there, study the coupling between guidance and control. A benchmark is developed, which creates a starting point for the improvement of current landing guidance and control algorithms. This work was the basis for the proposal of the DESCENDO algorithm, which stands for descending over extended envelopes using Successive Convexification-based optimization [49]. It follows a similar approach to [34], but is specifically designed to guide and control a reusable launch vehicle during all the distinct scenarios of its flight procedure, from takeoff to recovery. Also, while most methods presented until now are feed-forward trajectory generation algorithms, which run on an offline setting, DESCENDO is a closed-loop algorithm which looks to balance between computational efficiency and trajectory optimality. One of the limitations of DESCENDO is that assumes that the attitude control is done through a separate method,

so it is actually a 3-DOF method.

In the meantime, the German Aerospace Center DLR, the French National Center for Space Studies CNES and the Japan Aerospace Exploration Agency JAXA have decided to participate in a joint project to develop and fly a scaled reusable rocket stage called CALLISTO [50]. Such project is intended to develop and demonstrate, especially among the European aerospace industry, the technologies and landing capabilities necessary for the operation of future reusable launch vehicles. The cited article supports the idea that convex optimization is key to compute a landing trajectory and the corresponding feedforward control actions. Additionally, it emphasizes the importance of combining convex optimization strategies with structured robust control methods.

2.4 Funnel Synthesis Method

The reality is that a complete characterization of fuel optimal solutions for the 6-DOF PDG problem is still an active area of research. Many other articles have been published on the matter. However, a large percentage of this scientific work focuses on explicit trajectory generation methods [51–54]. The main goal of explicit methods is to compute a reference trajectory that connects given initial and final states and is optimal relative to a predefined metric. Even though there are successful and real-time implementable strategies that fall into this category [42, 55], explicit methods have two big limitations.

The major one is related with the fact that they are specific to a certain problem with given parameters. That means that, if the problem data changes after a solution to the optimization problem has been found (the best example would be if the vehicle's true initial condition is different from the assumed one) and one insists in following the same reference trajectory, it is not guaranteed that the dynamics and the constraints are satisfied. To ensure feasibility, one would need to solve again the optimal control problem and compute a new reference trajectory in real-time. This drawback is more noticeable when the uncertainties and disturbances that act on the system in real-life scenarios are taken into account, which, therefore, implies that it is almost guaranteed that some parameters will be different from the ones used in the initial optimization problem.

Secondly, there are no theoretical guarantees for the convergence of nonconvex optimal control problems. No known algorithm can be formally guaranteed to solve a nonconvex optimal control problem from an arbitrary initial guess [8]. All the convergence conclusions drawn on the scientific papers listed above are made based on experimental results. For that reason, it is only natural that other trajectory planning methods start to gain more attention.

On the other hand, implicit trajectory generation techniques do not compute a single trajectory, but rather a group of functions that define a set of feasible trajectories that connect the sets of initial and terminal boundary conditions. Without solving again the optimal control problem, these methods are able to find a new trajectory from the previously computed reference and, simultaneously, guarantee that it is feasible both with respect to dynamics and constraints.

This principle is closely related to neighboring optimal control, which was already introduced among the trajectory optimization community a while ago [56, 57]. The main idea is to use a first-order model

of a nonlinear system and a second-order model of the cost function around some trajectory, and study the necessary conditions for optimality given by the maximum principle. A state feedback law that is obtained by a solution of these necessary conditions can provide a near-optimal controller, rather robust to parameter variations. Across the literature, one can see the application of the neighboring optimal control theory in many cases, such as vehicle launch [58], orbit transfer [59], guidance in wind fields [60] and robot trajectory tracking [61].

Funnel synthesis is a method which can be used for trajectory planning and which precisely lies in the implicit trajectory generation techniques' category. It differs from the neighboring optimal control strategy described above because, instead of using the Pontryagin maximum principle, it resorts to a Lyapunov function strategy to compute a control invariant set and a corresponding feedback controller. The following thesis project is primarily based on this method and, therefore, a detailed introduction is required.

Research on the funnel synthesis topic was primarily motivated by the robotics community. In 1985, M. Mason introduced the problem of uncertainty in robot manipulators and named a mechanical operation which reduces uncertainty as a "funnel" [62]. This metaphor was used since, as a typical funnel allows to eliminate the uncertainty regarding the end location of something which is poured through its opening, the author's goal was for the mechanical manipulator to prioritize operations which accomplish a given goal even when object shapes and initial positions are uncertain. In 1999, the mathematical concept of a funnel was developed, making, for the first time, the association between a Lyapunov function and a funnel [63].

Figure 2.1 shows an ideal conceptual plot of a funnel centered at a goal point, represented by the dashed line, which then constitutes its only zero and unique minimum. The surface represents the $x - y$ plane where trajectories evolve, while the 3D space is a visual representation of the Lyapunov function value for each point on the plane. The goal is to find a control strategy that guarantees that the system trajectories along the $x - y$ plane lead strictly downward when projected up onto the funnel. This is an important result as it implies that all initial conditions on the plane will be drawn towards the center of the funnel.

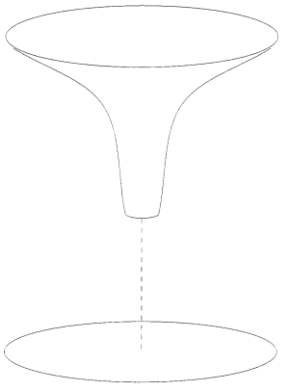

Figure 2.1: An ideal representation of a Lyapunov function as a funnel, [63]

The definition of a funnel is usually broader, in order to include not only asymptotically stable dy-

namical systems but also attracting dynamical systems. Asymptotically stable dynamical systems have a single equilibrium point, while attracting dynamical systems may have a broader range of attractors. In that sense, the funnel ends up representing an invariant region, such as the level sets of a Lyapunov function. The details on the mathematical formulations behind these theoretical concepts are provided on a further section. Figure 2.2 is a visual representation of a broader concept of a funnel, where the domain may represent, for example, the obstacle-free state-space in a robot path planning scenario.

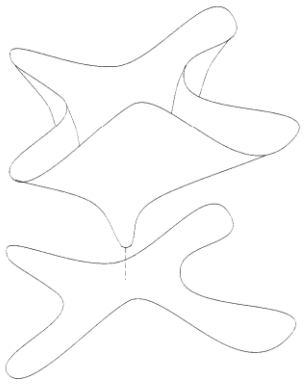

Figure 2.2: A visual representation of a broader concept of a funnel, [63]

However, it can be very difficult or even impossible to find globally attractive control laws, so Figure 2.2 is not implementable in practice. Instead, in some cases, especially in motion planning tasks, the chosen approach is the one precisely proposed in [63]: sequential composition of local funnels. Figure 2.3 is a visual representation of this strategy. The goal here is to derive controllers whose corresponding regions of attraction and induced funnels cover a large part of the state-space. With that, the state-space can be divided into cells where different controllers are active. As a controller drives the system toward its local minimum, the state will eventually cross a boundary into another region of the state-space where another controller is active. This process is repeated until the state reaches the attracting region of the last funnel, which contains the goal state.

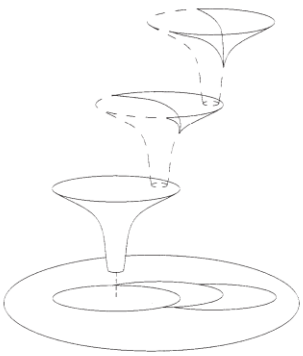

Figure 2.3: A visual representation of a funnel sequential composition, [63]

The ideas behind funnel synthesis and funnel sequential composition were later further developed by other authors, especially in the last decade [7, 64–66]. In order to compute the Lyapunov functions which

define the funnels, the methods in the aforementioned papers use a convex optimization tool, known as Sum-of-Squares (SOS) programming method. For systems whose dynamics can be described through polynomials, a polynomial Lyapunov function can be computed and its positive-definiteness can be checked through a SOS optimization problem.

A different approach was proposed in [8], where the authors apply the funnel synthesis method precisely to the 6-DOF PDG control problem. Here, the goal behind computing a funnel and the corresponding controller is not to plan a collision-free trajectory, but guaranteeing that the algorithm is capable of generating feasible trajectories for a wide set of possible initial conditions. For that reason, they look to maximize a control invariant set around a nonlinear function, which corresponds to the system dynamics.

Also, instead of considering a SOS method, the authors used a first-order approximation of the dynamics and derived a Differential Matrix Inequality (DMI), similar to the differential Riccati equations that often appear in the robust control literature, to guarantee quadratic stability of the system and, consequently, funnel invariance. That gives rise to a new subclass of funnels, referred to as quadratic funnels.

2.5 Robust Methods

As deterministic PDG guidance and control algorithms advance, a new interest has arisen among the research community: making the trajectory planning problem more robust to a multitude of uncertain factors [6].

One common approach is to design a feedback controller to correct possible deviations between the actual trajectory and the reference. In that sense, the control input applied to the system is determined by:

$$u(t) = \bar{u}(t) + K(t)(x(t) - \bar{x}(t)) \tag{2.1}$$

where $\bar{x}(t)$ and $\bar{u}(t)$ are the reference state and control inputs, while $K \in \mathbb{R}^{n_u \times n_x}$ is a feedback gain matrix.

As already mentioned, the funnel synthesis method simultaneously finds a control invariant set and associates it with a stabilizing attractive control law, defined by a time-varying feedback controller. Given the formulation of the funnel synthesis problem, this feedback controller guarantees that, even when the actual trajectory deviates from the reference, the system can follow a feasible path and, in the end, converge to the goal point.

Quadratic funnel synthesis has interesting similarities with robust control theory, such as $\mathcal{H}_\infty$ control and Robust Model Predictive Control (RMPC), especially when it comes to the derivation of matrix inequalities [67–69]. Additionally, there is a specific class of RMPC methods, called tube-based MPC, which also relies on the principle of invariant sets inside feasible regions [70].

Inside the scope of the 6-DOF PDG control problem, other strategies to compute a stabilizing feedback controller have been proposed in the past: chance-constrained dynamic programming [71]; covari-

ance steering [72, 73]; and Chebyshev interval method [74].

2.6 Research Relevance

Computing a reference trajectory from an initial condition to the landing site, in a fast, reliable and efficient manner, is a research topic that has already been largely studied and understood. When it comes to tackling this issue, the SCvx is the current state-of-the-art algorithm, having already proved its usefulness and gained popularity among the research community. Additionally, the efforts to develop real-time-capable convex solvers have paved the way for the SCvx to be implementable in real-world systems.

Therefore, the goal is to focus on a different aspect of the 6-DOF PDG problem: how to increase the robustness of the current landing control methods when faced with uncertainty and external disturbances. This is the next step that needs to be taken towards the development and improvement of landing algorithms because, as seen above, the scientific work on the topic of robust PDG methods is not very extensive.

Funnel synthesis is a robust control method that showed strong signs of being real-world implementable due to its increased computational efficiency. It is able to compute feasible trajectories, which can deviate from the reference trajectory, and connect a large set of initial conditions to the goal position. However, the work done towards applying funnel synthesis to the 6-DOF PDG control problem is still very introductory, with some limitations that could be further mitigated. In [8], the first steps within this topic were taken but there are still improvements that can be proposed. First of all, one should assess the performance of the funnel synthesis method when applied to a dynamical system that actually has external disturbances directly acting on it, something that the above article has not considered. Then, the impact of different problem formulations should be studied, such as how using a priori a model of the disturbances to derive the funnel invariance conditions can affect the performance of the feedback controller, and what are the advantages of simultaneously maximizing the funnel entry and minimizing the remaining funnel size. These are questions to be answered at the end of this thesis.

Nevertheless, the novel element to be introduced with this thesis project is how one can use real-time information regarding trajectory evolution to readjust the controller gains and decrease the effects of disturbances. Two different approaches are proposed. The first one is based on a MPC algorithm, where the parameters from a funnel computed offline are leveraged to formulate a cost function and a terminal constraint. The advantage of this strategy is outlined by comparing its performance with a standard MPC controller. The second approach considers real-time replanning, where, in a fast and computationally efficient way, the reference trajectory and the funnel are recomputed from time to time. Some theoretical remarks regarding the recursive feasibility and the stability of the latter method are also provided. Both these approaches have not been explored throughout the literature yet.

Chapter 3

Methods

In the third chapter, the mathematical formulations behind the different algorithms, as well as the dynamical model used to simulate the landing sequence, are presented. Not only the spacecraft's equations of motion but also the imposed state and input constraints which enable a safe and realistic system evolution are explained. The used reference trajectory generation algorithm, referred to as the Successive Convexification (SCvx) algorithm, is briefly reviewed. The most important parts of this thesis project soon follow, with the funnel synthesis method and the corresponding variations on its formulation. The chapter finishes with an explanation of the newly proposed strategies.

3.1 Prior Information

Before presenting the dynamical model used to simulate the landing procedure, the reader should have some prior information. First of all, it is important to know that some assumptions are made beforehand. It is assumed that the Powered Descent Guidance (PDG) maneuver is carried close enough to the terrain and for a short duration such that the gravity can be considered constant. Additionally, it is presumed that all maneuvers occur at speeds significantly lower than orbital speeds, thereby allowing us to disregard the influence of planetary rotation. Finally, the ambient pressure, the inertia matrix, the center of mass and the center of pressure for aerodynamic forces are all considered to be constant.

Also, for the dynamical model explained throughout the next Section, two different reference frames are considered. The first one is the frame fixed on the landing site, denoted by $\mathcal{F}_{\mathcal{I}}$ and described by the orthonormal vectors $\{\mathbf{x}_{\mathcal{I}}, \mathbf{y}_{\mathcal{I}}, \mathbf{z}_{\mathcal{I}}\}$. It is centered precisely on the desired landing location. $\mathbf{z}_{\mathcal{I}}$ points towards the sky, while $\mathbf{x}_{\mathcal{I}}$ and $\mathbf{y}_{\mathcal{I}}$ complete the right-handed orthogonal reference frame. On the other hand, a body-fixed frame $\mathcal{F}_{\mathcal{B}}$ is defined, which is especially useful for attitude control purposes. The orthonormal vectors are represented by $\{\mathbf{x}_{\mathcal{B}}, \mathbf{y}_{\mathcal{B}}, \mathbf{z}_{\mathcal{B}}\}$, where $\mathbf{z}_{\mathcal{B}}$ points along the vehicle's vertical axis.

Finally, in order to describe the orientation and rotational motion of the spacecraft, an Euler angle parameterization, with a 3-2-1 sequence, is used. It is important to outline that one cannot use quaternions or dual quaternions for this purpose, because, as seen later throughout the Section relative to the funnel synthesis method, differences between two distinct orientations should be described by an

additive factor and not a multiplicative factor.

The Euler angle vector $\boldsymbol{\Theta} \in \mathbb{R}^3$ is then defined as:

$$\boldsymbol{\Theta} = \begin{bmatrix} \varphi \\ \theta \\ \psi \end{bmatrix} \tag{3.1}$$

where φ is the roll angle, θ is the pitch angle and ψ is the yaw angle.

Transforming an Euler angle parameterization $\boldsymbol{\Theta}$ into a Direction Cosine Matrix (DCM), which can then be used for transformations between different references frames, is possible due to the following expression:

$$C(\boldsymbol{\Theta}) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\varphi) & \sin(\varphi) \\ 0 & -\sin(\varphi) & \cos(\varphi) \end{bmatrix} \begin{bmatrix} \cos(\theta) & 0 & -\sin(\theta) \\ 0 & 1 & 0 \\ \sin(\theta) & 0 & \cos(\theta) \end{bmatrix} \begin{bmatrix} \cos(\psi) & \sin(\psi) & 0 \\ -\sin(\psi) & \cos(\psi) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.2}$$

3.2 Spacecraft Dynamical Model

The equations of motion of the spacecraft's dynamical model are expressed in Cartesian variables, and, therefore, the complete state vector is given by:

$$\mathbf{x}(t) = \begin{bmatrix} m(t) \\ \mathbf{r}_{\mathcal{I}}(t) \\ \mathbf{v}_{\mathcal{I}}(t) \\ \boldsymbol{\Theta}(t) \\ \omega_{\mathcal{B}}(t) \end{bmatrix} \in \mathbb{R}^{13} \tag{3.3}$$

where $m(t) > 0$ is the vehicle's mass, $\mathbf{r}_{\mathcal{I}}(t) \in \mathbb{R}^3$ is the position vector relative to the frame $\mathcal{F}_{\mathcal{I}}$, $\mathbf{v}_{\mathcal{I}}(t) \in \mathbb{R}^3$ is the velocity vector relative to the same frame, $\boldsymbol{\Theta}(t) \in \mathbb{R}^3$ is the aforementioned Euler angle parameterization and $\omega_{\mathcal{B}}(t) \in \mathbb{R}^3$ is the angular velocity expressed in the body frame $\mathcal{F}_{\mathcal{B}}$.

The mass-depletion dynamics is a function of the thrust magnitude and, considering atmospheric effects acting on the rocket, its evolution can be described by:

$$\dot{m}(t) = -\alpha ||\mathbf{F}_{\mathcal{B}}(t)||_2 - \beta \tag{3.4}$$

where the constants α and β are given as:

$$\alpha := \frac{1}{I_{\text{sp}} g} \qquad\qquad \beta := \alpha P A_{\text{noz}} \tag{3.5}$$

In this case, $\mathbf{F}_{\mathcal{B}}(t) \in \mathbb{R}^3$ is the thrust force vector expressed in the body frame, I_{sp} is the vacuum specific impulse, g is the gravity acceleration constant, P is the atmospheric pressure and A_{noz} is the exit area of the engine's nozzle. The constant β represents the specific impulse reduction incurred by atmospheric back-pressure [75].

The translational equations of motion can be summarized as:

$$\begin{aligned}\dot{\mathbf{r}}_{\mathcal{I}}(t) &= \mathbf{v}_{\mathcal{I}}(t) \\ \dot{\mathbf{v}}_{\mathcal{I}}(t) &= \frac{1}{m(t)}\mathbf{F}_{\text{net},\mathcal{I}}(t) + \mathbf{g}_I = \frac{1}{m(t)}\Big(C_{\mathcal{I}\leftarrow\mathcal{B}}(t)\Big(\mathbf{F}_{\mathcal{B}}(t) + \mathbf{A}_{\mathcal{B}}(t)\Big)\Big) + \mathbf{g}_{\mathcal{I}}\end{aligned} \tag{3.6}$$

where $\mathbf{g}_{\mathcal{I}} \in \mathbb{R}^3$ is the gravity acceleration vector, $\mathbf{F}_{\text{net},\mathcal{I}}(t)$ is the net force acting on the spacecraft, which includes both propulsion and aerodynamic terms. The propulsion term is given by $C_{\mathcal{I}\leftarrow\mathcal{B}}(t)\mathbf{F}_{\mathcal{B}}(t)$, with $C_{\mathcal{I}\leftarrow\mathcal{B}}(t)$ being the DCM obtained through (3.2).

To estimate the aerodynamic forces, it is assumed that the velocity is subsonic throughout the entire trajectory and that the wind does not have a direct effect on the drag magnitude. Then, the spacecraft is modeled as a three-dimensional ellipsoid with a reference area A and a parameter C_A, which is a 3×3 matrix that captures how the aerodynamic forces interact with the vehicle in different directions. Also, since most rocket-powered vehicles can be approximated as axisymmetric, C_A usually has the following structure: $C_A = diag([c_{a,x}, c_{a,yz}, c_{a,yz}])$. The aerodynamic force $\mathbf{A}_{\mathcal{B}}(t)$ is then computed by [3]:

$$\mathbf{A}_{\mathcal{B}}(t) = -\frac{1}{2}\rho A||\mathbf{v}_{\mathcal{I}}(t)||_2 C_A C_{\mathcal{B}\leftarrow\mathcal{I}}(t)\mathbf{v}_{\mathcal{I}}(t) \tag{3.7}$$

with ρ being the atmospheric density.

The rotational equations of motion are given by:

$$\begin{aligned}\dot{\mathbf{\Theta}}(t) = \Omega(\mathbf{\Theta}(t))\omega_{\mathcal{B}}(t) &= \begin{bmatrix} 1 & \sin(\varphi(t))\tan(\theta(t)) & \cos(\varphi(t))\tan(\theta(t)) \\ 0 & \cos(\varphi(t)) & -\sin(\varphi(t)) \\ 0 & \sin(\varphi(t))\frac{1}{\cos(\theta(t))} & \cos(\varphi(t))\frac{1}{\cos(\theta(t))} \end{bmatrix} \begin{bmatrix} \omega_1(t) \\ \omega_2(t) \\ \omega_3(t) \end{bmatrix} \\ J_{\mathcal{B}}\dot{\omega}_{\mathcal{B}}(t) = \mathbf{M}_{\mathcal{B}}(t) - \omega_{\mathcal{B}}(t) \times J_{\mathcal{B}}\omega_{\mathcal{B}}(t) &= \tau_{\mathcal{B}}(t) + \mathbf{r}_{T,\mathcal{B}} \times \mathbf{F}_{\mathcal{B}}(t) + \mathbf{r}_{cp,\mathcal{B}} \times \mathbf{A}_{\mathcal{B}}(t) - \omega_{\mathcal{B}}(t) \times J_{\mathcal{B}}\omega_{\mathcal{B}}(t)\end{aligned} \tag{3.8}$$

where $J_{\mathcal{B}} \in \mathbb{S}^3_{++}$ is the body-fixed constant inertia tensor of the vehicle, while $\mathbf{r}_{T,\mathcal{B}}$ and $\mathbf{r}_{cp,\mathcal{B}}$ are constant body-frame vectors that go from the vehicle's center of mass to the gimbaled engine pivot point and aerodynamic center of pressure, respectively. Also, $\tau_{\mathcal{B}}(t)$ is an additional torque applied to the spacecraft by a Reaction Control System (RCS). A RCS is an actuator system consisting of multiple smaller thrusters which help provide additional attitude control, rotating the vehicle in any combination of directions.

Therefore, in a vehicle with a gimbaled rocket engine and a RCS, where a thrust force and a torque vectors are the corresponding control actions, the complete state-space system description is given by:

$$\mathbf{x}(t) = \begin{bmatrix} m(t) \\ \mathbf{r}_{\mathcal{I}}(t) \\ \mathbf{v}_{\mathcal{I}}(t) \\ \boldsymbol{\Theta}(t) \\ \omega_{\mathcal{B}}(t) \end{bmatrix} \in \mathbb{R}^{13} \qquad \mathbf{u}(t) = \begin{bmatrix} \mathbf{F}_{\mathcal{B}}(t) \\ \tau_{\mathcal{B}}(t) \end{bmatrix} \in \mathbb{R}^{6}$$

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} -\alpha||\mathbf{F}_{\mathcal{B}}(t)||_2 - \beta \\ \mathbf{v}_{\mathcal{I}}(t) \\ \frac{1}{m(t)}\Big(C_{\mathcal{I}\leftarrow\mathcal{B}}(t)\Big(\mathbf{F}_{\mathcal{B}}(t) + \mathbf{A}_{\mathcal{B}}(t)\Big)\Big) + \mathbf{g}_{\mathcal{I}} \\ \Omega(\boldsymbol{\Theta}_{\mathcal{B}}(t))\omega_{\mathcal{B}}(t) \\ J_{\mathcal{B}}^{-1}(\tau_{\mathcal{B}}(t) + \mathbf{r}_{T,\mathcal{B}} \times \mathbf{F}_{\mathcal{B}}(t) + \mathbf{r}_{cp,\mathcal{B}} \times \mathbf{A}_{\mathcal{B}}(t) - \omega_{\mathcal{B}}(t) \times J_{\mathcal{B}}\omega_{\mathcal{B}}(t)) \end{bmatrix} \tag{3.9}$$

3.3 State and Input Constraints

Throughout the following Section, the state and input constraints which should be imposed in the different optimization problems are explained. These constraints guarantee that the generated reference trajectory and corresponding control actions lead to a safe, realistic and feasible landing.

The first state constraint guarantees that the generated trajectory does not force the vehicle to use more fuel than what is actually available. Therefore, a minimum bound on the vehicle's mass m is imposed as following:

$$m \geq m_{\text{dry}} \tag{3.10}$$

where $m_{\text{dry}} \in \mathbb{R}_{++}$ is the mass of the vehicle when it has no fuel.

Next, constraints on two angles that characterize the vehicle's descent trajectory are defined: the approach angle γ and the tilt angle θ. The approach angle γ is the angle between $\mathbf{r}_{\mathcal{I}}$, the vector that goes from the landing site to the vehicle's center of mass, and $\mathbf{z}_{\mathcal{I}}$, one of the orthonormal vectors of the reference frame $\mathcal{F}_{\mathcal{I}}$. It is used on the optimization problem to guarantee that the complete trajectory is located above the surface, and, for positions further away from the landing site, that the vehicle is high enough to reach the target position safely. On the other hand, the tilt angle θ is defined as the angle between the vehicle's vertical direction $\mathbf{z}_{\mathcal{B}}$ and $\mathbf{z}_{\mathcal{I}}$. A visual representation of the cones that are defined by imposing limits on the approach and tilt angles is seen in Figure 3.1.

The approach angle constraint is given by:

$$-\mathbf{r}_{\mathcal{I}}^T\mathbf{z}_{\mathcal{I}} + ||\mathbf{r}_{\mathcal{I}}||_2 \cos\gamma_{\text{max}} \leq 0 \tag{3.11}$$

where γ_{max} is the maximum allowable approach angle.

On the other hand, the tilt angle constraint is written as:

$$\cos\theta_{\text{max}} - \mathbf{z}_{\mathcal{I}}^T\mathbf{z}_{\mathcal{B}} \leq 0 \tag{3.12}$$

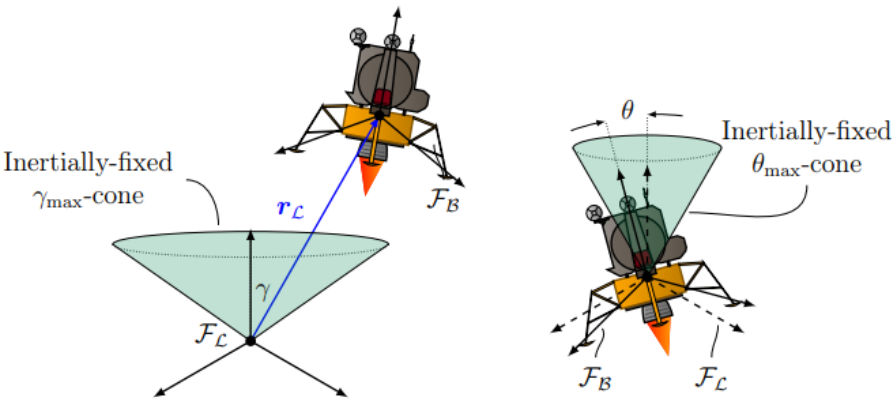

Figure 3.1: Representation of the cones defined by $\gamma_{\max}$ (left) and $\theta_{\max}$ (right), [76]

where $\theta_{\max}$ is the maximum allowable tilt angle.

Finally, the last state constraints should be imposed to bound linear and angular velocities. With $v_{\max},\ \omega_{\max} \in \mathbb{R}_{++}$ being the maximum allowable velocities, these constraints can be written as:

$$||\mathbf{v}_{\mathcal{B}}||_2 \leq v_{\max} \qquad\qquad ||\omega_{\mathcal{B}}||_\infty \leq \omega_{\max} \tag{3.13}$$

Proceeding with input constraints, firstly, one should impose limits on the thrust magnitude produced by the engine. Engines, in real-life scenarios, only operate within a certain thrust interval. For example, the main engines of the Apollo modules could either operate at 93% thrust or in the allowed interval of 11% to 65% of the rated thrust value [11]. For that reason, constraint (3.14) needs to be added to the optimization problem:

$$F_{\min} \leq ||\mathbf{F}_{\mathcal{B}}||_2 \leq F_{\max} \tag{3.14}$$

where $F_{\min},\ F_{\max} \in \mathbb{R}_{++}$ are the minimum and maximum permitted values for the engine thrust magnitude. Due to the lower bound enforced through $F_{\min}$, this constraint is nonconvex. However, through linearization around a reference trajectory, it is possible to reformulate it to be convex. In that case, (3.14) becomes:

$$F_{\min} - \frac{\bar{\mathbf{F}}_{\mathcal{B}}^T}{||\bar{\mathbf{F}}_{\mathcal{B}}||_2}\mathbf{F}_{\mathcal{B}} \leq 0 \tag{3.15}$$

where $\bar{\mathbf{F}}_{\mathcal{B}}$ is the thrust magnitude given by an initial guess or the solution of a previous iteration.

The engine is assumed to be gimbaled and able to rotate around two axes. The gimbal angle δ is the total angular deviation of the thrust vector from its nominal position. A maximum gimbal angle $\delta_{\max}$ is imposed, which then translates into a body-fixed cone that needs to contain the thrust vector. The gimbal angle constraint can be expressed in terms of Cartesian coordinates as:

$$||\mathbf{F}_{\mathcal{B}}||_2 \cos\delta_{\max} - \mathbf{z}_{\mathcal{B}}^T\mathbf{F}_{\mathcal{B}} \leq 0 \tag{3.16}$$

To guarantee that both the thrust magnitude and the direction of the thrust vector do not change too rapidly, constraints on the throttle rate and gimbal rate should be formulated. However, the equations that express precisely the temporal derivatives of the thrust magnitude and the gimbal angle are nonconvex [76]. Fortunately, there are convex reformulations that approximate these rates and are given by:

$$\begin{aligned} -\dot{F}_{z,max} &\leq \mathbf{z}_{\mathcal{B}}^{T}\dot{\mathbf{F}}_{\mathcal{B}} \leq \dot{F}_{z,\max} \\ ||E_{xy}\dot{\mathbf{F}}_{\mathcal{B}}||_2 &\leq \dot{\delta}_{\max}\mathbf{z}_{\mathcal{B}}\mathbf{F}_{\mathcal{B}} \end{aligned} \tag{3.17}$$

where $\dot{F}_{z,\max} \in \mathbb{R}_{++}$ is the maximum throttle rate, $\dot{\delta}_{\max}$ is the maximum gimbal rate and E_{xy} is a 2×3 matrix that selects the thrust vector components contained in the $\mathbf{x}_{\mathcal{B}}.\mathbf{y}_{\mathcal{B}}$ plane.

Finally, the torque values provided by the RCS system are constrained as:

$$||\tau||_\infty \leq \tau_{\max} \tag{3.18}$$

3.4 Reference Trajectory Generation Algorithm

To generate the reference trajectory, the Successive Convexification (SCvx) algorithm is used. The mathematical formulation behind it is briefly described in this Section since it was a part of the present thesis work. No toolboxes that apply this algorithm directly can be found, and, therefore, it had to be developed by hand from the beginning.

SCvx is an algorithm designed to solve the landing trajectory generation problem and which belongs to a broader class of solution methods to nonconvex optimization problems, commonly referred to as Sequential Convex Programming (SCP). It iteratively solves a convex relaxation of the original nonconvex problem and keeps updating the solution. The optimal solution is found with three goals in mind: exactly satisfying nonlinear dynamics; approximating the state and control constraints by enforcing them at a finite number of temporal nodes; and conservatively approaching optimality.

The SCvx iterative procedure is shown in Figure 3.2 and it is composed by three steps:

- Propagation step, which approximates the system nonlinear dynamics;
- Parameter update step, which is responsible for guaranteeing that the optimization problem is numerically well-conditioned;
- Solve step, which actually finds the optimal solution of the optimization subproblem.

The top half of the scheme in Figure 3.2 shows the analytical and mathematical operations that compose each of the three fundamental steps.

The SCvx algorithm will be explained based more on a practical rather than a theoretical approach. And, for that purpose, each of the steps mentioned above are applied to a general formulation of a nonconvex optimal control problem.

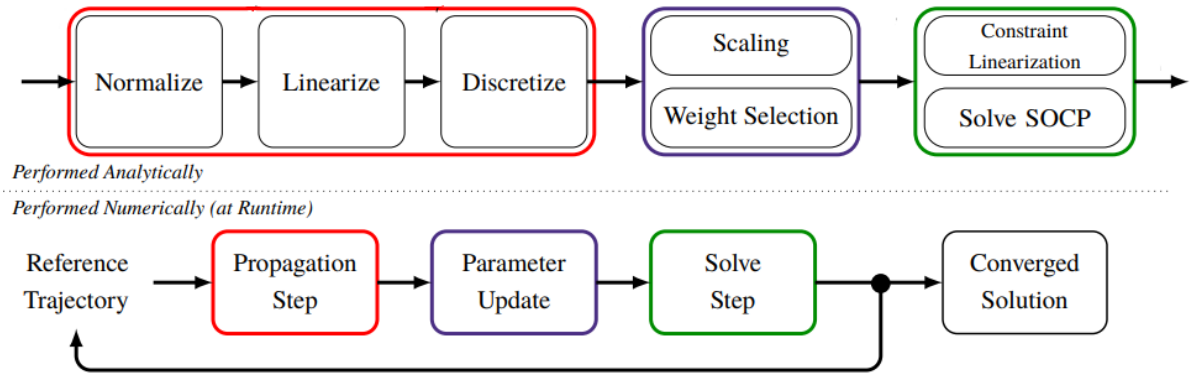

Figure 3.2: The sequence of steps that compose the SCvx algorithm, [42]

3.4.1 Propagation Step

The propagation step approximates the nonlinear dynamical equations of motion, which, in an optimization problem, are typically expressed as equality constraints. The goal is to convert these continuous-time nonlinear equations into discrete-time affine functions of the state, control and final time. A general nonlinear dynamical equation is considered, whose form is expressed as:

$$\dot{\mathbf{x}}(t) = f(\mathbf{x}(t), \mathbf{u}(t)), \ \forall t \in [t_0, t_f] \tag{3.19}$$

where $\mathbf{x} \in \mathbb{R}^{n_x}$ and $\mathbf{u} \in \mathbb{R}^{n_u}$.

The first operation of the propagation step is to temporally normalize such that the free-final-time problem becomes fixed-final-time. So, the interval $t \in [t_0, t_f]$ is converted to $\tau \in [0, 1]$, where τ is commonly referred as the normalized time. Applying the chain rule, the following can be derived:

$$\mathbf{x}'(t) := \frac{d}{d\tau}\mathbf{x}(t) = \frac{dt}{d\tau}\frac{d}{dt}\mathbf{x}(t) = \frac{dt}{d\tau}\dot{\mathbf{x}}(t) \tag{3.20}$$

The parameter $s := \frac{dt}{d\tau}$ is defined as the temporal dilation factor. After replacing t with τ and $\dot{\mathbf{x}}(t)$ with the right-hand side of (3.19), the temporal normalized dynamics are obtained:

$$\mathbf{x}'(\tau) = F(\mathbf{x}(\tau), \mathbf{u}(\tau), \tau) := sf(\mathbf{x}(\tau), \mathbf{u}(\tau)) \tag{3.21}$$

Subsequently, the linearization procedure leads to a fixed-final-time linear time-varying continuous-time problem, which is inherently convex. Using a reference trajectory $\bar{\mathbf{z}}(\tau) := \begin{bmatrix} \bar{s} & \bar{\mathbf{x}}(\tau) & \bar{\mathbf{u}}(\tau) \end{bmatrix}$, a first-order Taylor series expansion can be applied and the following linear time-varying dynamical equation is obtained:

$$\mathbf{x}'(t) \approx A(\tau)\mathbf{x}(\tau) + B(\tau)\mathbf{u}(\tau) + S(\tau)s + R(\tau), \ \ \forall \tau \in [0, 1] \tag{3.22}$$

where:

$$
\begin{aligned}
A(\tau) &:= \left.\frac{\partial F}{\partial x}\right|_{\bar{\mathbf{z}}(\tau)} & B(\tau) &:= \left.\frac{\partial F}{\partial u}\right|_{\bar{\mathbf{z}}(\tau)} \\
S(\tau) &:= \left.\frac{\partial F}{\partial s}\right|_{\bar{\mathbf{z}}(\tau)} & R(\tau) &:= -A(\tau)\bar{\mathbf{x}}(\tau) - B(\tau)\bar{\mathbf{u}}(\tau) - S(\tau)\bar{s}
\end{aligned}
\tag{3.23}
$$

Finally, the system is temporally discretized by projecting the infinite-dimensional control signal $\mathbf{u}(\tau)$ into a finite-dimensional space. For that purpose, the SCvx algorithm uses a First-Order-Hold (FOH) approach, as it leads to an increase in optimality and guarantees that the interpolated values satisfy constraints [3]. So, N temporal nodes are introduced, which result in $N-1$ time subintervals. Each temporal node is associated with an index $k \in \{1, 2, \ldots, N\}$, which corresponds to the normalized time $\tau_k = \frac{k-1}{N-1}$. The control signal is then decomposed in a continuous piecewise affine function as:

$$
\mathbf{u}(\tau) = \lambda_k^-(\tau)\mathbf{u}_k + \lambda_k^+(\tau)\mathbf{u}_{k+1},\ \forall \tau \in [0,1] \tag{3.24}
$$

where:

$$
\lambda_k^-(\tau) = \frac{\tau_{k+1} - \tau}{\tau_{k+1} - \tau_k} \qquad \lambda_k^+(\tau) = \frac{\tau - \tau_k}{\tau_{k+1} - \tau_k} \tag{3.25}
$$

Substituting (3.24) into (3.22):

$$
\mathbf{x}'(\tau) = A(\tau)\mathbf{x}(\tau) + \lambda_k^-(\tau)B(\tau)\mathbf{u}_k + \lambda_k^+(\tau)B(\tau)\mathbf{u}_{k+1} + S(\tau)s + R(\tau),\ \forall \tau \in [0,1] \tag{3.26}
$$

The state transition matrix $\phi(\tau, \tau_k) : [\tau_k, \tau_{k+1}] \to \mathbb{R}^{n_x \times n_x}$ associated with (3.26) is given by:

$$
\phi(\tau, \tau_k) = I_{n_x} + \int_{\tau_k}^{\tau} A(\xi)\phi(\xi, \tau_k)d\xi \tag{3.27}
$$

Denoting the discrete-time state vectors as $\mathbf{x}_k := \mathbf{x}(\tau_k)$ and using the inverse and transitive properties of ϕ, the final form of the discrete-time system dynamics is obtained:

$$
\mathbf{x}_{k+1} = A_k\mathbf{x}_k + B_k^-\mathbf{u}_k + B_k^+\mathbf{u}_{k+1} + S_k s + R_k \tag{3.28}
$$

where:

$$
\begin{aligned}
A_k &:= \phi(\tau_{k+1}, \tau_k) \\
B_k^- &:= A_k \int_{\tau_k}^{\tau_{k+1}} \phi^{-1}(\tau, \tau_k)\lambda_k^- B(\tau)d\tau & B_k^+ &:= A_k \int_{\tau_k}^{\tau_{k+1}} \phi^{-1}(\tau, \tau_k)\lambda_k^+ B(\tau)d\tau \\
S_k &:= A_k \int_{\tau_k}^{\tau_{k+1}} \phi^{-1}(\tau, \tau_k)s(\tau)d\tau & R_k &:= A_k \int_{\tau_k}^{\tau_{k+1}} \phi^{-1}(\tau, \tau_k)R(\tau)d\tau
\end{aligned}
\tag{3.29}
$$

In terms of practical implementation, the previous iteration of the SCvx procedure gives us $\bar{s}$, $\bar{\mathbf{x}}_k$ and $\bar{\mathbf{u}}_k$ for all $k \in \{1, 2, \ldots, N\}$. Using (3.24), one can obtain $\bar{\mathbf{u}}(\tau)$ for all $\tau \in [0, 1]$. (3.21), (3.27), (3.28) and (3.29) are computed simultaneously. For the integrals, a classical Runge-Kutta RK4 method is used [77]. The final results are obtained after the numerical integration, by right multiplying the final value of

A_k by the final value of each integral in the expressions of (3.29). Hence, one arrives to A_k, B_k^-, B_k^+, S_k and R_k.

3.4.2 Parameter Update Step

There is a high probability that the propagation step will result in a convex optimization subproblem that suffers from artificial infeasibility or artificial unboundness. For that reason, a few numerical procedures need to be implemented before solving it.

The first step is to eliminate artificial infeasibility. Artificial infeasibility can occur when linearization leads to inconsistencies in the constraints and they cannot be simultaneously satisfied. To avoid it, (3.28) is augmented with a virtual control term $\nu_k \in \mathbb{R}^{n_x}$:

$$\mathbf{x}_{k+1} = A_k \mathbf{x}_k + B_k^- \mathbf{u}_k + B_k^+ \mathbf{u}_{k+1} + S_k s + R_k + \nu_k \tag{3.30}$$

Additionally, to penalize the virtual control term such that it is only used when necessary for constraint satisfaction, the cost function is augmented with the term J_{vc}:

$$J_{vc}(\nu) := w_\nu \sum_{k=1}^{N-1} ||\nu_k||_1 \tag{3.31}$$

where $w_\nu > 0$ is a large user-defined weight. The addition of this virtual control term guarantees that each subproblem has a nonempty feasible set.

The next issue is artificial unboundness, which happens when constraints that have been previously linearized allow the cost function to be minimized indefinitely. To mitigate this effect and to, simultaneously, ensure that the optimization solver does not explore solutions too far away from the reference trajectory, the following quadratic trust region term is added to the cost function:

$$J_{tr}(\bar{\mathbf{z}}, \mathbf{z}) := \sum_{k=1}^{N} (\mathbf{z}_k - \bar{\mathbf{z}}_k)^T W_{tr} (\mathbf{z}_k - \bar{\mathbf{z}}_k) \tag{3.32}$$

where W_{tr} is a symmetric positive definite weighing matrix, defined by the user.

The parameter update step ends with a scaling procedure, which is important since numerical issues can arise when there is a large disparity between the orders of magnitude of the solution variables [78]. For that reason, the SCvx applies the following affine transformations:

$$\begin{aligned} \mathbf{x}_k &= P_x \hat{\mathbf{x}}_k + \mathbf{p}_x \\ \mathbf{u}_k &= P_u \hat{\mathbf{u}}_k + \mathbf{p}_u \\ s &= p_t \hat{s} \end{aligned} \tag{3.33}$$

where $\hat{\mathbf{x}}_k$, $\hat{\mathbf{u}}_k$ and $\hat{s}$ are the scaled terms. P_x, P_u, $\mathbf{p}_x$, $\mathbf{p}_u$ and p_s are chosen so that all scaled terms have a maximum order of magnitude of 1.

3.4.3 Solve Step

Finally, the solve step is reached, where the optimal solution of each convex subproblem is computed. However, there are still sources of nonconvexity inside the optimization subproblem which have not been addressed yet: the nonconvex constraints.

Consider the following general formulation of a nonconvex constraint, applied at a given temporal node $k \in \{1, 2, \ldots, N-1, N\}$:

$$h(\mathbf{z}_k) \leq 0 \tag{3.34}$$

where $h : \mathbb{R}^{n_z} \rightarrow \mathbb{R}$ is an inequality constraint function, at least once differentiable. Similarly to the nonlinear dynamics, a first-order Taylor series expansion around the reference trajectory is used to approximate the nonconvex constraint:

$$h(\bar{\mathbf{z}}_k) + \left.\frac{dh}{d\mathbf{z}}\right|_{\bar{\mathbf{z}}_k} (\mathbf{z}_k - \bar{\mathbf{z}}_k) \leq 0 \tag{3.35}$$

This procedure is especially useful to convexify the nonconvex constraint given by the minimum bound on the thrust magnitude.

When solving each subproblem, since the goal is to minimize the amount of fuel spent throughout the landing procedure, the cost function is actually formulated to maximize the value of the vehicle's mass at the last temporal node.

Now, one is ready to solve the convex relaxed optimization subproblem using a suitable customized interior-point method, designed to solve a Second-Order Cone Programming (SOCP) problem. At this stage, it is also important to emphasize that a straight-line initialization is considered. This means that, to start the algorithm, a straight trajectory between the initial and final points, where the nodes are uniformly distributed and equally spaced, is used as a guess for the first iteration.

Finally, a criteria which needs to be satisfied to terminate the iterative procedure should be defined. In the upcoming implementation, the deviation between two scaled state solutions of consecutive iterations is compared, and, if it is below a certain threshold, convergence is assumed to have been achieved and the iterative procedure ends. This can be mathematically formulated by stating that the iterations are terminated when the following condition is satisfied:

$$\max_{k \in \{1,2,\ldots,N-1,N\}} ||\hat{\mathbf{x}}_k - P_x^{-1}(\bar{\mathbf{x}}_k - p_x)||_2 < \delta x_{tol} \tag{3.36}$$

where $\delta x_{tol} > 0$ is an user-defined constant.

3.5 Funnel Synthesis Method

As discussed before, one common approach to correct the deviations from the reference trajectory is to associate a control invariant set and a stabilizing feedback controller to it. This invariant set is called a funnel and, if a system's initial condition is located inside its entry and no disturbances are considered,

it is mathematically guaranteed that the state and control input pairs remain inside the funnel for all future times. The method where a control invariant funnel and the corresponding feedback controller are optimized is called funnel synthesis.

Before introducing the funnel synthesis framework, one needs to derive a linearized system model from the original nonlinear model. This reformulation will serve as a basis to then compute the funnel around the reference trajectory.

Therefore, consider the following continuous-time nonlinear dynamical system:

$$\dot{x}(t) = f(t, x(t), u(t)) \ , \ t \in [t_0, t_f] \tag{3.37}$$

where $x(t) \in \mathbb{R}^{n_x}$ is the state vector, $u(t) \in \mathbb{R}^{n_u}$ is the control vector, t_0 and t_f are the initial and final times, respectively. The function $f : \mathbb{R}_+ \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_u} \to \mathbb{R}^{n_x}$ is assumed to be at least once differentiable. Note that, in this case, no direct disturbance or uncertainty effects are taken into account yet.

By considering a reference trajectory $\{\bar{x}(t), \bar{u}(t)\}_{t=t_0}^{t_f}$, the difference variables $\eta(t)$ and $\xi(t)$ can be defined as:

$$\eta(t) := x(t) - \bar{x}(t) \qquad\qquad \xi(t) := u(t) - \bar{u}(t) \tag{3.38}$$

Since f is differentiable, a new dynamical system in terms of the difference variables can be defined, by, simultaneously, using a first-order Taylor series expansion around the reference trajectory and resorting to the concept of structured nonlinearities [76]. The new dynamics are expressed as:

$$\begin{aligned} \dot{\eta}(t) &= A(t)\eta(t) + B(t)\xi(t) + \sum_{i=1}^{N_p} E_i p_i(t) \\ q_i(t) &= C_i\eta(t) + D_i\xi(t), \qquad\qquad i = 1, \dots, N_p \\ p_i(t) &= \phi_i(q_i(t)), \qquad\qquad i = 1, \dots, N_p \end{aligned} \tag{3.39}$$

$A(t)$ and $B(t)$ are the partial derivatives of the function f, with respect to the state and control input, respectively, evaluated along the reference trajectory $\{\bar{x}(t), \bar{u}(t)\}_{t=t_0}^{t_f}$. The last expression of (3.39) can be thought as a channel that groups all of the nonlinear effects of the system (3.37) in a single function, while N_p represents the number of nonlinear channels. $q_i \in \mathbb{R}^{n_{q,i}}$ is the input to a given nonlinear channel, $p_i \in \mathbb{R}^{n_{p,i}}$ is the output of the nonlinear channel and ϕ_i is the function that characterizes the nonlinear channel. The constant matrices $C_i \in \mathbb{R}^{n_{q,i} \times n_x}$, $D_i \in \mathbb{R}^{n_{q,i} \times n_u}$ and $E_i \in \mathbb{R}^{n_x \times n_{p,i}}$ serve as selectors of variables.

With the purpose of deriving the algorithm in a more understandable way, (3.39) is written in a compact format by stacking the effects of the nonlinear channels:

$$\begin{aligned} \dot{\eta}(t) &= A(t)\eta(t) + B(t)\xi(t) + Ep(t) \\ q(t) &= C\eta(t) + D\xi(t) \\ p(t) &= \phi(q(t)) \end{aligned} \tag{3.40}$$

where:

$$p = \begin{bmatrix} p_1 \\ \vdots \\ p_{N_p} \end{bmatrix} \quad q = \begin{bmatrix} q_1 \\ \vdots \\ q_{N_p} \end{bmatrix} \quad C = \begin{bmatrix} C_1 \\ \vdots \\ C_{N_p} \end{bmatrix} \quad D = \begin{bmatrix} D_1 \\ \vdots \\ D_{N_p} \end{bmatrix} \quad E = \begin{bmatrix} E_1 & \dots & E_{N_p} \end{bmatrix} \tag{3.41}$$

Note that the variables' dimensions change. Now, $p \in \mathbb{R}^{n_p}$, $q \in \mathbb{R}^{n_q}$, $C \in \mathbb{R}^{n_q \times n_x}$, $D \in \mathbb{R}^{n_q \times n_u}$ and $E \in \mathbb{R}^{n_x \times n_p}$, where $n_p = \sum_{i=1}^{N_p} n_{p.i}$ and $n_q = \sum_{i=1}^{N_p} n_{q.i}$.

Nevertheless, for the derivation of the funnel synthesis formulation, the focus is placed on the closed-loop system, where the control law is linear time-varying, given by $\xi(t) = K(t)\eta(t)$ for some $K \in \mathbb{R}^{n_u \times n_x}$. The final dynamical model is then written as:

$$\begin{aligned} \dot{\eta}(t) &= A_{cl}(t)\eta(t) + Ep(t) \\ q(t) &= C_{cl}(t)\eta(t) \\ p(t) &= \phi(q(t)) \end{aligned} \tag{3.42}$$

with $A_{cl}(t) := A(t) + B(t)K(t)$ and $C_{cl}(t) := C + DK(t)$.

3.5.1 Quadratic Funnels

The first step into the funnel mathematical formulation is understanding that a funnel, usually denoted by $\mathcal{F}(t)$, is a time-varying set in state and control space that is both control invariant and contained inside a feasible region. A control invariant set implies that, if a system's initial condition is located inside the entry of the funnel, it is always possible to find a control action that keeps the subsequent trajectory inside the given funnel for all future times. Formally, that can be formulated as $(x(t_0), u(t_0)) \in \mathcal{F}(t_0) \implies (x(t), u(t)) \in \mathcal{F}(t)$ for all $t \geq t_0$.

Quadratic funnels are a specific set of funnels which are obtained when quadratic stability is considered. The necessary and sufficient conditions to guarantee quadratic stability are expressed in [79] and [80]. These are based on quadratic Lyapunov functions as Lyapunov stability theory is the most generally valid approach when studying the stability of nonlinear systems. Therefore, a system is quadratically stable if, for a decay rate $\alpha > 0$ and a Lyapunov function $V(t, x(t)) = x(t)^T P x(t)$, with P being a symmetric positive-definite matrix, the following condition is met for all $t \geq 0$.:

$$\dot{V}(t, x(t)) \leq -\alpha V(t, x(t)) \tag{3.43}$$

For this specific case, the following Lyapunov function $V : \mathbb{R} \times \mathbb{R}^{n_x} \to \mathbb{R}$ is defined:

$$V(t, \eta(t)) = \eta(t)^T Q^{-1}(t)\eta(t) \tag{3.44}$$

where $Q(t) \in \mathbb{S}^{n_x}_{++}$ is a function dependent on time which generates positive-definite matrices as outputs.

The notion of level sets of a function will be used to parameterize the quadratic funnel, because, as long as condition (3.43) is met, the level sets of the Lyapunov function $V(t, \eta(t))$ are invariant. The 1-

level set of $V(t, \eta(t))$ is the set of states that satisfy the quadratic inequality $\eta^T Q^{-1} \eta \leq 1$. This inequality also parameterizes an ellipsoid defined in a n_x-dimensional space. The given ellipsoid is denoted as ε_Q and expressed as:

$$\varepsilon_Q = \left\{ \eta \in \mathbb{R}^{n_x} \mid \eta^T Q^{-1} \eta \leq 1 \right\} \tag{3.45}$$

Given the closed-loop control law $\xi = K\eta$, the following implication can also be stated [8]:

$$\eta \in \varepsilon_Q \implies \xi \in \varepsilon_{KQK^T} \tag{3.46}$$

To conclude the definition of quadratic funnels, one should assume that $\mathcal{X} \subset \mathbb{R}^{n_x}$ and $\mathcal{U} \subset \mathbb{R}^{n_u}$ are the feasible sets for the state and control vectors, respectively. Therefore, a quadratic funnel $\mathcal{F}$ can be formally defined as a control invariant and feasible set, within the state and control space, that is parameterized by a time-varying positive definite matrix $Q \in \mathbb{S}^{n_x}_{++}$ and a time-varying matrix $K \in \mathbb{R}^{n_u \times n_x}$. The correct mathematical formulation is given as:

$$\mathcal{F} = \varepsilon_Q \times \varepsilon_{KQK^T},\ \varepsilon_Q \subseteq \mathcal{X},\ \varepsilon_{KQK^T} \subseteq \mathcal{U} \tag{3.47}$$

On one hand, the invariance of a quadratic funnel is a consequence of the invariance of the level sets of the Lyapunov function defined in (3.44), which, simultaneously, needs to meet the quadratic stability condition (3.43). On the other hand, the feasibility of a quadratic funnel is guaranteed by forcing the ellipsoids ε_Q and ε_{KQK^T} to be inside the feasible sets of states and control inputs.

The next step is to derive the formal mathematical conditions that guarantee both invariance and feasibility when synthesizing the quadratic funnels.

Invariance

As explained above, the invariance of a quadratic funnel comes as a consequence of meeting the necessary and sufficient condition for quadratic stability, expressed in (3.43). To apply this condition to system (3.42), and given the Lyapunov function defined in (3.44), it is reformulated as following:

$$\dot{V}(t, \eta(t)) \leq -\alpha V(t, \eta(t)),\ \forall\, t \in [t_0, t_f],\ \forall\, q \in \varepsilon_{C_{cl} Q C_{cl}^T},\ p = \phi(q) \tag{3.48}$$

The decay rate α is positive, which, consequently, is a guarantee that $||\eta(t)||_2$ strictly decreases with time. That implies that the system's trajectory converges to the reference.

The terms "$\forall\, q \in \varepsilon_{C_{cl} Q C_{cl}^T},\ p = \phi(q)$" of condition (3.48) are a consequence of the invariance of the quadratic funnel. If $\eta \in \varepsilon_Q$ for all $t \in [t_0, t_f]$, then $q \in \varepsilon_{C_{cl} Q C_{cl}^T}$ for all $t \in [t_0, t_f]$. The problem that now arises is how to reformulate "$\forall\, q \in \varepsilon_{C_{cl} Q C_{cl}^T},\ p = \phi(q)$" in a way that can be included inside the quadratic form of the Lyapunov function. For that purpose, the concept of local multiplier matrices is leveraged [80, 81], which allows to rewrite the above condition as:

$$\dot{V}(t,\eta) \leq -\alpha V(t,\eta),\ \forall\, t \in [t_0, t_f],\ \forall \begin{bmatrix} q \\ \phi(q) \end{bmatrix}^T M \begin{bmatrix} q \\ \phi(q) \end{bmatrix} \geq 0, M \in \mathcal{M}_{\phi,\varepsilon_{C_{cl}QC_{cl}^T}} \tag{3.49}$$

where:

$$\mathcal{M}_{\phi,\varepsilon_{C_{cl}QC_{cl}^T}} = \left\{ M \in \mathbb{S}^{n_q+n_p} \middle| \begin{bmatrix} q \\ \phi(q) \end{bmatrix}^T M \begin{bmatrix} q \\ \phi(q) \end{bmatrix} \geq 0,\ \text{for all } q \in \varepsilon_{C_{cl}QC_{cl}^T} \right\} \tag{3.50}$$

Expanding the Lyapunov function and its derivative, while considering the system's dynamics (3.42), the above condition is reformulated as:

$$\begin{bmatrix} \eta \\ p \end{bmatrix}^T \begin{bmatrix} A_{cl}^T Q^{-1} + Q^{-1}A_{cl} - Q^{-1}\dot{Q}Q^{-1} + \alpha Q^{-1} & Q^{-1}E \\ E^T Q^{-1} & 0 \end{bmatrix} \begin{bmatrix} \eta \\ p \end{bmatrix} \leq 0$$
$$\text{for all } \begin{bmatrix} \eta \\ p \end{bmatrix}^T \begin{bmatrix} C_{cl}^T & 0 \\ 0 & I \end{bmatrix} M \begin{bmatrix} C_{cl} & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} \eta \\ p \end{bmatrix} \geq 0 \tag{3.51}$$

(3.51) can be expressed as a single Matrix Inequality using the $\mathcal{S}$-procedure. $\mathcal{S}$-procedure is often applied when a condition that constraints a specific quadratic function to be negative when other quadratic forms are also negative is encountered [82]. (3.51) is exactly one of those cases. Therefore, and after partitioning the matrix M as $M = \begin{bmatrix} M_{11} & M_{12} \\ M_{12}^T & M_{22} \end{bmatrix}$, (3.51) is verified if and only if there exists a scalar $\lambda \geq 0$ such that:

$$\begin{bmatrix} A_{cl}^T Q^{-1} + Q^{-1}A_{cl} - Q^{-1}\dot{Q}Q^{-1} + \alpha Q^{-1} + \lambda C_{cl}^T M_{11} C_{cl} & Q^{-1}E + \lambda C_{cl}^T M_{12} \\ E^T Q^{-1} + \lambda M_{12}^T C_{cl} & \lambda M_{22} \end{bmatrix} \leq 0 \tag{3.52}$$

The Matrix Inequality expressed in (3.52) cannot be solved using convex optimization techniques since one of the variables is the inverse of the matrix Q, not Q itself. For that reason, a reformulation is required. Both sides are multiplied by $diag[Q; I]$ to obtain:

$$\begin{bmatrix} QA_{cl}^T + A_{cl}Q - \dot{Q} + \alpha Q + \lambda Q C_{cl}^T M_{11} C_{cl} Q & E + \lambda Q C_{cl}^T M_{12} \\ E^T + \lambda M_{12}^T C_{cl} Q & \lambda M_{22} \end{bmatrix} \leq 0 \tag{3.53}$$

On top of that, by imposing $Q > 0$, $\lambda \geq 0$ and $M \in \mathcal{M}_{\phi,\varepsilon_{C_{cl}QC_{cl}^T}}$, the complete conditions that need to be satisfied in order for the quadratic funnel $\mathcal{F}$ to be invariant are now obtained.

(3.53) is a nonlinear Differential Matrix Inequality (DMI) with four unknown variables: the matrix Q of the quadratic Lyapunov function, the local multiplier matrix M, the controller gain K and the scalar λ.

For readability and convenience purposes, from this point on, the DMI (3.53) is rewritten as:

$$\begin{bmatrix} F - \dot{Q} + \lambda G^T M_{11} G & E + \lambda G^T M_{12} \\ E^T + \lambda M_{12}^T G & \lambda M_{22} \end{bmatrix} \leq 0 \tag{3.54}$$

where:

$$\begin{aligned} F &= QA^T + AQ + BY + Y^T B^T + \alpha Q \\ G &= CQ + DY \\ Y &= KQ \end{aligned} \tag{3.55}$$

Feasibility

Now, one needs to understand which conditions need to be met in order to guarantee that the funnel is contained inside a feasible region. As said before, the feasible sets for state and control vectors are denoted by $\mathcal{X} \subset \mathbb{R}^{n_x}$ and $\mathcal{U} \subset \mathbb{R}^{n_u}$, respectively. Additionally, $\bar{x} \in int(\mathcal{X})$ and $\bar{u} \in int(\mathcal{U})$, which means that the reference trajectory is located in the interior of the two respective sets and never intersects its borders. At the same time, $\mathcal{X}_f \subset \mathbb{R}^{n_x}$ is the set of acceptable state deviations at $t = t_f$.

The first step is to find the largest possible funnel that is able to fit inside the feasible region. It is denoted by $\mathcal{F}_{\text{max}}$ and characterized as following:

$$\mathcal{F}_{\text{max}} = \varepsilon_{Q_{\text{max}}} \times \varepsilon_{R_{\text{max}}},\ \varepsilon_{Q_{\text{max}}} \subset \mathcal{X},\ \varepsilon_{R_{\text{max}}} \subset \mathcal{U},\ \varepsilon_{Q_{\text{max}}} \subset \mathcal{X}_f \text{ for } t = t_f \tag{3.56}$$

where Q_{max} and R_{max} are two symmetric positive-definite matrices with size equal to the number of states and the number of inputs, respectively.

It is assumed that the set $\mathcal{X}$ can be constructed through inequality constrains as following:

$$\mathcal{X} = \left\{ x \mid h_i(x) \leq 0, i = 1, \ldots, m_x,\ ||x||_2 \leq x_{\text{max}} \right\} \tag{3.57}$$

where each $h_i(x)$ is a scalar function dependent on the state vector. m_x is the number of inequality constraints required to define the set of feasible states.

The set $\mathcal{X}$ can be approximated into a polytopic set $\mathcal{P}_{\bar{x}}$ by using information from the reference trajectory and turning the inequality constraints into affine functions. This approximation can be done by directly linearizing the state functions $h_i(x)$ around the reference trajectory. Hence, $P_{\bar{x}}$ is defined as:

$$\mathcal{P}_{\bar{x}} = \left\{ x \mid a_i^T x \leq b_i, i = 1, \ldots, m_x \right\} \cap \left\{ x \mid ||x||_2 \leq x_{\text{max}} \right\} \tag{3.58}$$

If the state functions $h_i(x)$ are already affine, then $\mathcal{X} = \mathcal{P}_{\bar{x}}$.

Then, the matrix Q_{max} is computed such that $\bar{x} \oplus \varepsilon_{Q_{\text{max}}}$ corresponds to the maximum volume ellipsoid centered at the reference trajectory that fits inside the set $\mathcal{P}_{\bar{x}}$. So, $\bar{x} \oplus \varepsilon_{Q_{\text{max}}} \subset \mathcal{P}_{\bar{x}}$. Hence, Q_{max} for any time $t \in [t_0, t_f]$ can be computed through [82]:

$$\begin{aligned} Q_{\text{max}}^{\frac{1}{2}}(t) = \text{arg} \quad & \max_{Z} \ \log\det(Z) \\ \text{subject to} \quad & ||Z a_i(t)||_2 + a_i(t)^T \bar{x}(t) \leq b_i(t), \quad i = 1, \ldots, m_x \\ & 0 \leq Z \leq x_{\text{max}} I_{n_x} \end{aligned} \tag{3.59}$$

When $t = t_f$, the data used to construct the affine constraints has to be relative to $\mathcal{X}_f$. This way, it is

guaranteed that $\varepsilon_{Q_{\max}} \subset \mathcal{X}_f$ for $t = t_f$.

The same approach can be used to compute $R_{\max}$. The set of feasible inputs $\mathcal{U}$ is constructed as:

$$\mathcal{U} = \left\{ u \mid h_j(u) \leq 0, j = 1, \ldots, m_u, \ ||u||_2 \leq u_{\max} \right\} \tag{3.60}$$

with m_u being the number of inequality constraints necessary to completely define $\mathcal{U}$.

Again, $\mathcal{U}$ can be approximated into a polytopic set $\mathcal{P}_{\bar{u}}$ using information from the reference trajectory:

$$\mathcal{P}_{\bar{u}} = \left\{ u \mid a_j^T u \leq b_j, j = 1, \ldots, m_u \right\} \cap \left\{ u \mid ||u||_2 \leq u_{\max} \right\} \tag{3.61}$$

The optimization problem to compute $R_{\max}$ for any time $t \in [t_0, t_f]$ is given as:

$$\begin{aligned} R_{\max}^{\frac{1}{2}}(t) = \text{arg} \quad & \max_{Z} \ \log\det(Z) \\ \text{subject to} \quad & ||Z a_j(t)||_2 + a_j(t)^T \bar{u}(t) \leq b_j(t), \ \ j = 1, \ldots, m_u \\ & 0 \leq Z \leq u_{\max} I_{n_u} \end{aligned} \tag{3.62}$$

Following this procedure, $Q_{\max}$ and $R_{\max}$, which define the largest ellipsoids fully contained within the feasible state and control sets, respectively, have just been obtained. Now, these matrices must be included in the quadratic funnel formulation. Consider the conditions below, which must be satisfied to guarantee the funnel feasibility:

$$\varepsilon_Q \subseteq \varepsilon_{Q_{\max}} \implies Q \leq Q_{\max} \qquad \varepsilon_{KQK^T} \subseteq \varepsilon_{R_{\max}} \implies KQK^T \leq R_{\max} \tag{3.63}$$

The first inequality is affine, but the second is nonconvex in the variables Q and K. A reformulation is, for that reason, required. Using the Schur complement and multiplying both sides by $diag[Q; I_{n_u}]$, it can be rewritten as:

$$\begin{aligned} KQK^T \leq R_{\max} \iff R_{\max} - KQK^T \geq 0 \iff \begin{bmatrix} Q^{-1} & K^T \\ K & R_{\max} \end{bmatrix} \geq 0 \iff \\ \iff \begin{bmatrix} Q & 0 \\ 0 & I_{n_u} \end{bmatrix} \begin{bmatrix} Q^{-1} & K^T \\ K & R_{\max} \end{bmatrix} \begin{bmatrix} Q & 0 \\ 0 & I_{n_u} \end{bmatrix} \geq 0 \iff \begin{bmatrix} Q & Y^T \\ Y & R_{\max} \end{bmatrix} \geq 0 \end{aligned} \tag{3.64}$$

where $Y = KQ$, as defined in (3.55). So, $Q \leq Q_{\max}$ and (3.64) guarantee that the quadratic funnel defined by the ellipsoids ε_Q and ε_{KQK^T} is fully contained inside a state and control feasible region.

Final Formulation

The complete structure of the standard optimization problem that synthesizes a continuous-time quadratic funnel for the nonlinear system (3.37) can be summarized as:

$$
\begin{aligned}
\underset{Q(t),Y(t),\lambda(t),M(t)}{\max} \quad & \log\det(Q(t_0)) \\
\text{subject to} \quad & 0 < Q \leq Q_{\max},\ \forall t \in [t_0, t_f] \\
& \lambda \geq 0,\ \forall t \in [t_0, t_f] \\
& M \in \mathcal{M}_{\phi,\varepsilon_{GQ^{-1}G^T}},\ \forall t \in [t_0, t_f] \\
& \begin{bmatrix} F - \dot{Q} + \lambda G^T M_{11} G & E + \lambda G^T M_{12} \\ E^T + \lambda M_{12}^T G & \lambda M_{22} \end{bmatrix} \leq 0,\ \forall t \in [t_0, t_f] \\
& \begin{bmatrix} Q & Y^T \\ Y & R_{\max} \end{bmatrix} \geq 0,\ \forall t \in [t_0, t_f]
\end{aligned}
\tag{3.65}
$$

This is a complex optimization problem because it is nonconvex, nonlinear and with four unknown matrix variables. Additionally, the set $\mathcal{M}_{\phi,\varepsilon_{GQ^{-1}G^T}}$ cannot be precisely defined but, instead, only approximated [8]. Numerical methods to solve the above optimization problem are discussed in the next Section.

Once the quadratic funnel and the feedback controller have been computed, the control input and the state trajectory are given by numerical integration, through the following expressions:

$$
\begin{aligned}
u(t) &= \bar{u}(t) + K(t)(x(t) - \bar{x}(t)) \\
x(t) &= x(t_0) + \int_{t_0}^{t} f(x(\tau), u(\tau)) d\tau
\end{aligned}
\tag{3.66}
$$

Given the invariance property, for any initial condition $x(t_0)$ such that $x(t_0) - \bar{x}(t_0) \in \varepsilon_{Q(t_0)}$, it is guaranteed that, for all $t \in [t_0, t_f]$, $x - \bar{x} \in \varepsilon_Q$ and $u - \bar{u} \in \varepsilon_{KQK^T}$. Also, the system's trajectory is feasible in respect to the sets $\mathcal{X}$ and $\mathcal{U}$. Finally, the action u is stabilizing for the nonlinear system (3.37).

3.5.2 Solving the Funnel Synthesis Problem

To solve the quadratic funnel synthesis optimization problem, a method called the γ-iteration was proposed [8, 76]. Its name is related with the fact that the matrix M is decomposed as:

$$
M_\gamma = \begin{bmatrix} \gamma^2 I & 0 \\ 0 & -I \end{bmatrix}
\tag{3.67}
$$

M_γ is a valid local multiplier matrix if γ is a local Lipschitz constant for the nonlinear function ϕ over the set $\varepsilon_{GQ^{-1}G^T}$.

The γ-iteration is an iterative method where the solution is obtained by fixing a set of variables, solving for the remaining ones and then switch. This working principle has already proved to be successful in other contexts, such as generation of funnel libraries [7] or iteratively solving the Ricatti equation [83]. In the specific case above, the problem becomes convex relative to the variable M if the variables (Q, Y, λ) are fixed. On the other hand, if M is fixed and (Q, Y, λ) are the variables to be solved for, a second convex optimization problem is obtained.

Firstly, the problem where the variables (Q, Y, λ) are fixed is considered. Given the decomposition in (3.67), the following reformulation is possible:

$$\begin{bmatrix} q \\ p \end{bmatrix}^T M \begin{bmatrix} q \\ p \end{bmatrix} \geq 0 \iff \begin{bmatrix} q^T & p^T \end{bmatrix} \begin{bmatrix} \gamma^2 I & 0 \\ 0 & -I \end{bmatrix} \begin{bmatrix} q \\ p \end{bmatrix} \iff \begin{bmatrix} \gamma^2 q^T & -p^T \end{bmatrix} \begin{bmatrix} q \\ p \end{bmatrix} \geq 0 \iff$$
$$\iff \gamma^2 q^T q \geq p^T p \iff \gamma ||q||_2 \geq ||p||_2 \tag{3.68}$$

By defining a matrix $\Delta \in \mathbb{R}^{n_p \times n_q}$ such that $p = \Delta q$ and $||\Delta||_2 \leq \gamma$, the nonlinear function ϕ can be replaced and the dynamics of the closed-loop system are written as:

$$\dot{\eta} = (A_{cl} + E\Delta C_{cl})\eta \tag{3.69}$$

The difficulty now lies in computing a valid value for γ. There are several possible procedures designed for the purpose of estimating a Lipschitz constant for a given function [84], but the focus will be placed on a sampling-based procedure.

To find the value of γ, two different optimization problems need to be formulated and solved, one inside the other. The goal is to find the point η in the funnel defined by ε_Q that maximizes $||\Delta||_2$, which needs to be equal or less than γ.

The outer optimization problem is given by:

$$\begin{aligned} \gamma = \max_{\eta} \quad & \Gamma^*(\eta) \\ \text{subject to} \quad & \eta \in \varepsilon_Q \iff \eta^T Q^{-1} \eta \leq 1 \end{aligned} \tag{3.70}$$

On the other hand, the inner optimization problem is given by:

$$\begin{aligned} \Gamma^*(\eta) = \min_{\Delta} \quad & ||\Delta||_2 \\ \text{subject to} \quad & \dot{\eta} = (A_{cl} + E\Delta C_{cl})\eta \end{aligned} \tag{3.71}$$

The latter optimization problem focuses on computing the smallest matrix Δ that satisfies the dynamics of the closed-loop system at a particular point η. The former finds the point η that maximizes the Γ^* inside the funnel. The maximum value of Γ^* corresponds to the estimated value of the Lipschitz constant γ.

In fact, (3.70) must be solved through a spatiotemporal sampling procedure. If different states $\{\eta_s\}_{s=1}^{N_s}$ are collected from the state-funnel ε_Q, one can obtain a bound of the Lipschitz constant as:

$$\gamma = \max_{s=1,\ldots,N_s} \quad \Gamma^*(\eta_s) \tag{3.72}$$

By selecting the sample from $\{\eta_s\}_{s=1}^{N_s}$ that corresponds to the maximum value of Γ^*, the solution for (3.70) is found. It has been found that a number of points of an order of magnitude of 10^4 must be collected in order to obtain reasonably accurate results.

During the implementation process, one issue regarding this step of the solution method was found:

the equality constraint in (3.71) was not exactly satisfied. In some rare cases, it would actually lead to an infeasible solution. Therefore, to solve this problem, a virtual control term v was added to the constraint and penalized in the cost function. So, (3.71) is reformulated to:

$$\begin{aligned} \Gamma^*(\eta) = \min_{\Delta} \quad & ||\Delta||_2 + w_{vc} v \\ \text{subject to} \quad & \dot{\eta} = (A_{cl} + E\Delta C_{cl})\eta + v \end{aligned} \tag{3.73}$$

where w_{vc} is a large weight, defined by the user.

After obtaining the Lipschitz constant γ, one can advance to the second part of the iterative procedure. Now, M is fixed and a solution for the triple (Q, Y, λ) is found.

Given the M_γ structure in (3.67), it is certain that $M_{11} \geq 0$ and (3.54) can be rewritten as:

$$\begin{bmatrix} F - \dot{Q} & E \\ E^T & -\lambda I \end{bmatrix} + \begin{bmatrix} \gamma G^T \\ 0 \end{bmatrix} \lambda I \begin{bmatrix} \gamma G & 0 \end{bmatrix} \leq 0 \tag{3.74}$$

If $E \neq 0$ then $\lambda > 0$. For that purpose, by resorting to the Schur complement and multiplying both sides by $diag(I, \lambda^{-1} I, I)$, an equivalent condition is obtained. This final condition is a Differential Linear Matrix Inequality (DLMI) in the variables Q, Y and $\nu = \lambda^{-1}$ and is expressed as:

$$\begin{bmatrix} F - \dot{Q} & \nu E & \gamma G^T \\ \nu E^T & -\nu I & 0 \\ \gamma G & 0 & -\nu I \end{bmatrix} \leq 0 \tag{3.75}$$

However, to make the above optimization problem tractable, some assumptions must be made regarding $A(t)$, $B(t)$, $Q(t)$, $Q_{\max}(t)$, $R_{\max}(t)$ and $Y(t)$. It is considered that each of these matrix-valued functions can be approximated by a convex combination such that $\square(t) = \sum_{i=1}^{n_M} \sigma_i(t) \square_i$, for some integer number $n_M > 1$, constant matrices $\square_i$ and interpolating functions $\sigma_i(t)$ such that $\sum_{i=1}^{n_M} \sigma_i(t) = 1$ [85]. The symbol $\square$ can stand for A, B, Q, $Q_{\max}$, $R_{\max}$ or Y. On top of that, ν and γ are assumed to be piecewise constant for a given time interval between two consecutive nodes. So, they are rewritten as ν_i and γ_i, $\forall i = 1, \ldots, n_M$.

The above matrix temporal decomposition is important to guarantee that the solution of the optimization problem satisfies the conditions for all $t \in [t_0, t_f]$, while being tractable and requiring only a finite set of LMIs. The DLMI (3.75) can be written as a set of $\frac{1}{2} n_M (n_M + 1)$ LMIs, while the others can be expressed as a set of LMIs where each individual matrix is constrained. For example, the last condition of the optimization problem (3.65) is equivalent to:

$$\begin{bmatrix} Q & Y^T \\ Y & R_{\max} \end{bmatrix} = \sum_{i=1}^{n_M} \sigma_i(t) \begin{bmatrix} Q_i & Y_i^T \\ Y_i & R_{max.i} \end{bmatrix} \geq 0 \tag{3.76}$$

By requiring each matrix of the right-hand term to be positive-semidefinite, the continuous-time condition is satisfied.

In the specific case of this thesis project, n_M has the same interpretation as the number of temporal nodes used to compute the reference trajectory through the SCvx algorithm, so $n_M = N$. That

means that, for $i = 1, \ldots, n_M$, the constant matrices A_i and B_i are obtained by linearizing the nonlinear continuous-time equations of motion around each temporal node of the reference trajectory. On the other hand, $Q_{\text{max},i}$ and $R_{\text{max},i}$ are computed through (3.59) and (3.62), respectively, by using the information of the reference trajectory at each temporal node. Finally, each node is also associated with a matrix Q_i and Y_i, to be computed by the given optimization problem.

Additionally, to arrive to the final form of the optimization problem, a theoretical remark is leveraged. It states that, to guarantee $M(t)N(t) \leq 0,\ t \in [t_0; t_f]$ with $M(t) = \sum_{i=1}^{n_M} \sigma_i(t) M_i$ and $N(t) = \sum_{i=1}^{n_M} \sigma_i(t) N_i$, it is sufficient to enforce $M_i N_i \leq 0,\ \forall i = 1, \ldots, n_M$ and $M_i N_j + M_j N_i \leq 0,\ \forall i = 1, \ldots, n_M - 1,\ \forall j = i+1, \ldots, n_M$. The proof lies in [8].

Hence, the final structure of the optimization problem for the variables (Q, Y, ν) is given as [85]:

$$
\begin{aligned}
\max_{Q_i, Y_i, \nu_i} \quad & \log\det(Q_1) \\
\text{s.t.} \quad & \nu_i \geq 0 \\
& 0 < Q_i \leq Q_{\text{max},i},\ i = 1, \ldots, n_M \\
& \begin{bmatrix} F_{ii} - \dot{Q}_i & \nu_i E & \gamma_i G_i^T \\ \nu_i E^T & -\nu_i I & 0 \\ \gamma_i G_i & 0 & -\nu_i I \end{bmatrix} \leq 0,\ i = 1, \ldots, n_M \\
& \begin{bmatrix} F_{ij} + F_{ji} - 2\dot{Q} & 2\nu_i E & \gamma_i G_i^T + \gamma_j G_j^T \\ 2\nu_i E^T & -2\nu_i I & 0 \\ \gamma_i G_i + \gamma_j G_j & 0 & -2\nu_i I \end{bmatrix} \leq 0, \quad \begin{matrix} i = 1, \ldots, n_M - 1 \\ j = i+1, \ldots, n_M \end{matrix} \\
& \begin{bmatrix} Q_i & Y_i^T \\ Y_i & R_{\text{max},i} \end{bmatrix} \geq 0,\ i = 1, \ldots, n_M
\end{aligned}
\tag{3.77}
$$

where:

$$
\begin{aligned}
F_{ij} &= Q_i A_j^T + A_j Q_i + B_j Y_i + Y_i^T B_j^T + \alpha Q_i \\
G_i &= C Q_i + D Y_i
\end{aligned}
\tag{3.78}
$$

Now that the two fundamental optimization problems that, together, form the iterative procedure of the γ-iteration have been understood, one should establish a criteria which needs to be satisfied in order to terminate the algorithm. For this purpose, the principle to be employed is to decrease the upper bound for the state funnel, given by the matrix Q_{max}, until the volume occupied by the ellipsoid ε_Q is approximately equal to the volume of $\varepsilon_{Q_{\text{max}}}$. For that reason, a parameter κ, called the fill ratio, is defined as:

$$
\kappa = \min_{i=1,\ldots,n_x} \left(\frac{\text{proj}_i\ \varepsilon_Q}{\text{proj}_i\ \varepsilon_{Q_{\text{max}}}} \right)^{\frac{1}{2}}
\tag{3.79}
$$

where $\text{proj}_i\ \varepsilon_Q$ is the projection of the state funnel ε_Q onto dimension i. More specifically, this projection represents the maximum distance that ε_Q extends along the i-th dimensional axis. This value can be obtained through a Cholesky factorization of the corresponding matrices.

Every time a new solution for the optimization variable Q is obtained, the fill ratio κ is computed and compared to a threshold value. If it is above, convergence is assumed to be reached and the algorithm is terminated. However, if it is below, the algorithm proceeds to the next iteration but using a smaller ellipsoid $\varepsilon_{Q_{\text{max}}}$. For that purpose, a new parameter, denoted as ζ and called contraction factor, is defined. ζ is computed through:

$$\zeta = \zeta_{\text{min}} + (1 - \zeta_{\text{min}}) \frac{1}{1 + e^{h(0.5-\kappa)}} \tag{3.80}$$

where ζ_{min} is the minimum contraction factor and h is the width parameter. Both are defined by the user at the beginning of the algorithm. So, by multiplying Q_{max} and Y_{max} by the contraction factor ζ, the new maximum feasible sets to be used for the next iteration are obtained.

Using this termination criteria, the algorithm is guaranteed to converge to a solution in a finite number of iterations [8].

To sum up, the complete γ-iteration method, which solves the funnel synthesis optimization problem (3.65) and outputs not only the funnel parameters but also the feedback controller, is summarized in Algorithm 1.

Algorithm 1 The γ-iteration method, designed to solve the quadratic funnel synthesis problem

Require: A reference trajectory $\{\bar{x}(t), \bar{u}(t)\}_{t=t_0}^{t_f}$; the system matrices in (3.40); a decay rate $\alpha > 0$; a suitable description of the sets $\mathcal{X}$, $\mathcal{X}_f$ and $\mathcal{U}$; a convergence tolerance $0 < \kappa_{\text{tol}} < 1$; a maximum number of iterations i_{max}; parameters n_M, N_s, ζ_{min} and h.

```
Compute Q_max and Y_max through (3.59) and (3.62);
Initialize the Lipschitz constant γ as γ_i = 0, ∀i = 1, ..., n_M;
Compute {Q_0, Y_0} through (3.77);
Set Q_max,1 = Q_0 and Y_max,1 = Y_0;
for i = 1 : i_max do
    Obtain new estimations for the Lipschitz constant γ through (3.72) and (3.73), with Q_max,i and Y_max,i;
    Compute {Q_i, Y_i} through (3.77);
    if κ_i(t_0) ≥ κ_tol then
        Convergence reached;
        Terminate the algorithm;
    else
        Compute ζ through (3.80);
        Q_max,i+1 = ζ Q_max,i and Y_max,i+1 = ζ Y_max,i;
    end if
end for
```

3.5.3 Balancing the Funnel Size

In the original quadratic funnel synthesis optimization problem expressed in (3.65), the goal is to maximize the funnel size. This can be beneficial as a larger control invariant set will guarantee that there is a higher probability that the initial state will converge to the reference trajectory and not leave a safe region. On the other hand, as perturbations and uncertainty come into play, minimizing the funnel size also helps decreasing the effects of these unknown factors into the system over time.

For that reason, scientific articles related with the funnel synthesis topic can be separated into two groups according to their final objective: whether it is funnel size maximization [8, 66, 86] or minimization [7, 87, 88]. Nevertheless, it is possible to combine the advantages of both approaches into one single optimization problem. In [89], a method that balances the maximization of the size of the funnel entry and the minimization of the size of the remaining funnel is proposed. As a consequence, a larger set of possible initial states is guaranteed to converge to the nominal trajectory, while the uncertainty effects are attenuated.

This is not a complex procedure, since, as seen below, it is a matter of adapting the cost function of the funnel synthesis optimization problem. However, for the scope of this thesis project, the interest is in the practical results. The goal is to understand what is the impact of this approach on the spacecraft's landing trajectories and if it decisively helps the system reject the effects of external disturbances.

So, for this scenario, a new objective function is considered. It is given by:

$$\begin{aligned} & J = -w_{Q_0} \ \log\det(Q(t_0)) + \int_{t_0}^{t_f} w_Q \ \lambda^Q(t) dt \\ & \text{subject to } Q(t) \leq \lambda^Q(t) I \quad \forall t \in [t_0, t_f] \end{aligned} \tag{3.81}$$

where $\lambda^Q(t)$ is an auxiliary variable introduced to minimize the maximum eigenvalue of the matrices Q, while w_{Q_0} and w_Q are weights defined by the user. Combining the objective function and the additional constraint in (3.81) with the invariance and feasibility constraints derived in the previous cases allow to synthesize a funnel which maximizes the number of initial states contained in the control invariant set, while minimizing the effects of disturbances over time.

In order to make the optimization tractable, a temporal matrix decomposition procedure should be applied again. That leads to a reformulation of (3.81) into:

$$\begin{aligned} & J = -w_{Q_0} \ \log\det(Q_1) + \sum_{i=2}^{n_M} w_Q \ \lambda_i^Q \\ & \text{subject to } Q_i \leq \lambda_i^Q I \quad \forall i = 2, \ldots, n_M \end{aligned} \tag{3.82}$$

3.5.4 Funnel Formulation with Disturbances

The previous quadratic funnel synthesis problem formulation does not consider a system model where disturbances directly affect the dynamics. However, one of the objectives of the present thesis project is to evaluate how including such component in the formulation can affect the results.

The new nonlinear continuous-time system dynamics is given by:

$$\dot{x}(t) = f(t, x(t), u(t), w(t)) \tag{3.83}$$

where the only difference from the previous case is that a vector-valued function $w(t)$ is now considered, in order to represent an uncertainty effect or the external disturbances acting on the system. Additionally, it is assumed that $||w(.)||_\infty \leq 1$ where $||w(.)||_\infty := \sup_{t\in[t_0,t_f]}||w(t)||$. All the remaining assumptions and formulations are still valid. The reference trajectory is denoted as $\{\bar{x}(t), \bar{u}(t), \bar{w}(t)\}_{t=t_0}^{t_f}$, but a disturbance-

free solution, such that $\bar{w}(t) = 0,\ t \in [t_0, t_f]$, is chosen.

Once again, the nonlinear system model is rewritten into a linear time-varying one by considering the state and input differences $\eta := x - \bar{x}$ and $\xi := u - \bar{u}$. The new dynamics are given by:

$$\begin{aligned} \dot{\eta}(t) &= A(t)\eta(t) + B(t)\xi(t) + F(t)w(t) + Ep(t) \\ p(t) &= \phi(q(t)) \\ q(t) &= C\eta(t) + D\xi(t) + Gw(t) \end{aligned} \tag{3.84}$$

where $p(t)$ is a the term that gathers the system nonlinearities and is computed through the function ϕ, whose argument is $q(t)$. The time-varying matrices $A(t)$, $B(t)$ and $F(t)$ are first-order approximations of the original nonlinear system around the nominal trajectory and evaluated with respect to the state, control input and disturbance, respectively. Finally, E, C, D and G are time-invariant matrices.

A linear feedback controller is again considered, such that $\xi(t) = K(t)\eta(t)$, and the quadratic Lyapunov function is still defined as $V(t, \eta) := \eta^T(t)Q^{-1}(t)\eta(t)$. Therefore, the condition that should be imposed to the closed-loop system in order to guarantee invariance and attractiveness is expressed as [89]:

$$\begin{aligned} \dot{V}(t, \eta) \leq -\alpha V(t, \eta), \text{ for all } ||p(t)||_2 \leq \gamma(t)\ ||q(t)||_2 \\ \text{and } V(t, \eta) \geq ||w(t)||_2^2 \quad \forall t \in [t_0, t_f] \end{aligned} \tag{3.85}$$

for some decay rate $\alpha > 0$.

If there exists $Q : [t_0, t_f] \longrightarrow \mathbb{S}_{n_x}^{++}$, $Y : [t_0, t_f] \longrightarrow \mathbb{R}^{n_x \times n_u}$, $\nu : [t_0, t_f] \longrightarrow \mathbb{R}_{++}$, $\lambda_w > 0$ and $\alpha > 0$ such that the following DLMI is satisfied, then (3.85) is met for the the closed-loop system, with $K = YQ^{-1}$ [89]:

$$\begin{aligned} &\begin{bmatrix} M - \dot{Q} & \nu E & F & (CQ + DY)^T \\ \nu E^T & -\nu I & 0 & 0 \\ F^T & 0 & \lambda_w I & G^T \\ CQ + DY & 0 & G & -\nu \frac{1}{\gamma^2} I \end{bmatrix} \leq 0 \\ &M = AQ + QA^T + BY + Y^T B^T + \alpha Q + \lambda_w Q \end{aligned} \tag{3.86}$$

This condition can replace (3.75) in the quadratic funnel synthesis optimization problem and, consequently, funnel invariance is still guaranteed for a system where the uncertainty directly affects the system dynamics. It is important to note that the need for temporal decomposition remains.

3.6 MPC with Funnel Parameters Strategy

The first approach to be proposed within the scope of the current thesis project is a Model Predictive Control (MPC) algorithm which leverages the parameters from a funnel computed offline to formulate a cost function and a terminal constraint. The advantage of employing such formulation is outlined, by

comparing the corresponding performance with a standard MPC controller.

In that sense, the optimization problem to be solved at each temporal node is given by:

$$\begin{aligned}
\underset{\substack{x_1,\ldots,x_N \\ u_0,\ldots,u_{N-1}}}{\min} \quad & \sum_{i=1}^{N-1}(x_i-\bar{x}_i)^T Q_i^{-1}(x_i-\bar{x}_i) + w_N(x_N-\bar{x}_N)^T Q_N^{-1}(x_N-\bar{x}_N) + \\
& + \sum_{i=0}^{N-1}(u_i-u_{i-1})^T W_U(u_i-u_{i-1}) \\
\text{s.t.} \quad & x_{i+1}-\bar{x}_{i+1} = A_i(x_i-\bar{x}_i) + B_i(u_i-\bar{u}_i),\ i=0,\ldots,N-1 \\
& Sx_i + Ru_i <= h,\ i=0,\ldots,N-1 \\
& (x_N-\bar{x}_N)^T Q_N^{-1}(x_N-\bar{x}_N) \leq 1 \\
& x_0 \text{ given}
\end{aligned} \tag{3.87}$$

where N is the control horizon, w_N is an user-defined scalar weight designed to penalize even further the deviation at the terminal node N, W_U is an user-defined weight matrix, $\bar{x}_i$ is the state vector of the reference trajectory at temporal node i and Q_i is the positive-definite matrix that parameterizes the funnel at temporal node i.

On one hand, the constraint $Sx_i + Ru_i <= h,\ i = 0,\ldots,N-1$, imposes maximum limits on the state and control input variables, guaranteeing feasibility. On the other hand, the terminal constraint is added to guarantee that, according to the nominal model, the state at temporal node N is inside the given funnel. Note that the state funnel ε_Q is defined as $\varepsilon_Q = \{x \in \mathbb{R}^{n_x} \mid (x-\bar{x})^T Q^{-1}(x-\bar{x}) \leq 1\}$.

Additionally, the cost term $\sum_{i=0}^{N-1}(u_i-u_{i-1})^T W_U(u_i-u_{i-1})$ penalizes sudden and rough changes in the control input values, as such behavior could lead to an unrealistic and undesirable control strategy. Even though the control input variables have different units and, consequently, different orders of magnitude, the weight matrix W_U is defined in a way that gives equal preponderance to each one.

The error dynamics is used to describe the system evolution, as that allows to derive a linearized version of the dynamical model. As it was found during the implementation procedure, using the exact nonlinear equations of motion would result in a computational time larger than the interval between consecutive nodes. With a linearized dynamical model, the computational time is much more reasonable and the algorithm still has a high degree of accuracy.

Additionally, since this is a MPC optimization problem, the need to discretize the system arises. Therefore, the linear time-varying dynamics matrices A_i and B_i are obtained by linearizing the system around the reference trajectory, and then applying a forward Euler discretization method. The sampling instant is actually dependent on the time to land, which is determined by the SCvx algorithm, and the total number of temporal nodes, previously specified by the user.

Standard MPC Formulation

Nevertheless, one can only take strong conclusions regarding the performance of the proposed MPC method, which leverages the funnel parameters to formulate a cost function and a terminal constraint, when comparing it with a standard MPC controller.

For that reason, the following optimal control problem is formulated in order to simulate a typical MPC approach:

$$\begin{aligned}
\min_{\substack{x_1,\dots,x_N \\ u_0,\dots,u_{N-1}}} \quad & \sum_{i=1}^{N-1} (x_i - \bar{x}_i)^T W_X (x_i - \bar{x}_i) + (x_N - \bar{x}_N)^T P (x_N - \bar{x}_N) + \\
& + \sum_{i=0}^{N-1} (u_i - u_{i-1})^T W_U (u_i - u_{i-1}) \\
\text{s.t.} \quad & x_{i+1} - \bar{x}_{i+1} = A_i (x_i - \bar{x}_i) + B_i (u_i - \bar{u}_i),\ i = 0, \dots, N-1 \\
& S x_i + R u_i <= h,\ i = 0, \dots, N-1 \\
& x_N - \bar{x}_N \in \mathbb{X}_N \\
& x_0 \text{ given}
\end{aligned} \tag{3.88}$$

where W_X and W_U are user-defined weight matrices for the cost function, while P is the solution of the discrete-time algebraic Ricatti equation, used to define the terminal cost. To solve such equation, the dynamics matrices relative to the terminal node, A_N and B_N, are used.

Similarly to (3.87), the condition $S x_i + R u_i <= h,\ i = 0, \dots, N$ is a feasibility guarantee, and large changes in the control input values between two consecutive temporal nodes are also penalized.

Finally, the terminal constraint for the standard MPC controller formulation is $x_N - \bar{x}_N \in \mathbb{X}_N$, where $\mathbb{X}_N$ represents a control invariant set for the dynamical system $x_{i+1} - \bar{x}_{i+1} = (A_i + B_i K_i)(x_i - \bar{x}_i)$. Therefore, $x_N - \bar{x}_N \in \mathbb{X}_N \implies (A_N + B_N K_N)(x_N - \bar{x}_N) \in \mathbb{X}_N$. In this case, K is the feedback controller computed through the funnel synthesis method. $\mathbb{X}_N$ is computed through the *MPT3* toolbox, and, given its polyhedral form, the terminal condition can be enforced through a series of inequality constraints.

3.7 Real-Time Recomputation Algorithm

The second proposed method combines both trajectory and funnel recomputation steps during the real-time evolution of the system. For that purpose, a parameter N_1 is defined, which represents the shorter control horizon and determines in which temporal nodes the recomputation procedure is performed.

Considering a random initial condition, the first step is to compute, through the Successive Convexification (SCvx) algorithm, a reference trajectory from that given point to the goal state. Due to its unique principle, SCvx is considerably fast and converges to an accurate solution in few deciseconds. On top of that, the mentioned computation time decreases as the system approaches the goal state, because few temporal nodes are considered inside the SCvx algorithm.

After a reference trajectory from the initial condition to the goal state has been optimized, the matrices Q_{max} and R_{max} are computed through (3.59) and (3.62) and the continuous-time linearized dynamics matrices are obtained. Nevertheless, such computations are only performed for the subsequent N_1 temporal nodes. With these matrices, a funnel and a feedback controller are then computed, but, again,

only for the subsequent N_1 temporal nodes. For that purpose, the formulation relative to robust control invariant funnels is used. That implies that, essentially, (3.77) is solved but the invariance condition is substituted by (3.86). At the same time, the cost function maximizes the funnel entry and minimizes the funnel size, simultaneously.

As a result, the optimization problem relative to the funnel recomputation is formulated as:

$$
\begin{aligned}
\min_{Q_i, Y_i, \nu_i, \lambda_i^Q} \quad & -\log\det(Q_1) + \sum_{i=2}^{N_1} w_Q \lambda_i^Q \\
\text{s.t.} \quad & Q_i \leq \lambda_i^Q I,\ i = 2, \ldots, N_1 \\
& \nu_i \geq 0,,\ i = 1, \ldots, N_1 \\
& 0 < Q_i \leq Q_{\max,i},\ i = 1, \ldots, N_1 \\
& \begin{bmatrix} M_{ii} - \dot{Q} & \nu_i E & F & H_i^T \\ \nu_i E^T & -\nu_i I & 0 & 0 \\ F' & 0 & -\lambda_w I & G^T \\ H_i & 0 & G & -\nu_i \frac{1}{\gamma_i^2} I \end{bmatrix} \leq 0,\ i = 1, \ldots, N_1 \\
& \begin{bmatrix} M_{ij} + M_{ji} - 2\dot{Q} & 2\nu_i E & 2F & H_i^T + H_j^T \\ 2\nu_i E^T & -2\nu_i I & 0 & 0 \\ 2F' & 0 & -2\lambda_w I & 2G^T \\ H_i + H_j & 0 & 2G & -\nu_i \left(\frac{1}{\gamma_i^2} + \frac{1}{\gamma_j^2} \right) I \end{bmatrix} \leq 0, \quad \begin{matrix} i = 1, \ldots, N_1 - 1 \\ j = i+1, \ldots, N_1 \end{matrix} \\
& \begin{bmatrix} Q_i & Y_i^T \\ Y_i & R_{\max,i} \end{bmatrix} \geq 0,\ i = 1, \ldots, N_1
\end{aligned}
\tag{3.89}
$$

where:

$$
\begin{aligned}
M_{ij} &= Q_i A_j^T + A_j Q_i + B_j Y_i + Y_i^T B_j^T + \alpha Q_i + \lambda_w Q_i \\
H_i &= C Q_i + D Y_i
\end{aligned}
\tag{3.90}
$$

Since the above funnel synthesis optimization problem is formulated for a short control horizon, it can be quickly solved. This low computational demand allows the approach to be run in real-time, which is the main goal here. This iterative procedure is then repeated every N_1 temporal nodes, until a previously specified maximum number of nodes has been reached.

By recomputing the reference trajectory, the system can adapt to the effects of the uncertainty and disturbances, especially in terms of landing time. For example, if an external perturbation takes the spacecraft further away from the goal state, the landing procedure may need to take longer in order to guarantee that all the physical constraints are satisfied and, simultaneously, the landing location is reached. This is an aspect which is not contemplated in the previous approaches, where the landing time is fixed and independent of the disturbance effects.

Additionally, with real-time trajectory and funnel recomputation, one can be more confident that the system's trajectory will not deviate too much from the reference. The Lyapunov function defined to formu-

late the stability and invariance conditions of the funnel is given by $V(t, x(t)) = (x(t)-\bar{x}(t))^T Q^{-1}(t)(x(t)-\bar{x}(t))$, whereas the funnel is parameterized as $\varepsilon_Q = \{x \in \mathbb{R}^{n_x} \mid (x-\bar{x})^T Q^{-1}(x-\bar{x}) \leq 1\}$. With this proposed approach, by generating the reference trajectory from the initial condition and updating this given reference from time to time, the Lyapunov function value is, in several occasions, initialized close to 0. Then, the feedback controller computed through (3.89) is able to quickly determine the control actions for the following temporal nodes, correct deviations and make the system track as closely as possible the reference, even with unforeseen factors acting directly on it.

It is true that this approach can pose some issues in terms of recursive feasibility and stability guarantees. Such topic is discussed throughout the next subsections.

The real-time trajectory and funnel recomputation control algorithm can be summarized as:

Algorithm 2 Real-time trajectory and funnel recomputation algorithm

Require: The maximum number of temporal nodes N; the control horizon N_1; the decay rate α; the parameter λ_w; the scalar weight w_Q; the goal location x_N; the scaling matrices; the invariant matrices C, D, E, F and G.

```
Randomly initialize the trajectory and obtain x_1;
Set i = 1;
while i ≠ N do
    Compute the reference trajectory, using a total of N − i + 1 temporal nodes;
    Compute Q_max,i and R_max,i through (3.59) and (3.62), for the subsequent N_1 nodes;
    Obtain the dynamics matrices A_i and B_i for the subsequent N_1 nodes;
    Solve optimization problem (3.89);
    Set K_i = Y_i Q_i^{-1}, i = 1, ..., N_1;
    Simulate the closed-loop system to obtain x_{i+1}, i = 1, ..., N_1 − 1;
    Set i = i + N_1 − 1;
end while
```

Recursive Feasibility

As mentioned before, the real-time trajectory and funnel recomputation algorithm poses some doubts relative to recursive feasibility and stability issues. For that reason, some theoretical remarks regarding these two properties need to be made.

Firstly, to overcome recursive feasibility concerns, when a new reference trajectory is computed, it is imposed that the next N_1 state vectors, from the current one until the one where the next recomputation procedure is going to be performed, must be inside a previously computed control invariant funnel. That aspect, along with the fact that, due to the optimized time-varying feedback controller, the actual trajectory closely tracks the reference, guarantees that one can always find a feasible stabilizing control law that takes the system from the state at the N_1th node to the goal location. The premise here is that, if one can guarantee that the system keeps evolving inside of a control invariant set, it is always possible to compute a sequence of feasible control input values that take the system from the given current state to the final state. This conclusion is valid because, in the worst-case scenario, the feedback controller associated with the aforementioned funnel can be the one to propagate the trajectory to the goal location.

Additionally, after different simulations, it was found that the actual trajectory is always inside of the

several recomputed funnels. Since these funnels represent feasible control invariant sets, one can argue that the computed control actions are feasible.

Hence, by imposing, at the trajectory recomputation step, that the next N_1 reference states are within a feasible control invariant set and guaranteeing that the actual trajectory keeps evolving deep inside the different recomputed funnels, recursive feasibility is ensured.

Stability

Due to time constraints, it was not possible to come up with a strategy that mathematically proves that the real-time trajectory and funnel recomputation algorithm is stable. Any conclusion regarding stability is drawn from empirical observations of the different simulations, with distinct random initial conditions.

Given the fact that, in every test case, it was found that the actual trajectory evolves deep inside the recomputed funnels and converges to the reference, one can be confident that the proposed method leads to a stable control law. Nevertheless, it should be reinforced that a more formal analysis to the stability of the given approach is still required.

Chapter 4

Results

In the fourth Chapter, the main algorithm results, as well as short respective discussions, are presented. The findings of the different phases of the implementation are shown and explained, from the computation of a reference trajectory through the Successive Convexification (SCvx) algorithm, to the performance of the funnel synthesis method and the novel approaches.

Before any results are presented, it is important for the reader to know that the simulator was designed to replicate a spacecraft's landing procedure on Earth. For that reason, the values for the physical parameters used inside the dynamical model, which are listed in Table 4.1, are defined based on a SpaceX Falcon 9 rocket model. The others are related with the atmospheric conditions at sea level.

Table 4.1: Values for the simulation physical parameters

Physical parameter	Value	Units
Specific impulse (I_{sp})	280	s
Gravitational acceleration vector ($\mathbf{g}_{\mathcal{I}}$)	$[0;\ 0;\ -9.81]$	m/s^2
Atmospheric pressure (P)	10^5	Pa
Nozzle's area (A_{noz})	10.75	m^2
Atmospheric density (ρ)	1.293	kg/m^3
Drag coefficient matrix (C_A)	$0.7I_3$	-
Moment of inertia matrix ($J_{\mathcal{B}}$)	$diag\Big[3.643 \times 10^6; 3.643 \times 10^6; 4.38 \times 10^4\Big]$	kg/m^2
Engine's pivotal point position vector ($r_{T,\mathcal{B}}$)	$[0;\ 0;\ -3]$	m
Center of pressure position vector ($r_{cp,\mathcal{B}}$)	$[0;\ 0;\ 1.5]$	m

The values for the state and control input constraints, explained on Section 3.3, are shown in Table 4.2.

4.1 Reference Trajectory

The current Section shows the results of the implemented SCvx algorithm. Note that there are no toolboxes that directly apply this method, which means that its full implementation was written by hand from scratch. For that reason, it is important to make sure that the reference trajectory is feasible,

Table 4.2: Maximum values for the physical constraints

Constraint parameter	Value	Units
Vehicle's mass full on fuel (m_{wet})	45100	kg
Vehicle's mass with no fuel (m_{dry})	25000	kg
Maximum approach angle (γ_{max})	75	deg
Maximum tilt angle (γ_{max})	45	deg
Maximum velocity (v_{max})	30	m/s
Maximum angular velocity (ω_{max})	20	deg/s
Minimum thrust value (F_{min})	10^5	N
Maximum thrust value (F_{max})	10^6	N
Maximum torque value (τ_{max})	10^4	N/m
Maximum gimbal deviation angle (δ_{max})	30	deg
Maximum throttle rate ($\dot{F}_{z,\text{max}}$)	8×10^4	N/s
Maximum gimbal rate ($\dot{\delta}_{\text{max}}$)	10	deg/s

realistic and leads to a safe and accurate landing. This reference trajectory is the central piece around which the funnel will be constructed. Consequently, one must be certain that, at this stage, there are no undesirable results that could later negatively impact the performance of the funnel synthesis method.

For the results to be shown below, a total of $N = 30$ temporal nodes are used. The convergence tolerance δ_{tol} is set to 10^{-5}. The virtual control penalty weight w_{vc} and the trust region weight w_{tr} are initialized as 10^3 and 10, respectively.

The following boundary conditions are considered:

$$
\begin{aligned}
&t_0 : m = 45000 \text{ kg} \quad \mathbf{r}_{\mathcal{I}} = (200; 0; 300) \text{ m} \quad \mathbf{v}_{\mathcal{I}} = (-10; 0; -15) \text{ m/s} \quad \boldsymbol{\Theta} = (0; 0; 0) \text{ deg} \quad \omega_{\mathcal{B}} = (0; 0; 0) \text{ rad/s} \\
&t_f : m = 25100 \text{ kg} \quad \mathbf{r}_{\mathcal{I}} = (0; 0; 20) \text{ m} \quad \mathbf{v}_{\mathcal{I}} = (0; 0; -1) \text{ m/s} \quad \boldsymbol{\Theta} = (0; 0; 0) \text{ deg} \quad \omega_{\mathcal{B}} = (0; 0; 0) \text{ rad/s}
\end{aligned} \tag{4.1}
$$

To solve each Second-Order Cone Programming optimization subproblem, the MOSEK solver is used, through a YALMIP interface. It took 6 iterations until convergence was reached, totalling a computation time of 0.279 seconds.

After finding the optimal control inputs, the original continuous-time nonlinear system is simulated through a 4th order Runge-Kutta method. Figure 4.1 shows the evolution of the three position coordinates. As observed, the spacecraft is able to land softly, with the entire procedure taking around 25 seconds. As the spacecraft's center of mass is considered to be located 20 m above the planet's surface, the goal position $r_{\mathcal{I},f} = [0; 20; 0]$ m is reached perfectly.

The plot in Figure 4.2 shows how the velocity components, expressed in the three different axis, evolve with time. Velocity in the y-axis remains null for the entire trajectory, while the z-velocity keeps decreasing with time. There are some oscillations in the x-component but, as intended, it ends up reaching 0 m/s.

Figures 4.3 and 4.4 help understand how the spacecraft behaves in terms of rotational motion. The angular velocity components relative to the x- and z-axis are equal to zero for the entire trajectory, so the

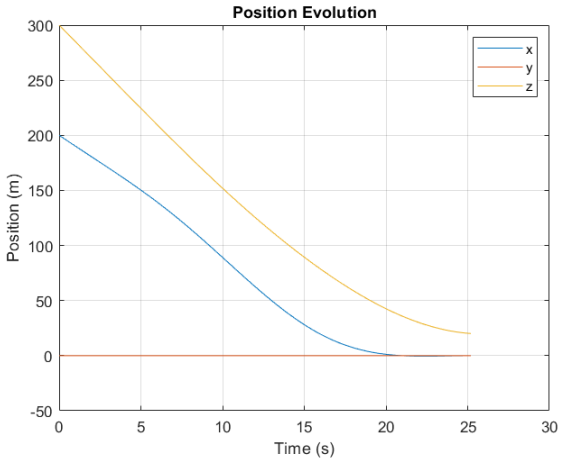

Figure 4.1: Position coordinates' evolution, for the SCvx algorithm implementation

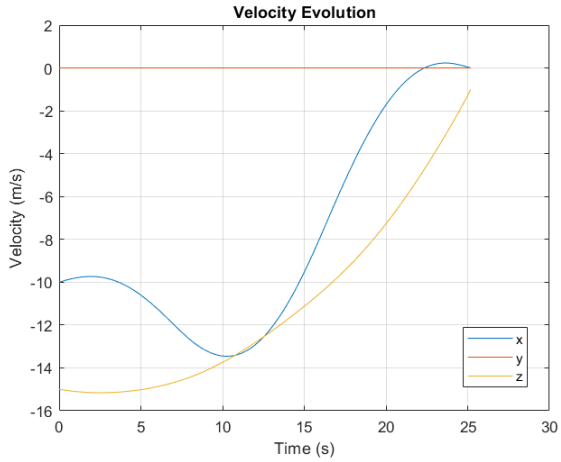

Figure 4.2: Velocity components' evolution, for the SCvx algorithm implementation

corresponding Euler angles do not change. However, it is possible to observe that the angle measured relative to the y-axis undergoes some variations, almost reaching a value of 8 degrees. Although this is not concerning, such behaviour can be explained by the fact that, in order for the spacecraft to go from $x_0 = 200$ m to $x_f = 0$ m, the gimbaled engine needs to deviate from the nominal position and, consequently, causes a rotation around the y-axis. Nevertheless, at the end of the landing procedure, the spacecraft is aligned in the upright position.

Figures 4.5 and 4.6 are two side-view perspectives of the spacecraft's trajectory evolution. The upright direction and the thrust force vectors are also visually represented, for analysis purposes. Please be aware of the scale of the plot in Figure 4.6, it is of an order of magnitude of 10^{-4}. If both axis had the same scale, a straight downwards trajectory would be shown.

All the discussed Figures make a case for the successful implementation of the SCvx algorithm, as the computed reference trajectory is feasible, realistic and achieves the goal of carrying the spacecraft from the initial position to the landing location. In addition, Figure 4.7 is an assurance that the spacecraft did not spend more fuel that what is available. In fact, the goal of minimizing fuel consumption is accomplished, since, at the end of the PDG maneuver, there is plenty of fuel still available.

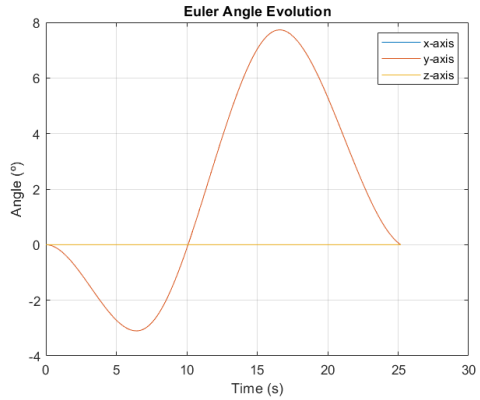

Figure 4.3: Euler angles' evolution, for the SCvx algorithm implementation

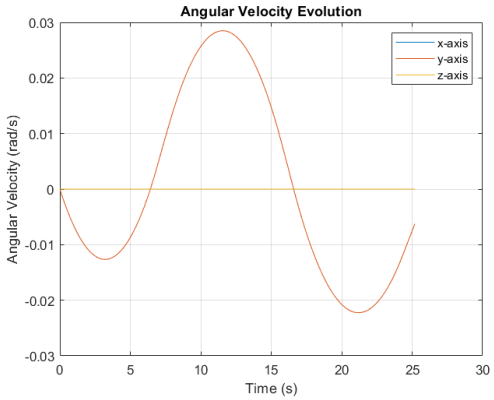

Figure 4.4: Angular velocity components' evolution, for the SCvx algorithm implementation

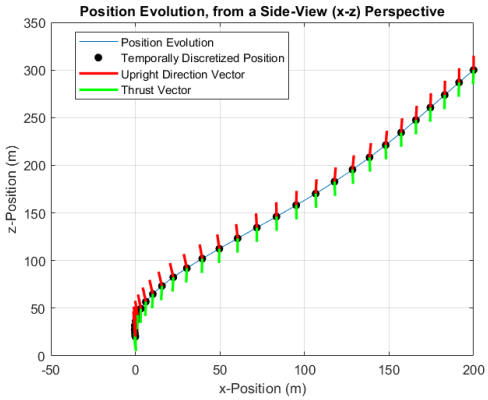

Figure 4.5: Position evolution, from a $X - Z$ perspective

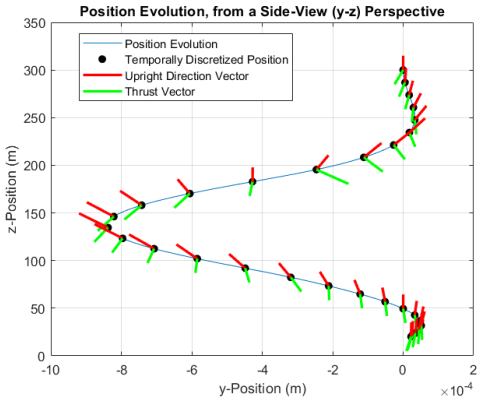

Figure 4.6: Position evolution, from a $Y - Z$ perspective

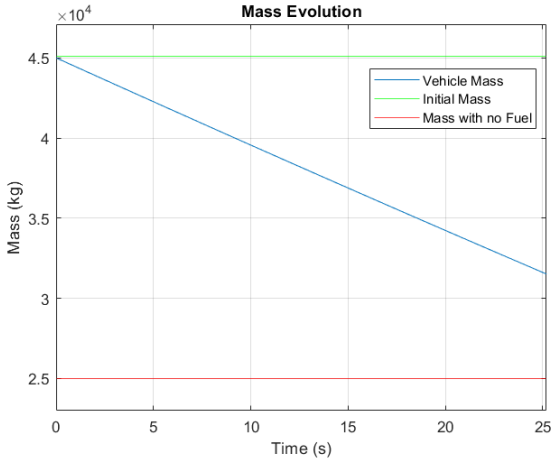

Figure 4.7: Vehicle's mass evolution, for the SCvx algorithm implementation

Hence, as it is possible to conclude that a perfect reference trajectory has been obtained from the SCvx algorithm, the focus should move on to the next phase of the thesis project: the performance of the funnel synthesis method.

4.2 Funnel Synthesis

4.2.1 Initial Funnel Formulation

First and foremost, to derive a linearized version of the original nonlinear system, which is then used to synthesize a funnel and design a reliable feedback controller, 9 different nonlinear channels are considered:

- From the thrust vector $\mathbf{F}_{\mathcal{B}}$ to the vehicle's mass evolution $\dot{m}$;
- From the vehicle's mass m to the velocity evolution $\dot{\mathbf{v}}$;
- From the vehicle's velocity $\mathbf{v}$ to its evolution $\dot{\mathbf{v}}$;
- From the vehicle's orientation $\boldsymbol{\Theta}$ to the velocity evolution $\dot{\mathbf{v}}$;
- From the vehicle's orientation $\boldsymbol{\Theta}$ to its evolution $\dot{\boldsymbol{\Theta}}$;
- From the vehicle's angular velocity ω to the orientation evolution $\dot{\boldsymbol{\Theta}}$;
- From the vehicle's velocity $\mathbf{v}$ to the angular velocity evolution $\dot{\omega}$;
- From the vehicle's orientation $\boldsymbol{\Theta}$ to the angular velocity evolution $\dot{\omega}$;
- From the vehicle's angular velocity ω to its evolution $\dot{\omega}$.

The constraints sets $\mathcal{X}, \mathcal{U}$ and $\mathcal{X}_f$ are constructed through the following conditions:

$$\begin{aligned}
\mathcal{X} &= \{\mathbf{x} \in \mathbb{R}^{n_x} \mid \mathbf{x}_{\text{lb}} \leq \mathbf{x} \leq \mathbf{x}_{\text{ub}},\ \delta\mathbf{x}_{\text{lb}} \leq \mathbf{x} - \bar{\mathbf{x}} \leq \delta\mathbf{x}_{\text{ub}}\} \\
\mathcal{U} &= \{\mathbf{u} \in \mathbb{R}^{n_u} \mid \mathbf{u}_{\text{lb}} \leq \mathbf{u} \leq \mathbf{u}_{\text{ub}},\ \delta\mathbf{u}_{\text{lb}} \leq \mathbf{u} - \bar{\mathbf{u}} \leq \delta\mathbf{u}_{\text{ub}}\} \\
\mathcal{X}_f &= \{\mathbf{x} \in \mathbb{R}^{n_x} \mid \mathbf{x}_{\text{lb}} \leq \mathbf{x} \leq \mathbf{x}_{\text{ub}},\ \delta\mathbf{x}_{\text{lb},f} \leq \mathbf{x} - \bar{\mathbf{x}} \leq \delta\mathbf{x}_{\text{ub},f}\}
\end{aligned} \tag{4.2}$$

where:

$$\begin{aligned}
&\mathbf{x}_{\text{lb}} = \{m_{\text{dry}};\ -300;\ -300;\ 0;\ -v_{\text{max}}\begin{bmatrix}1 & 1 & 1\end{bmatrix}^T;\ -\pi\begin{bmatrix}1 & 1 & 1\end{bmatrix}^T;\ -\omega_{\text{max}}\begin{bmatrix}1 & 1 & 1\end{bmatrix}^T\} \\
&\mathbf{x}_{\text{ub}} = \{m_{\text{wet}};\ 300;\ 300;\ 500;\ v_{\text{max}}\begin{bmatrix}1 & 1 & 1\end{bmatrix}^T;\ \pi\begin{bmatrix}1 & 1 & 1\end{bmatrix}^T;\ \omega_{\text{max}}\begin{bmatrix}1 & 1 & 1\end{bmatrix}^T\} \\
&\mathbf{u}_{\text{lb}} = \{-F_{\text{max}};\ -F_{\text{max}};\ F_{\text{min}};\ -\tau_{\text{max}}\begin{bmatrix}1 & 1 & 1\end{bmatrix}^T\} \\
&\mathbf{u}_{\text{ub}} = \{F_{\text{max}}\begin{bmatrix}1 & 1 & 1\end{bmatrix}^T;\ \tau_{\text{max}}\begin{bmatrix}1 & 1 & 1\end{bmatrix}^T\} \\
&\delta\mathbf{x}_{\text{lb}} = \{-5000;\ -100\begin{bmatrix}1 & 1 & 1\end{bmatrix}^T;\ -10\begin{bmatrix}1 & 1 & 1\end{bmatrix}^T;\ -15\frac{\pi}{180}\begin{bmatrix}1 & 1 & 1\end{bmatrix}^T;\ -30\frac{\pi}{180}\begin{bmatrix}1 & 1 & 1\end{bmatrix}^T\} \\
&\delta\mathbf{x}_{\text{ub}} = -\delta\mathbf{x}_{\text{lb}} \\
&\delta\mathbf{u}_{\text{lb}} = \{-5\times10^4\begin{bmatrix}1 & 1 & 1\end{bmatrix}^T;\ -10^3\begin{bmatrix}1 & 1 & 1\end{bmatrix}^T\} \qquad \delta\mathbf{u}_{\text{ub}} = -\delta\mathbf{u}_{\text{lb}} \\
&\delta\mathbf{x}_{\text{lb},f} = \{-500;\ -\begin{bmatrix}1 & 1 & 1\end{bmatrix}^T;\ -0.5\begin{bmatrix}1 & 1 & 1\end{bmatrix}^T;\ -3\frac{\pi}{180}\begin{bmatrix}1 & 1 & 1\end{bmatrix}^T;\ -\frac{\pi}{180}\begin{bmatrix}1 & 1 & 1\end{bmatrix}^T\} \\
&\delta\mathbf{x}_{\text{ub},f} = -\delta\mathbf{x}_{\text{lb},f}
\end{aligned} \tag{4.3}$$

The vectors $\mathbf{x}_{\text{lb}}$, $\mathbf{x}_{\text{ub}}$, $\mathbf{u}_{\text{lb}}$ and $\mathbf{u}_{\text{ub}}$ define the feasibility limits for the state and control input vectors, whereas all the vectors with the δ symbol guarantee that the system's trajectory and the input do not deviate too much from the reference. For $t = t_f$, the allowed maximum deviation is largely decreased, since the spacecraft's final state should be as close as possible to the goal state.

For the following results, the number of the temporal nodes is equal to the one from the reference trajectory algorithm. So, $n_M = N = 30$. The decay rate α is set to 0.05, while the convergence tolerance δ_{tol} is equal to 0.5. For the contraction procedure, the width parameter h and the minimum contraction factor ζ_{max} are set to 15 and 0.6, respectively. On the other hand, for the Lipschitz constant estimation procedure, a total of $N_s = 100$ samples are used for each temporal node, and γ_i is initialized as 0 for all nodes.

The MOSEK solver, combined with a YALMIP interface, is again used to solve the different optimization problems, whether it is related with the funnel synthesis or the Lipschitz constant estimation procedure. The algorithm is able to converge after 3 iterations. Each funnel synthesis optimization problem takes about 5 to 6 minutes to solve, while the γ estimation step requires 20 minutes of computation time.

One way to evaluate the size of the funnel is to check the values of its projections onto each of the n_x state dimensions. As mentioned before, a projection onto the i-th dimension represents the maximum distance that the funnel extends along the i-th dimensional axis, and can be obtained by applying a Cholesky factorization technique to the matrix Q. In this scenario, the funnel ε_Q has the following projections onto each of the state dimensions at the initial time $t = 0\ s$:

$$m:\ 99.98 \text{ kg} \quad \mathbf{r}_{\mathcal{I}}: \begin{bmatrix} 58.24 \\ 58.36 \\ 65.44 \end{bmatrix} \text{ m} \quad \mathbf{v}_{\mathcal{I}}: \begin{bmatrix} 9.31 \\ 9.34 \\ 8.56 \end{bmatrix} \text{ m/s} \quad \boldsymbol{\Theta}_{\mathcal{B}}: \begin{bmatrix} 14.44 \\ 14.44 \\ 14.86 \end{bmatrix} \text{ deg} \quad \omega_{\mathcal{B}}: \begin{bmatrix} 0.1224 \\ 0.1226 \\ 0.1065 \end{bmatrix} \text{ rad/s} \tag{4.4}$$

Then, to assess the performance of the feedback controller computed through the funnel synthesis method, a Monte Carlo simulation setting is used. A total of 100 different trajectories are randomly initialized, from a normal distribution, and simulated through a 4th order Runge Kunta integration method. All the initial conditions are guaranteed to actually be inside the funnel entry. Figures 4.8 to 4.19 show the evolution of each state variable for the 100 test cases (the vehicle's mass is omitted).

Hence, with the projection values in (4.4) and after observing the Figures below, one can argue that the initial funnel synthesis results are very promising. The performance of the feedback controller is noteworthy, by, without any sudden variations, making the actual trajectory converge to the reference for any initial condition inside the funnel entry.

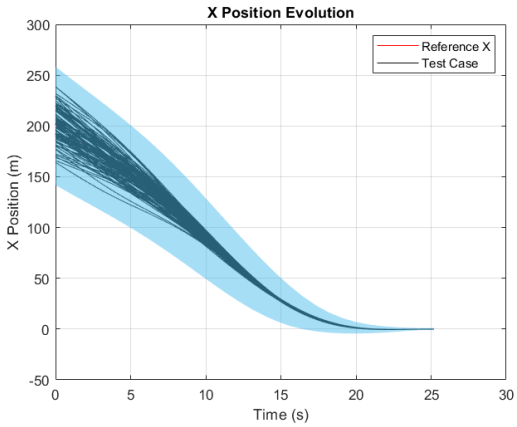

Figure 4.8: Evolution of the x-position

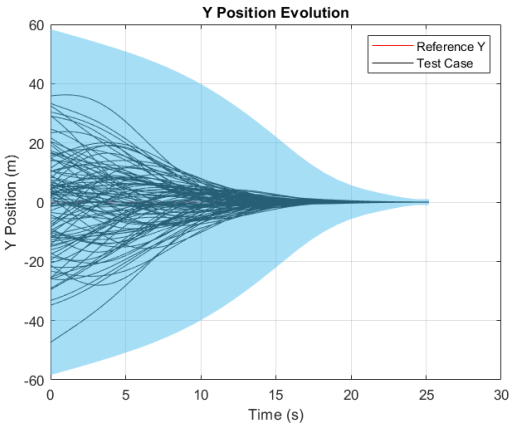

Figure 4.9: Evolution of the y-position

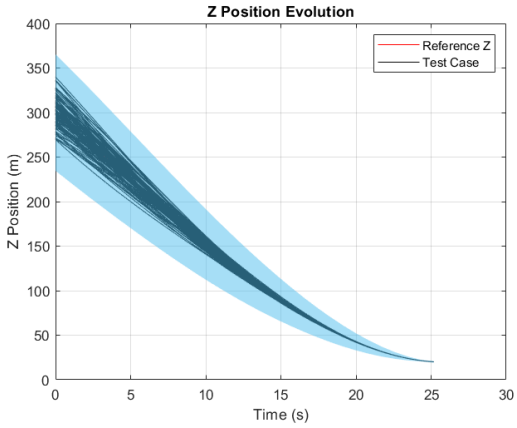

Figure 4.10: Evolution of the z-position

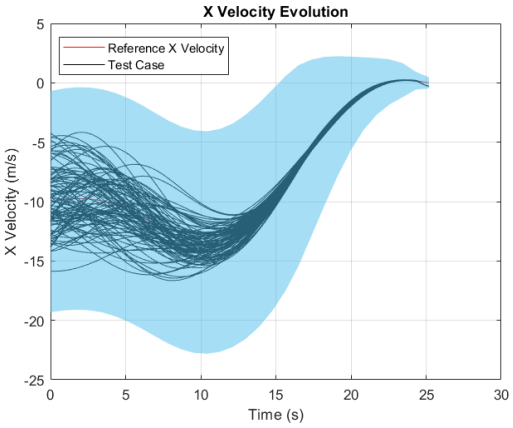

Figure 4.11: Evolution of the x-velocity

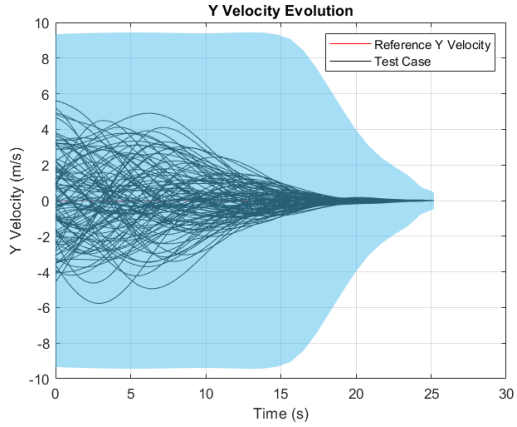

Figure 4.12: Evolution of the y-velocity

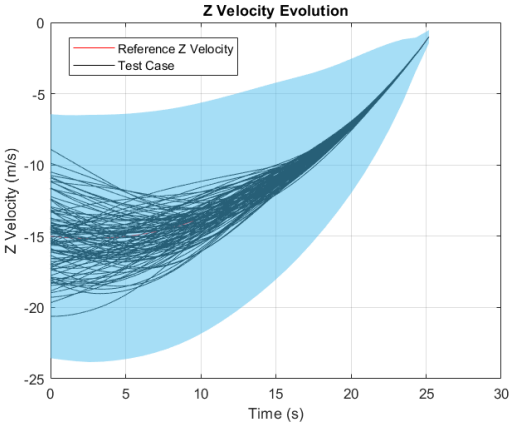

Figure 4.13: Evolution of the z-velocity

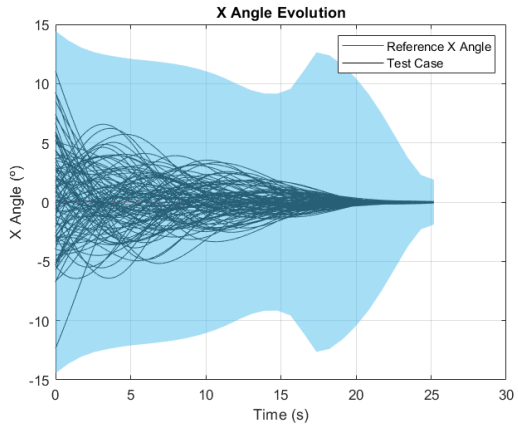

Figure 4.14: Evolution of the Euler angle relative to the x-axis

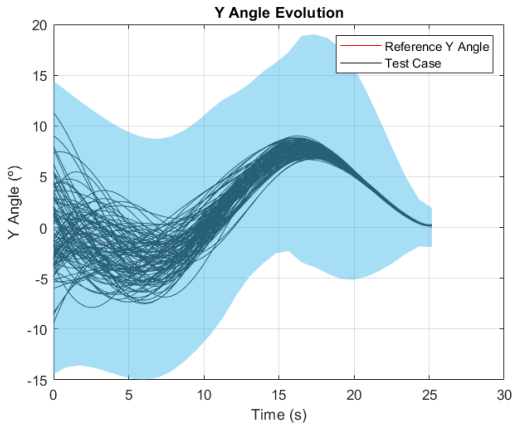

Figure 4.15: Evolution of the Euler angle relative to the y-axis

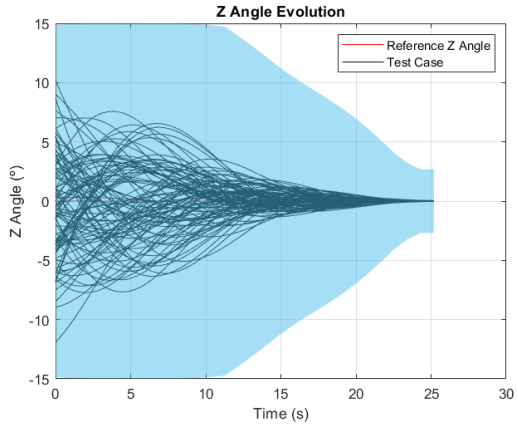

Figure 4.16: Evolution of the Euler angle relative to the z-axis

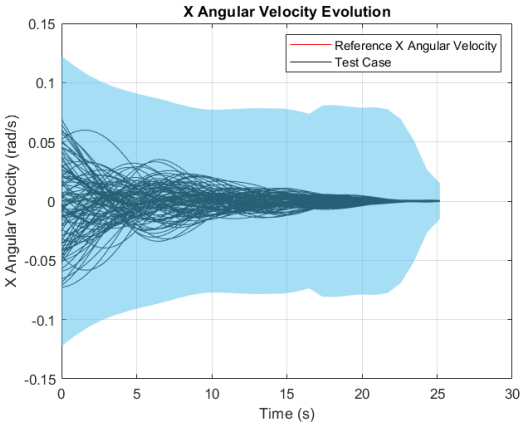

Figure 4.17: Evolution of the angular velocity relative to the x-axis

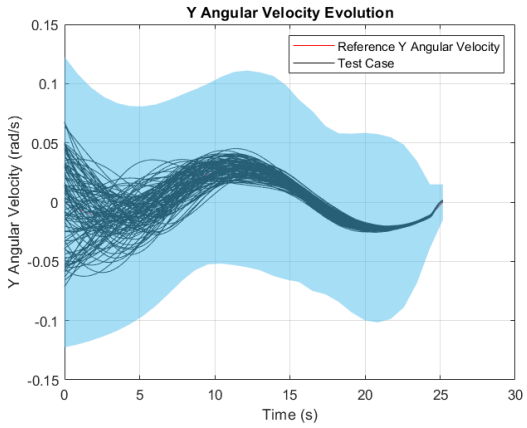

Figure 4.18: Evolution of the angular velocity relative to the y-axis

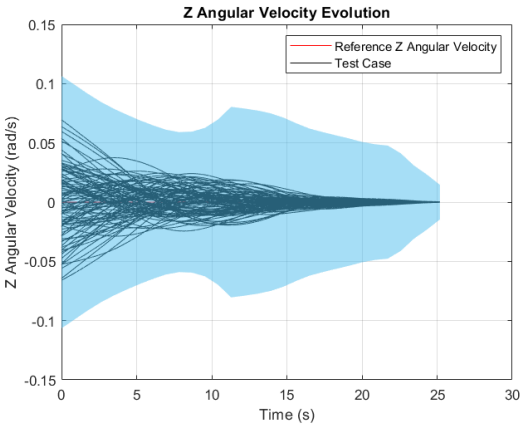

Figure 4.19: Evolution of the angular velocity relative to the z-axis

Additionally, apart from no constraints being violated, it is possible to show that the funnel is actually invariant by plotting, in Figure 4.20, the evolution of the Lyapunov function $V(t, x(t)) = (x(t) - \bar{x}(t))^T Q^{-1}(t)(x(t) - \bar{x}(t))$. One is able to conclude that, since the value of $V(t, x(t))$ strictly decreases

with time, the difference between the reference and the real system trajectories tends to zero and, consequently, the controller is stabilizing.

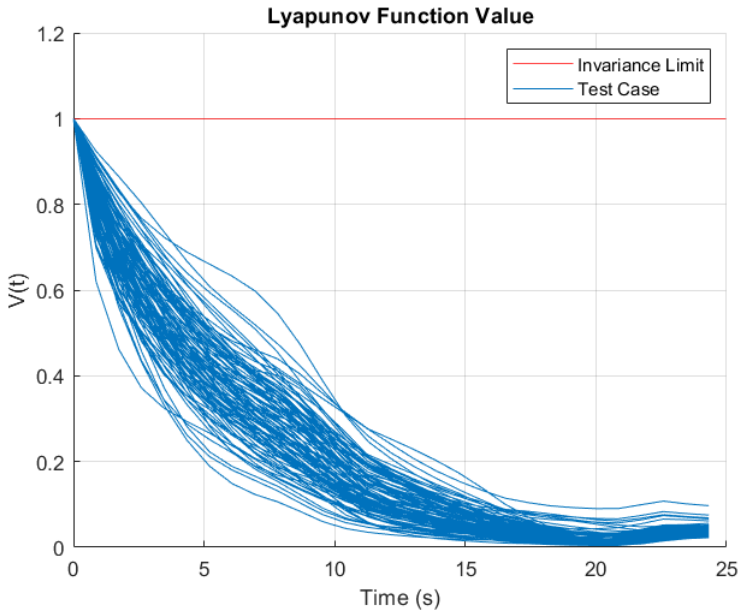

Figure 4.20: Evolution of the value of the Lyapunov function $V(t, \mathbf{x}(t))$

Hence, despite the challenges which inherently arise when working with a highly complex and nonlinear dynamical model, with a large number of state and control input dimensions, the quadratic funnel synthesis method shows a very promising performance. It is able to land the spacecraft safely and softly for a wide set of initial conditions, which stretches more than 116 m in any position direction, 20 m/s in any velocity component, 30 degrees in any Euler angle and 0.2 rad/s in any angular velocity component.

However, the reality is that, for the simulations shown above, there is a key element missing: disturbance and uncertainty effects acting directly on the system. For the initial quadratic funnel synthesis formulation, the goal was to, firstly, demonstrate a different robustness aspect of this algorithm: its ability to find a control law that mathematically guarantees that the system's trajectory converges to the reference in a short amount of time, for a large set of different initial conditions. Now, moving on to the actual research objectives of the thesis project, the performance of the funnel synthesis method should be evaluated when applied to a system where direct perturbations are considered in the simulation model.

For that purpose, a new dynamical model is then implemented, where random factors, which simulate the uncertainty and disturbances, affect the system's evolution. In a real landing sequence, three sources of external perturbations can be pointed out: model uncertainty, faulty sensor readings and unpredictable atmospheric phenomena.

The aforementioned factors can be combined into a simulation model where additive disturbances directly affect the system evolution. This is a common approach across the literature on robust methods [87, 89–91]. Consequently, now the continuous-time equations of motion with disturbances are given by:

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} -\alpha||\mathbf{F}_{\mathcal{B}}(t)||_2 - \beta + w_1 \\ \mathbf{v}_{\mathcal{I}}(t) + w_2 \\ \frac{1}{m(t)}\Big(C_{\mathcal{I}\leftarrow\mathcal{B}}(t)\Big(\mathbf{F}_{\mathcal{B}}(t) + \mathbf{A}_{\mathcal{B}}(t)\Big)\Big) + \mathbf{g}_{\mathcal{I}} + w_3 \\ \Omega(\mathbf{\Theta}_{\mathcal{B}}(t))\omega_{\mathcal{B}}(t) + w_4 \\ J_{\mathcal{B}}^{-1}(\tau_{\mathcal{B}}(t) + \mathbf{r}_{T,\mathcal{B}} \times \mathbf{F}_{\mathcal{B}}(t) + \mathbf{r}_{cp,\mathcal{B}} \times \mathbf{A}_{\mathcal{B}}(t) - \omega_{\mathcal{B}}(t) \times J_{\mathcal{B}}\omega_{\mathcal{B}}(t)) + w_5 \end{bmatrix} \tag{4.5}$$

where $w_1 \in \mathbb{R}$ and $w_2, w_3, w_4, w_5 \in \mathbb{R}^3$. All disturbance values change from temporal node to temporal node, and satisfy the following conditions: $||w_i||_\infty \leq 0.1,\ i = 1, 2, 3$ and $||w_i||_\infty \leq 0.01,\ i = 4, 5$. The angle and angular velocity disturbance bound is smaller because a deviation of 0.1 rad, which equals to 5.72 degrees, has a much bigger impact on the system than a deviation of 0.1 m in position or 0.1 m/s in velocity.

Figures 4.21 to 4.32 show the perturbed system's evolution, when controlled through the feedback controller designed by the funnel synthesis method. A Monte Carlo simulation setting, with 20 randomly initialized trajectories, is used. The results are not as promising as before. The controller shows a poorer performance, as the disturbances make the system's trajectories oscillate a lot around the reference. They behave in an erratic and undesired way, which is especially noticeable in the plots relative to the Euler angles and angular velocities. Additionally, even though the system approaches the goal state, it is never capable of precisely reaching it.

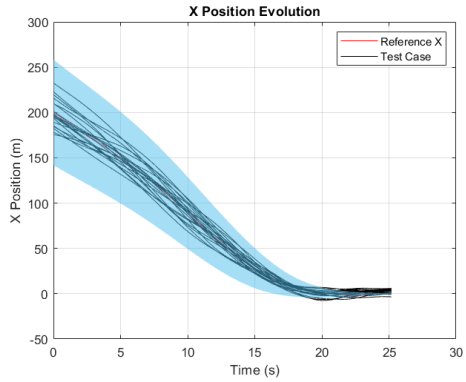

Figure 4.21: Evolution of the x-position

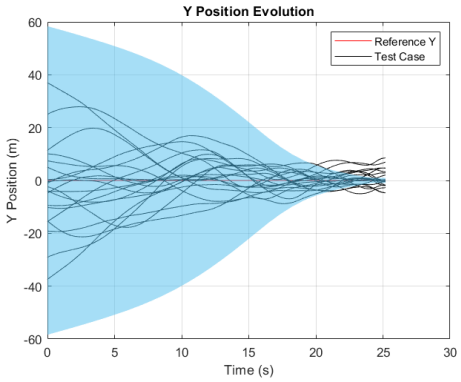

Figure 4.22: Evolution of the y-position

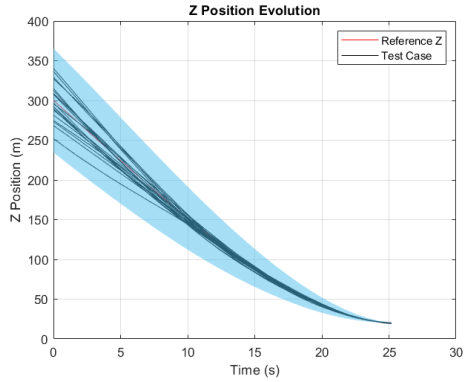

Figure 4.23: Evolution of the z-position

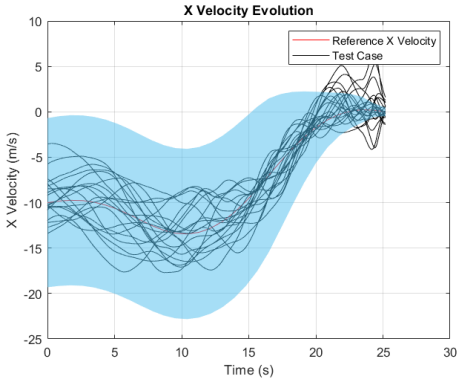

Figure 4.24: Evolution of the x-velocity

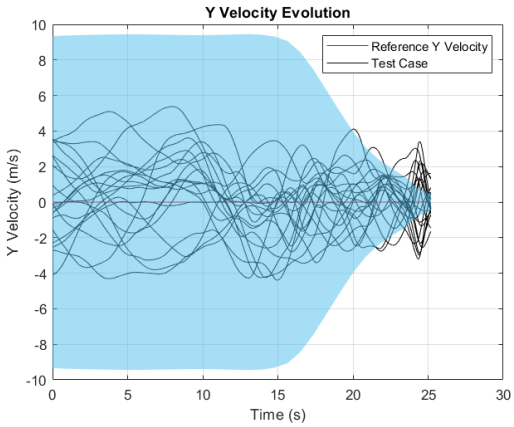

Figure 4.25: Evolution of the y-velocity

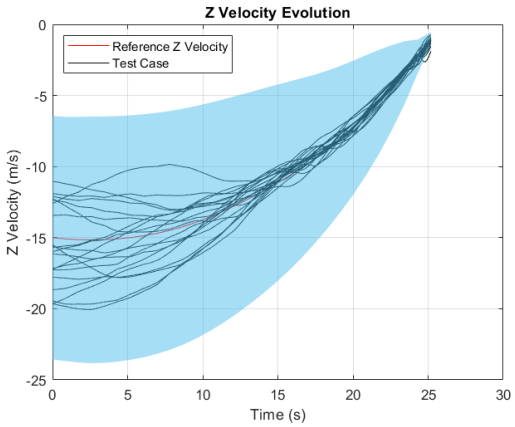

Figure 4.26: Evolution of the z-velocity

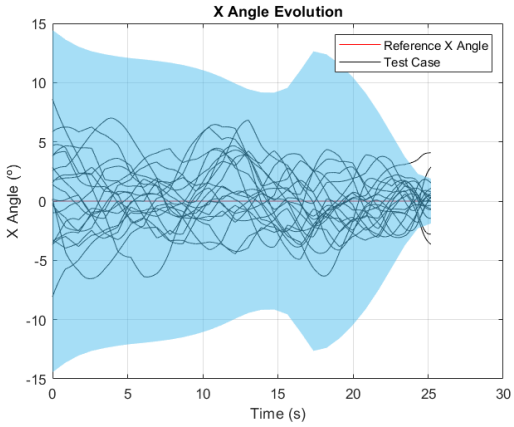

Figure 4.27: Evolution of the Euler angle relative to the x-axis

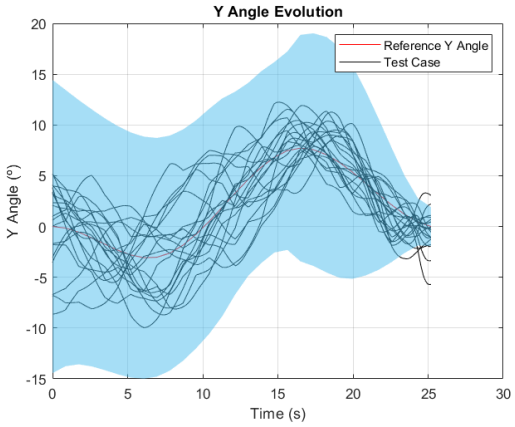

Figure 4.28: Evolution of the Euler angle relative to the y-axis

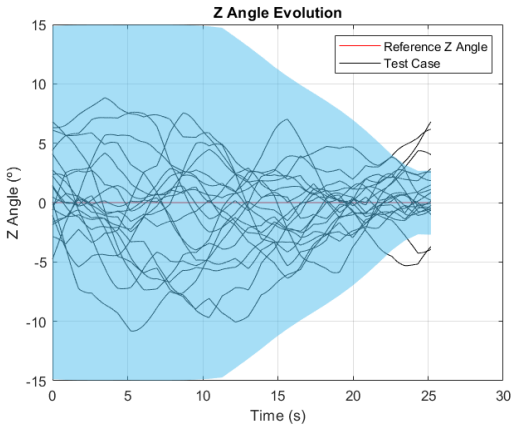

Figure 4.29: Evolution of the Euler angle relative to the z-axis

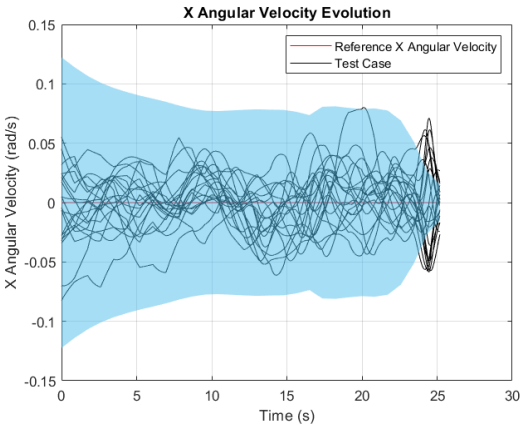

Figure 4.30: Evolution of the angular velocity relative to the x-axis

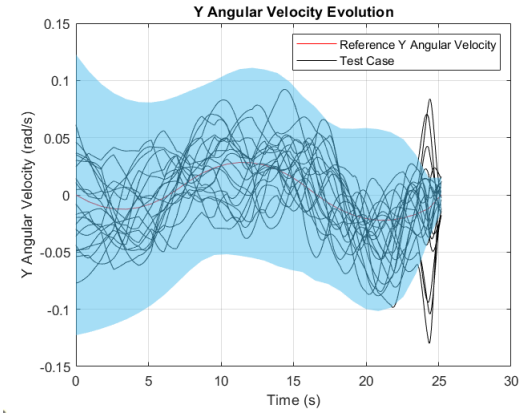

Figure 4.31: Evolution of the angular velocity relative to the y-axis

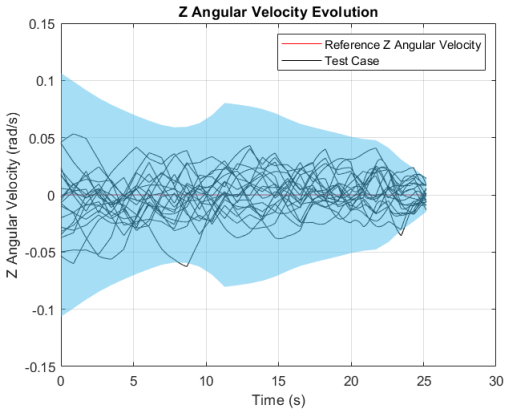

Figure 4.32: Evolution of the angular velocity relative to the z-axis

It is possible to confirm that the funnel is not invariant anymore by plotting the evolution of the Lyapunov function $V(t, x(t))$. Opposed to the initial scenario, no trajectory in this case stays below the invariance limit, which corresponds to $V(t, x(t)) = 1$.

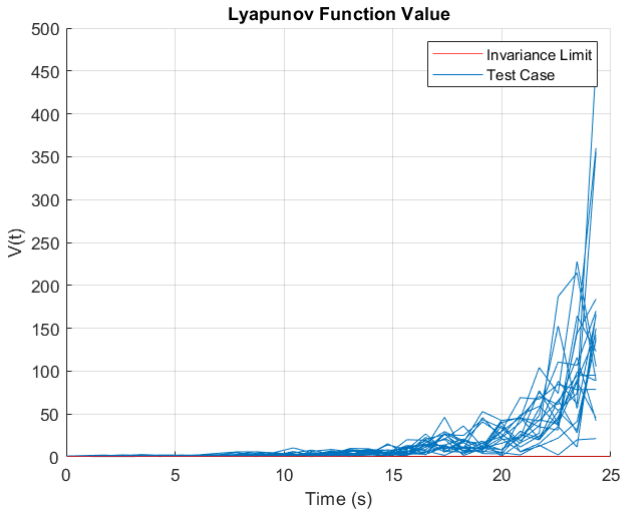

Figure 4.33: Evolution of the value of the Lyapunov function $V(t, \mathbf{x}(t))$

So, from the above results, one can establish that, when applied to a system without direct effect of disturbances, a 6-DOF PDG control algorithm can, to a great extent, improve its robustness by considering a funnel synthesis approach. Trajectories starting from a wide set of different initial conditions quickly converge to the reference, and stabilizability is mathematically guaranteed. However, when additive disturbances are added to the simulation model, the performance of the feedback controller is poor, the trajectories have an undesirable behavior and there is a very high probability that the system's trajectory will quickly leave the funnel. For that reason, alternative solutions should be explored.

4.2.2 Balancing the Funnel Size

Across the literature, it is often stated that minimizing the funnel size can help decrease the undesired effects of external perturbations on the system [7, 89]. In this thesis project, the goal is to evaluate the

impact of this modification and understand if it leads to a major performance improvement.

As mentioned before, to achieve this objective, one simply needs to reformulate the cost function and add a single constraint, as following:

$$\begin{aligned} J &= -w_{Q_0} \ \log\det(Q_1) + \sum_{i=2}^{n_M} w_Q \ \lambda_i^Q \\ \text{subject to } & Q_i \leq \lambda_i^Q I \qquad \forall i = 2, \ldots, n_M \end{aligned} \tag{4.6}$$

where w_Q is a penalty weight and, in the following case, is initialized as 15.

Figures 4.34 to 4.45 show how minimizing the funnel size affects the results in a scenario where the dynamical model does not include any direct uncertainty or disturbance effects yet. Only 2 test cases are considered, and the corresponding initial conditions are randomly generated. For each one of the test cases, both controllers are used to determine the control actions, which allows to better compare their performance. Trajectory 1 corresponds to the previous formulation, while trajectory 2 results from minimizing the funnel size. Additionally, the projections of both funnels into each state dimension are visually represented. After a first analysis, one can observe that the trajectories from the two different scenarios evolve in a very similar way. When there is no direct influence of disturbances on the equations of motion, both controllers show a remarkable performance, making the trajectory quickly converge to the reference.

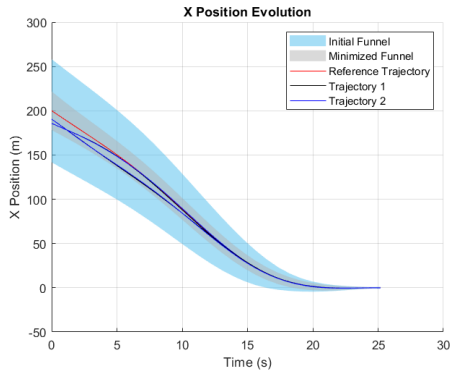

Figure 4.34: Evolution of the x-position

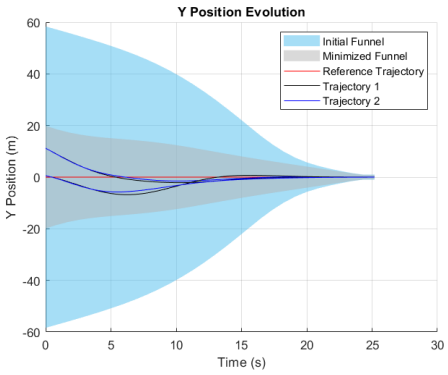

Figure 4.35: Evolution of the y-position

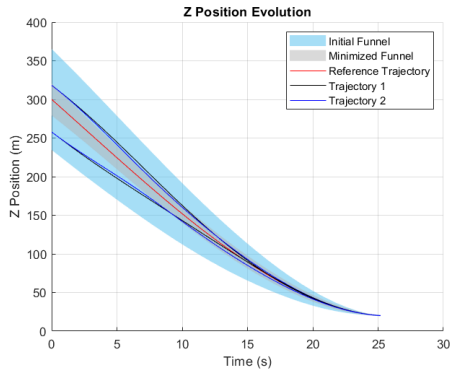

Figure 4.36: Evolution of the z-position

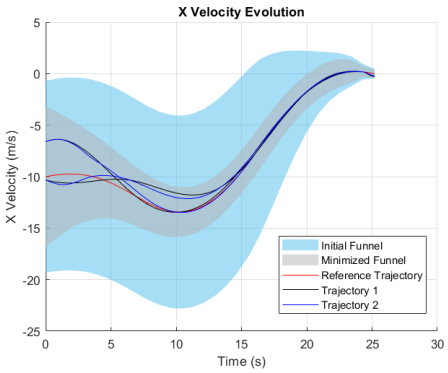

Figure 4.37: Evolution of the x-velocity

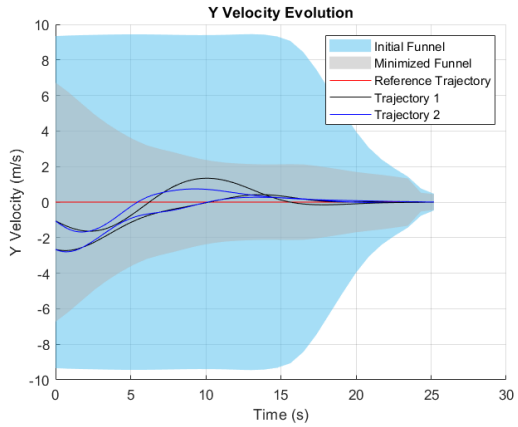

Figure 4.38: Evolution of the y-velocity

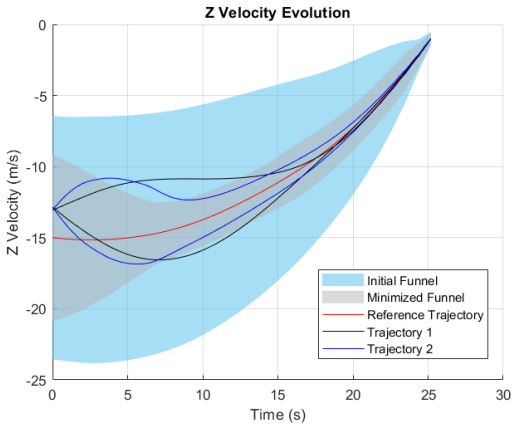

Figure 4.39: Evolution of the z-velocity

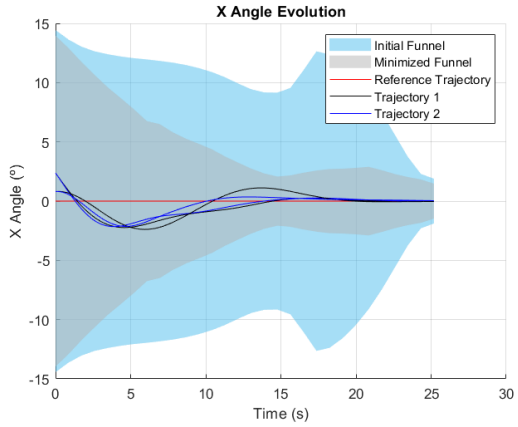

Figure 4.40: Evolution of the Euler angle relative to the x-axis

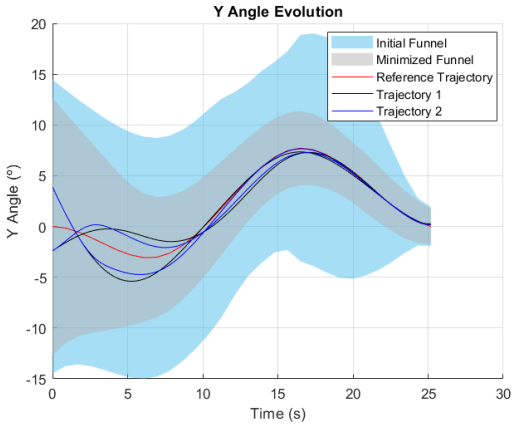

Figure 4.41: Evolution of the Euler angle relative to the y-axis

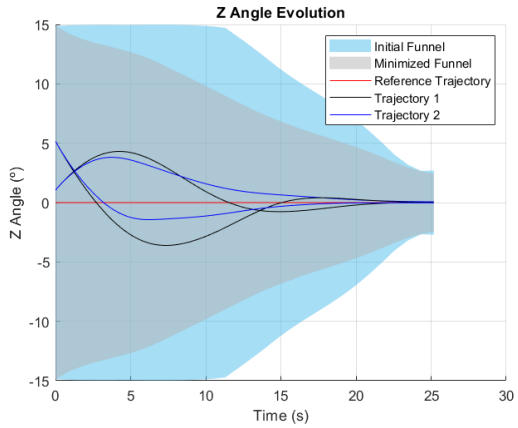

Figure 4.42: Evolution of the Euler angle relative to the z-axis

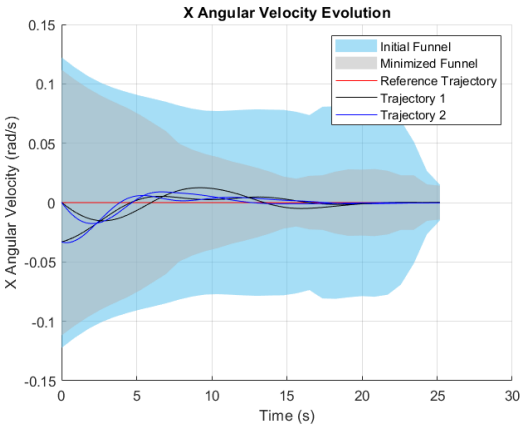

Figure 4.43: Evolution of the angular velocity relative to the x-axis

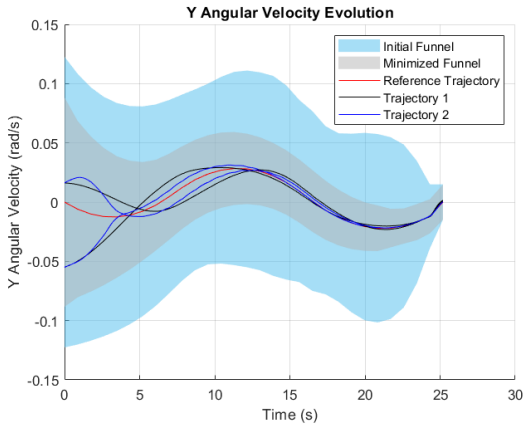

Figure 4.44: Evolution of the angular velocity relative to the y-axis

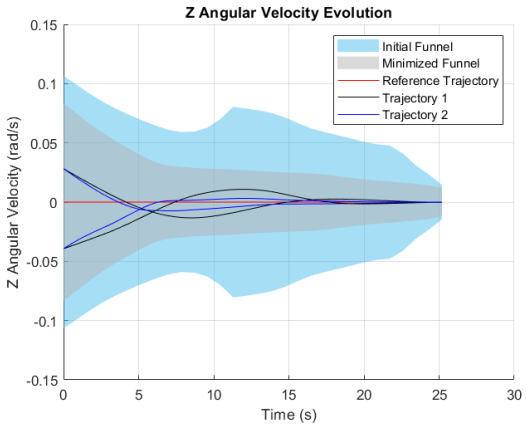

Figure 4.45: Evolution of the angular velocity relative to the z-axis

However, more specific conclusions can be drawn by observing the evolution of the Lyapunov function $V(t, x(t))$, plotted in Figure 4.46. In this case, 15 randomly initialized trajectories are represented. Even though it is possible to confirm that both funnels are invariant, the corresponding control actions are more conservative when the funnel is minimized and, consequently, in general, the system's trajectory evolves closer to the reference than in the original case.

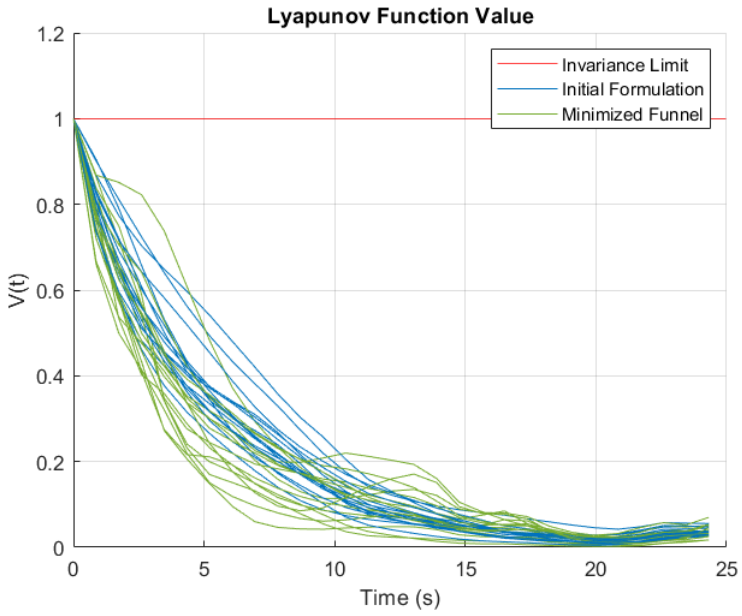

Figure 4.46: Evolution of the value of the Lyapunov function $V(t, \mathbf{x}(t))$

Despite this, to truly assess the results of the funnel minimization formulation, one should test it in a simulation model where additive disturbances act directly on the system, similar to what has been done from Figures 4.21 to 4.32. Figures 4.47 to 4.58 display the aforementioned scenario and allow to compare the performance of the minimized funnel formulation with the initial formulation. Even though, overall, the more conservative feedback controller makes the trajectory evolve a little closer to the reference, its performance is still not reasonable. The state variables oscillate too much and the landing

sequence is not as smooth and soft as it should be. Regardless of the initial condition, it is highly likely that the trajectories will leave the funnel in the last seconds of the simulation.

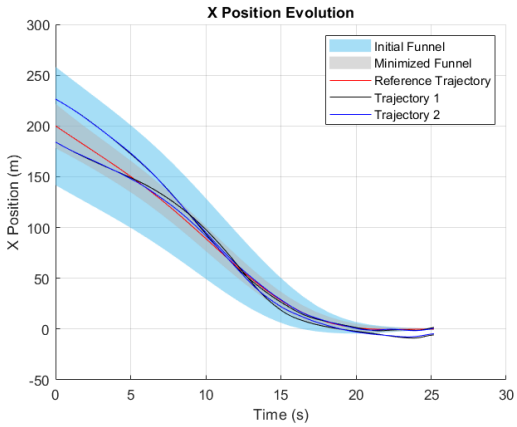

Figure 4.47: Evolution of the x-position

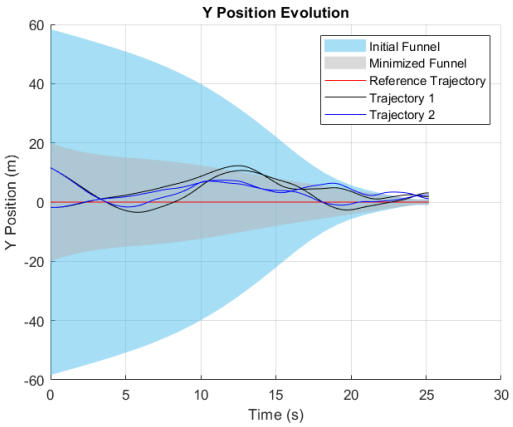

Figure 4.48: Evolution of the y-position

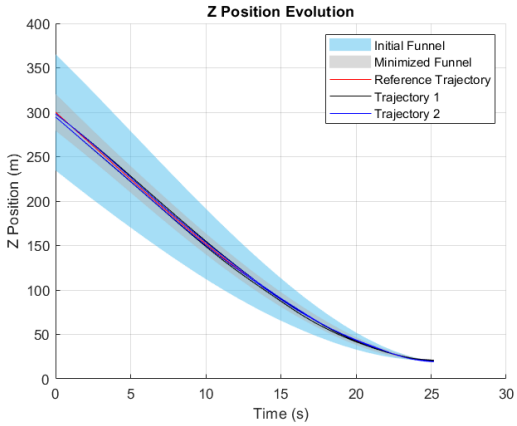

Figure 4.49: Evolution of the z-position

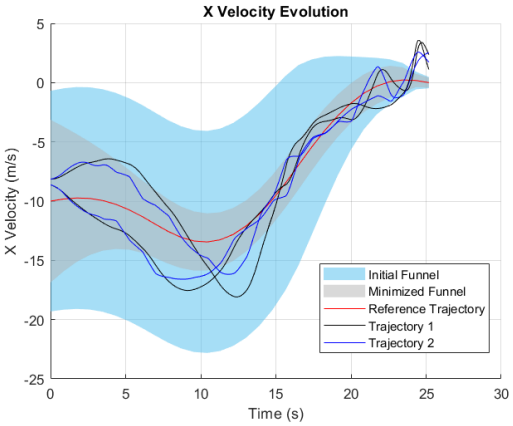

Figure 4.50: Evolution of the x-velocity

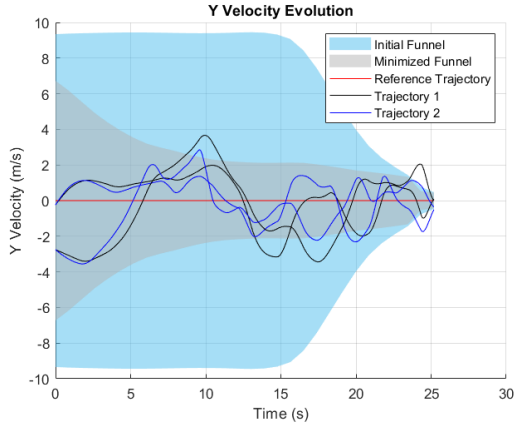

Figure 4.51: Evolution of the y-velocity

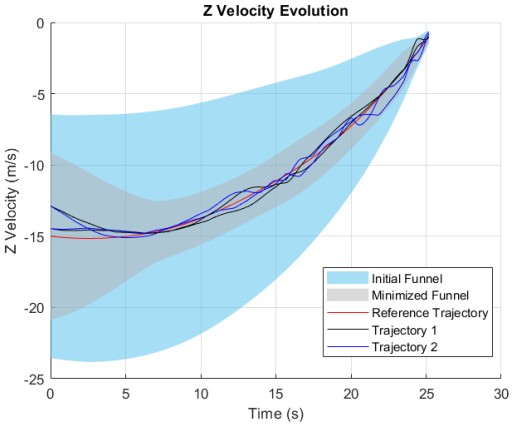

Figure 4.52: Evolution of the z-velocity

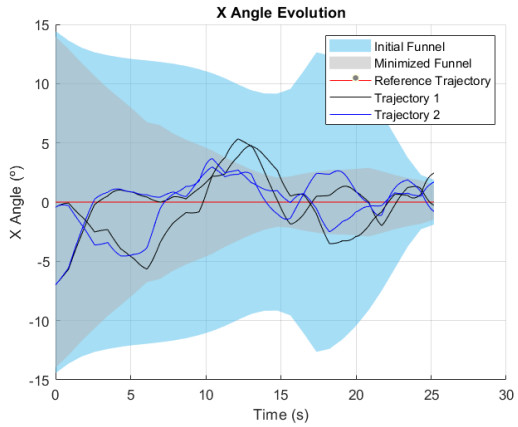

Figure 4.53: Evolution of the Euler angle relative to the x-axis

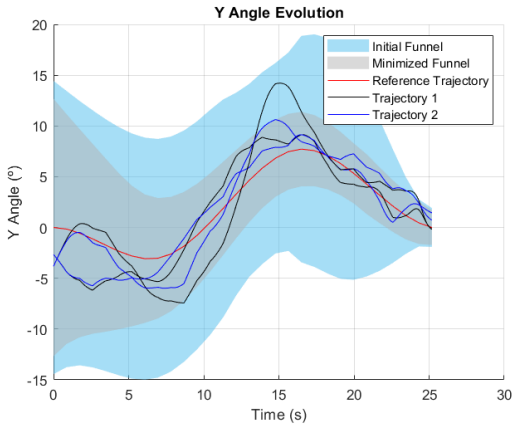

Figure 4.54: Evolution of the Euler angle relative to the y-axis

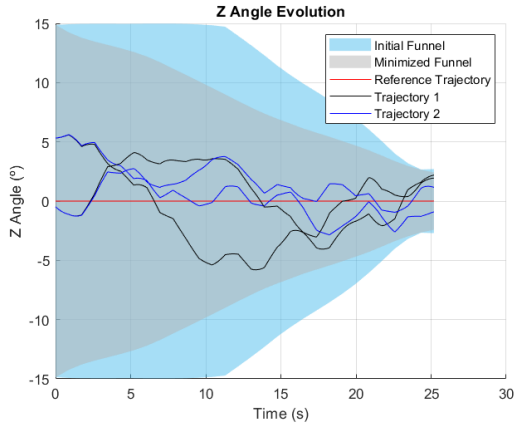

Figure 4.55: Evolution of the Euler angle relative to the z-axis

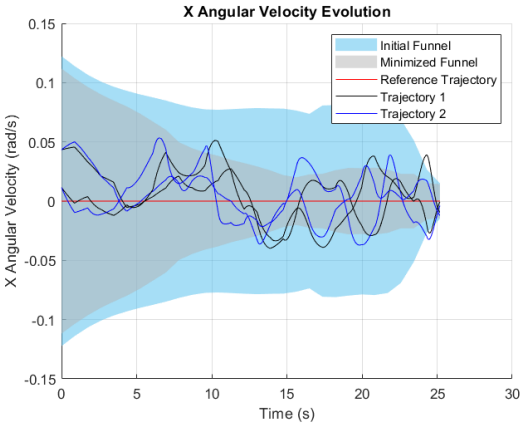

Figure 4.56: Evolution of the angular velocity relative to the x-axis

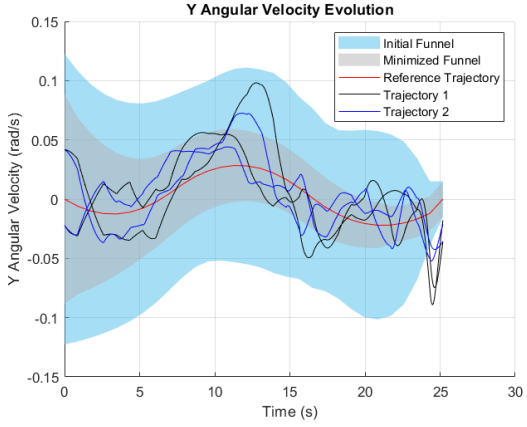

Figure 4.57: Evolution of the angular velocity relative to the y-axis

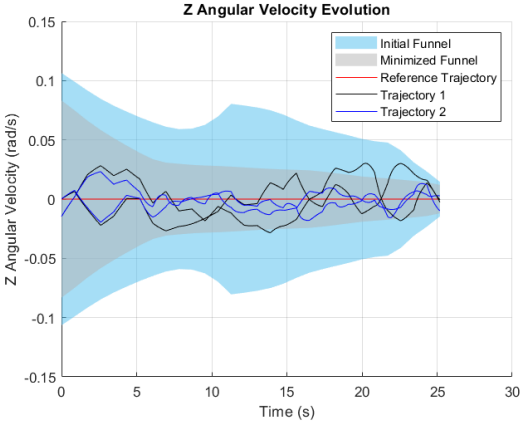

Figure 4.58: Evolution of the angular velocity relative to the z-axis

Hence, it is straightforward to conclude that adding the funnel minimization condition is not enough to guarantee that the system can successfully mitigate uncertainty and external disturbances. It makes the funnel and the feedback controller more conservative, which, consequently, translate in trajectories

evolving closer to the reference. Nevertheless, this difference is not as significant as expected, and the landing sequence remains rough and inaccurate when additive disturbances are applied to the system. Therefore, a different control strategy needs to be exploited.

4.2.3 Funnel Formulation with Disturbances

The next step is to understand how formulating the funnel synthesis optimization problem with a dynamical model that includes the effect of disturbances impacts the ability of the feedback controller to reject these unexpected factors. As explained in Section 3.5.4, the main difference in this new formulation is that the invariance condition (3.75) is substituted by (3.86). The remaining constraints remain the same.

For the results below, the decay rate α is set to $\alpha = 0.05$, analogous to the previous cases. The constant λ_w is initialized as $\lambda_w = 0.01$. The funnel and the feedback controller are computed in a single iteration, and the Lipschitz constants are given the values estimated by the initial funnel synthesis method.

For the simulations above, where additive disturbances are considered in the dynamical model, the following conditions are satisfied: $||w(t)||_\infty \leq 0.1,\ \forall t \in [t_0, t_f]$ for the disturbances that act on the mass, position and velocity variables, and $||w(t)||_\infty \leq 0.01,,\ \forall t \in [t_0, t_f]$ for the disturbances that affect the rotational motion of the spacecraft. The problem here is that the new funnel formulation often results in an infeasible optimization problem, since it is not possible to compute a robust control invariant set and a stabilizing feedback controller for such high disturbance values. Only when the disturbance bounds are decreased to 0.01 and 0.001, respectively, reasonable results are finally obtained.

From Figures 4.59 to 4.70, the evolution of the different state variables for 10 randomly initialized trajectories can be observed, where the disturbances' magnitudes respect the maximum values specified above. In this scenario, the robust control invariant funnel and the corresponding feedback controller show a favourable performance, as the trajectories seem to definitely converge to the reference and the effect of the disturbances is almost null.

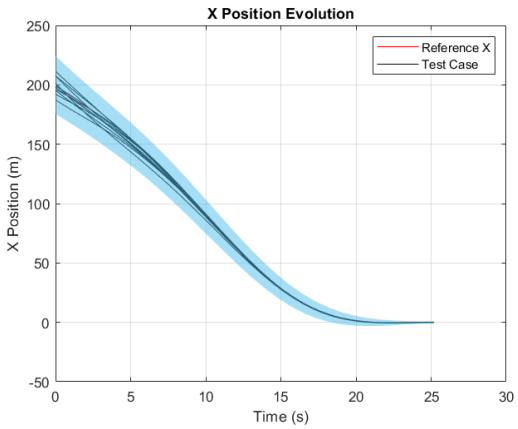

Figure 4.59: Evolution of the x-position

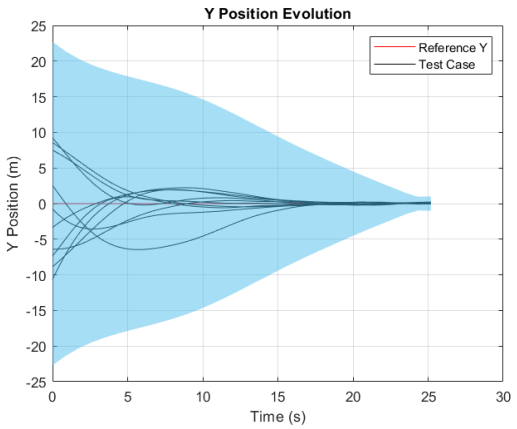

Figure 4.60: Evolution of the y-position

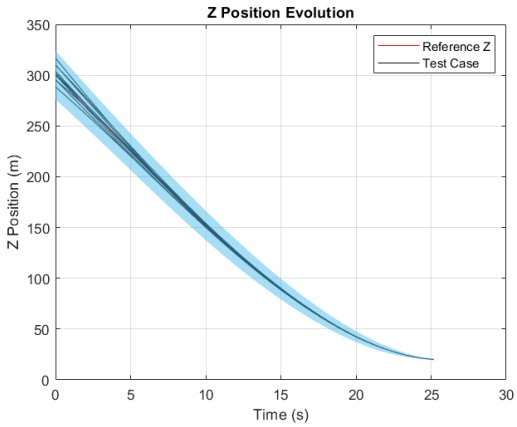

Figure 4.61: Evolution of the z-position

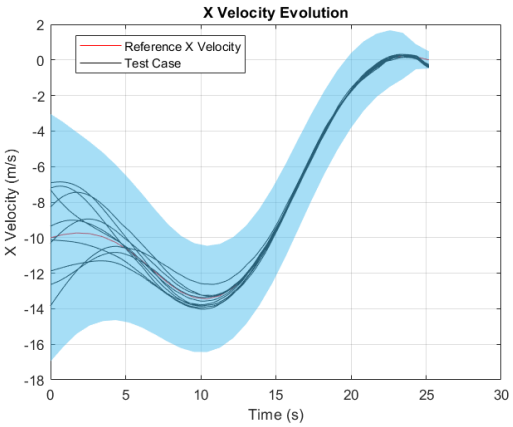

Figure 4.62: Evolution of the x-velocity

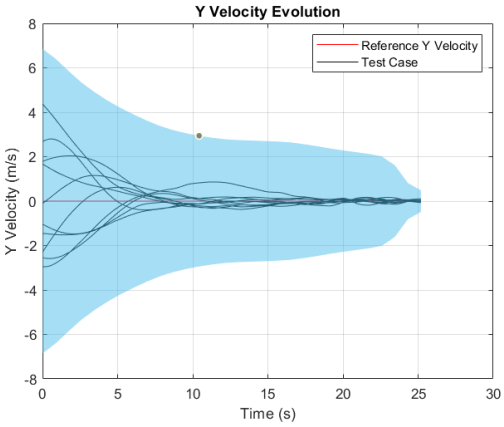

Figure 4.63: Evolution of the y-velocity

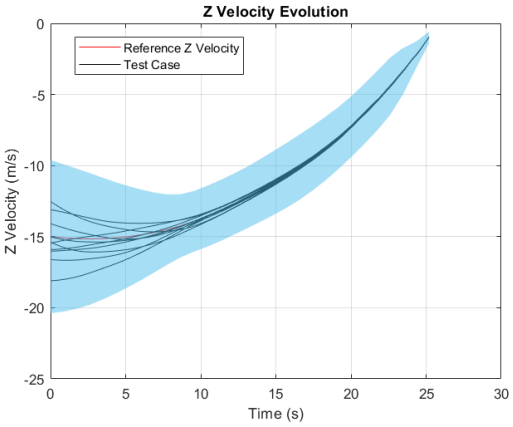

Figure 4.64: Evolution of the z-velocity

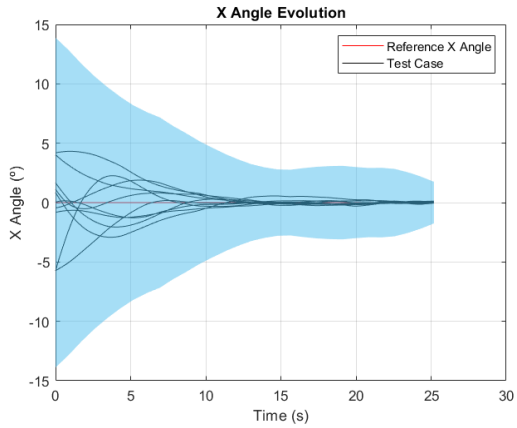

Figure 4.65: Evolution of the Euler angle relative to the x-axis

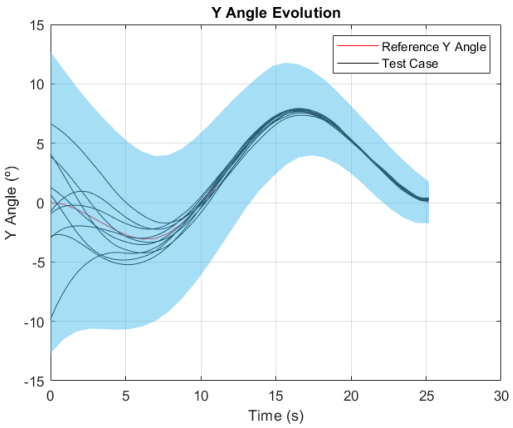

Figure 4.66: Evolution of the Euler angle relative to the y-axis

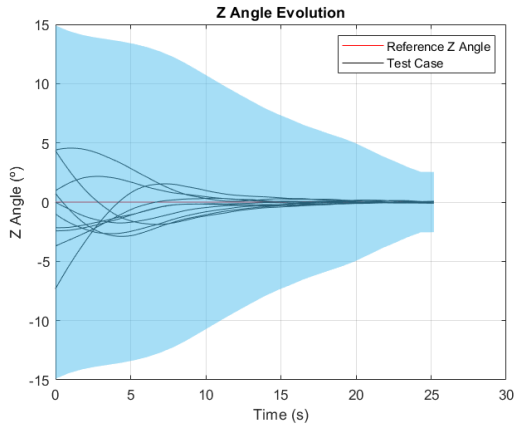

Figure 4.67: Evolution of the Euler angle relative to the z-axis

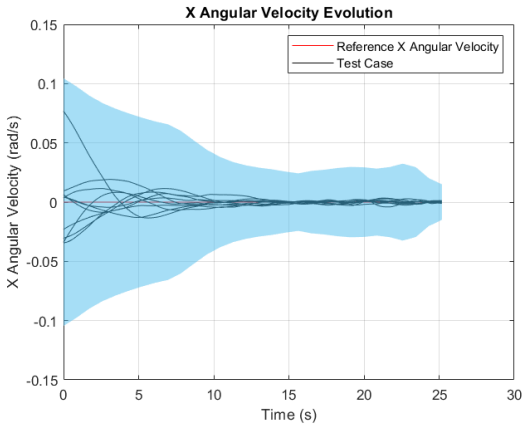

Figure 4.68: Evolution of the angular velocity relative to the x-axis

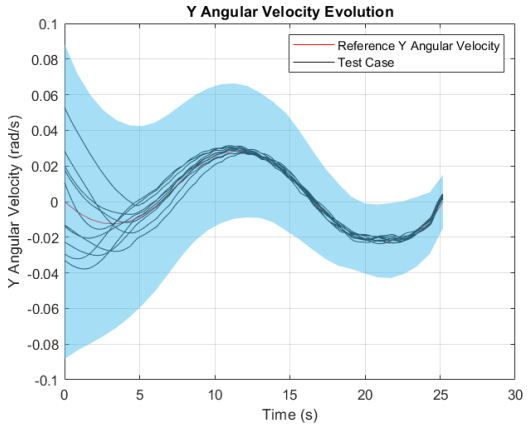

Figure 4.69: Evolution of the angular velocity relative to the y-axis

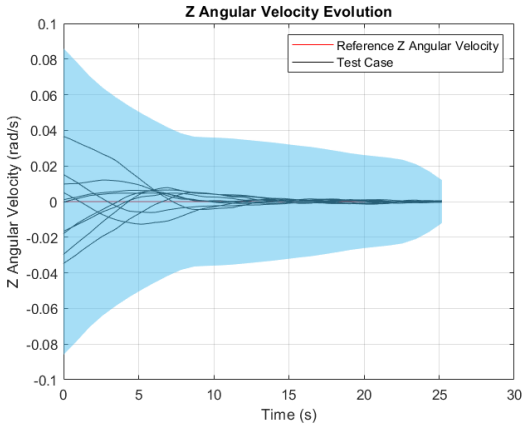

Figure 4.70: Evolution of the angular velocity relative to the z-axis

Nonetheless, if the upper bounds of the disturbances applied in the simulation model are increased to 0.1 and 0.01, similarly to the cases studied in the previous Sections, the results are not so satisfactory. In Figures 4.71 to 4.82, a comparison between the performance of the newly computed feedback controller and the initial controller can be seen. Trajectory 1 corresponds to the latter scenario, while trajectory 2 to the former. Even though the feedback controller associated with the new formulation makes the trajectory evolve again closer to the reference, the effects of disturbances is still very visible. The state variables, especially the Euler angles and the angular velocities, experience too sudden variations, which contributes to a rougher landing procedure. Additionally, in general, this control approach still leads to inaccuracies, as the final state is not as close to the goal state as pretended.

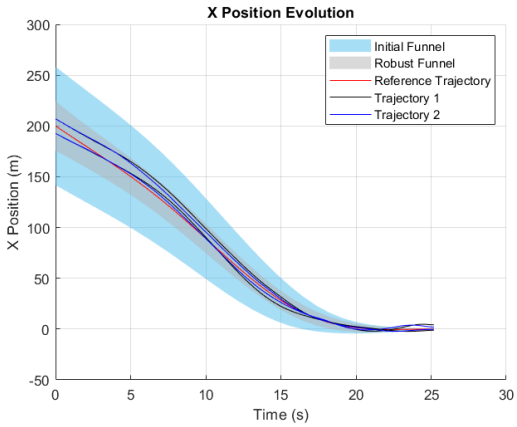

Figure 4.71: Evolution of the x-position

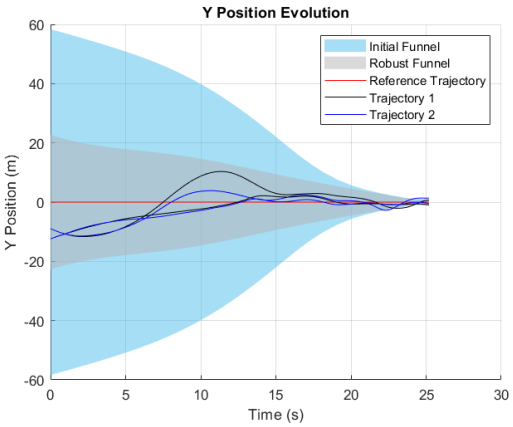

Figure 4.72: Evolution of the y-position

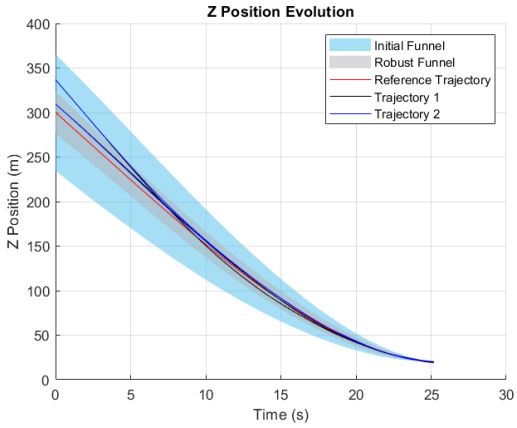

Figure 4.73: Evolution of the z-position

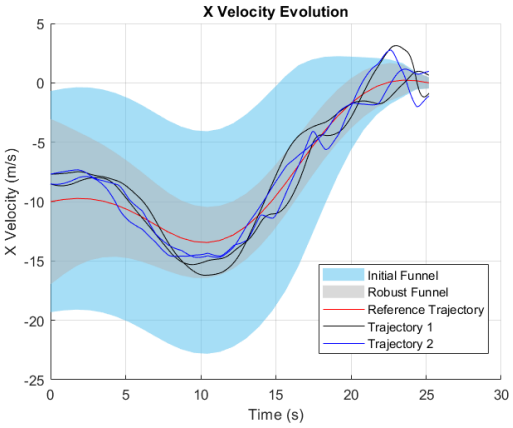

Figure 4.74: Evolution of the x-velocity

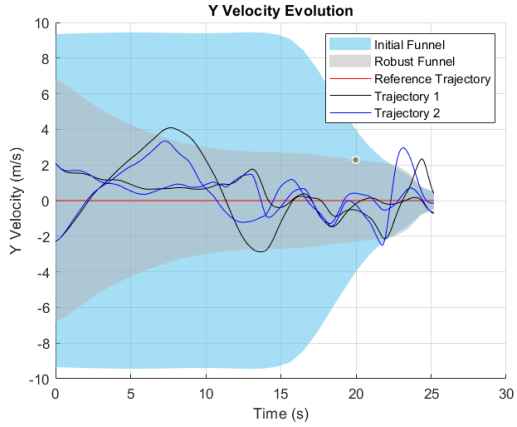

Figure 4.75: Evolution of the y-velocity

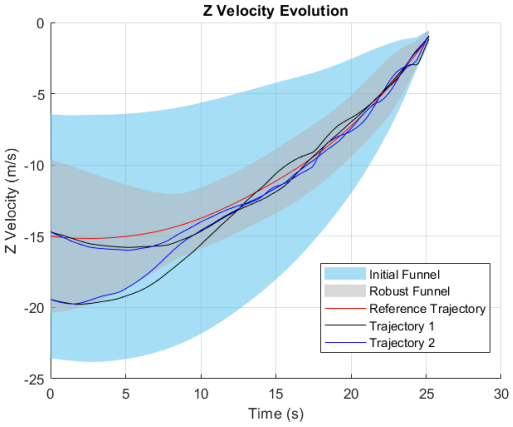

Figure 4.76: Evolution of the z-velocity

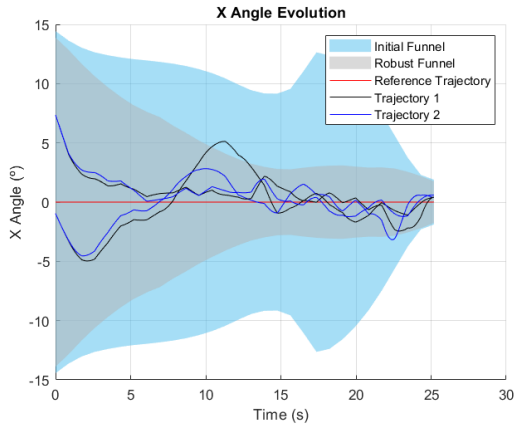

Figure 4.77: Evolution of the Euler angle relative to the x-axis

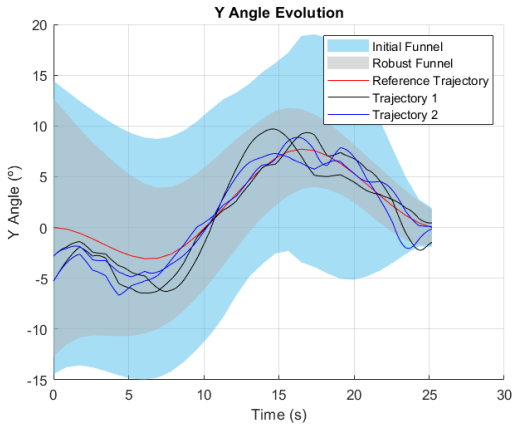

Figure 4.78: Evolution of the Euler angle relative to the y-axis

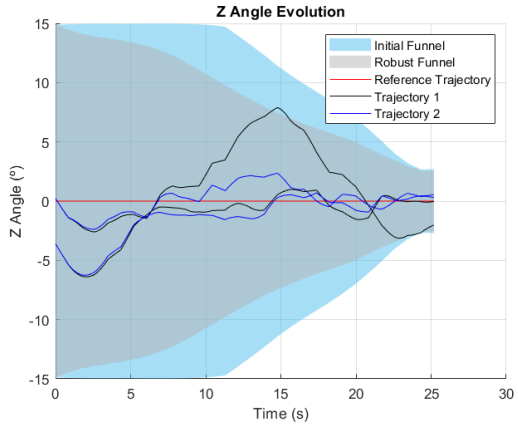

Figure 4.79: Evolution of the Euler angle relative to the z-axis

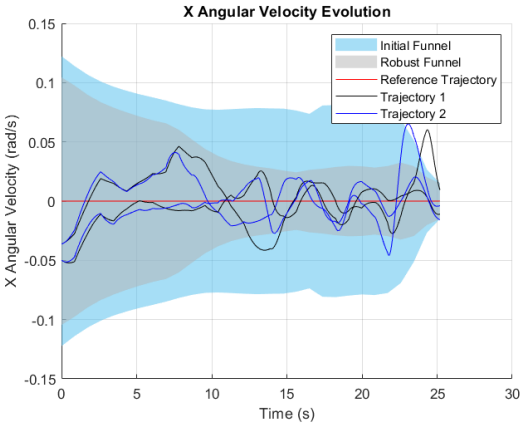

Figure 4.80: Evolution of the angular velocity relative to the x-axis

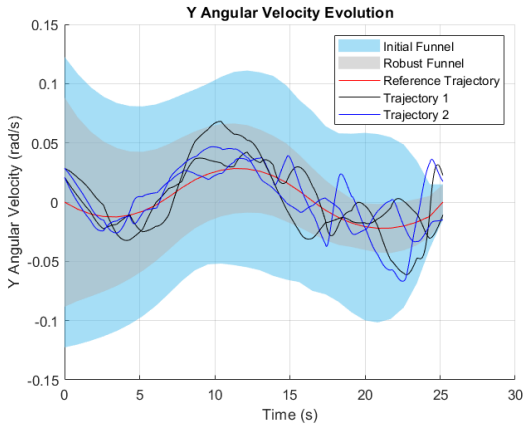

Figure 4.81: Evolution of the angular velocity relative to the y-axis

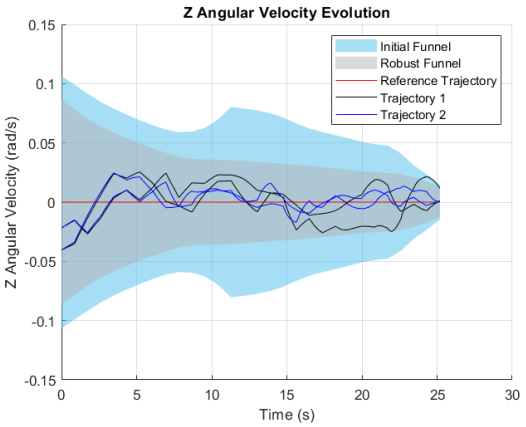

Figure 4.82: Evolution of the angular velocity relative to the z-axis

Accordingly to these results, one can infer that minimizing the funnel size or formulating the funnel synthesis optimization problem with a disturbed dynamical model contributes to an improvement in

performance. The trajectories evolve closer to the reference and the degree of accuracy increases. Nevertheless, this improvement is not as significant as one would expect, as the effects of the disturbances are still clear and persistent. For that reason, the importance of using real-time information regarding the trajectory evolution to further reject external perturbations is becoming more and more comprehensible.

4.3 MPC with Funnel Parameters Strategy

For the results of the proposed funnel Model Predictive Control (MPC) strategy, a total of 50 temporal nodes are used. Since the landing procedure takes around 25 seconds, that translates to a sampling time of 0.5 seconds. The control horizon N is set to $N = 4$ and the initial condition is randomly initialized from inside the funnel. One can observe the evolution of the system state variables from Figures 4.83 to 4.94. From the analysis of such evolution, it is possible to conclude that the proposed MPC strategy can accomplish a safe, smooth and accurate landing. The influence of the disturbances is minimized, the system's trajectory evolves close to the reference and the final state matches almost perfectly the goal state. Even though some small oscillations are still noticeable, especially in the state variables regarding the rotational motion, the effects are minor compared to all the scenarios previously shown.

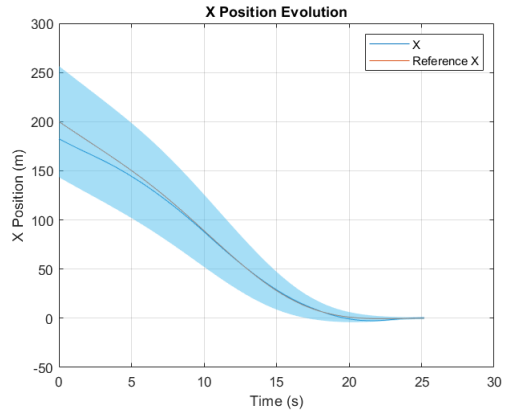

Figure 4.83: Evolution of the x-position

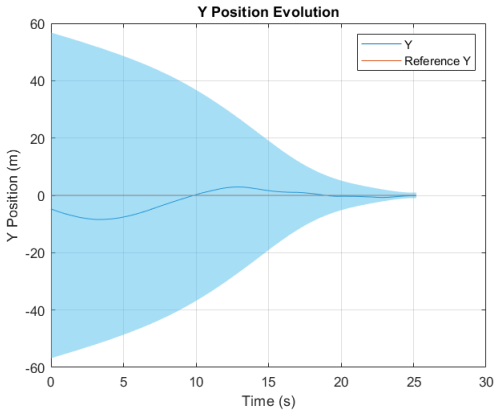

Figure 4.84: Evolution of the y-position

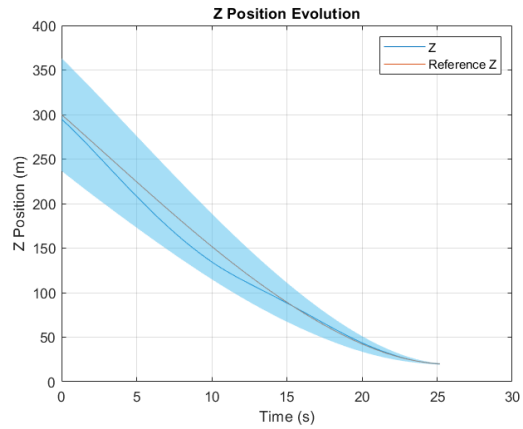

Figure 4.85: Evolution of the z-position

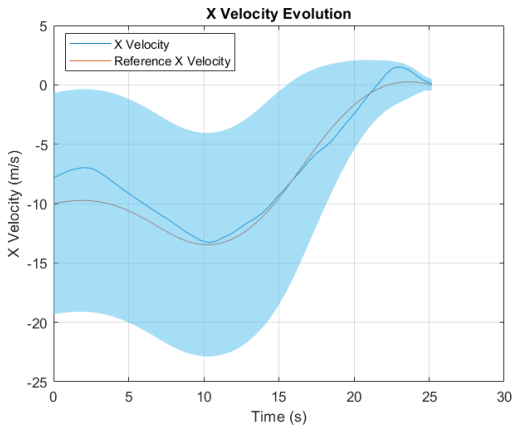

Figure 4.86: Evolution of the x-velocity

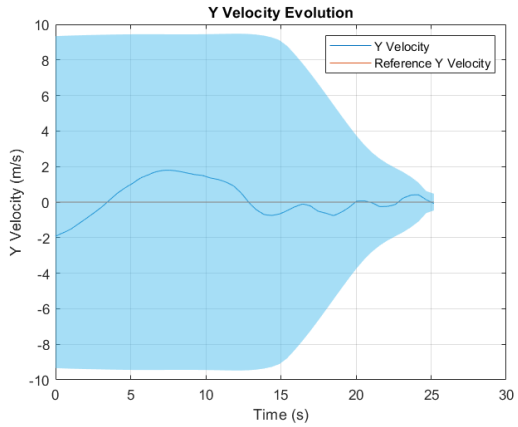

Figure 4.87: Evolution of the y-velocity

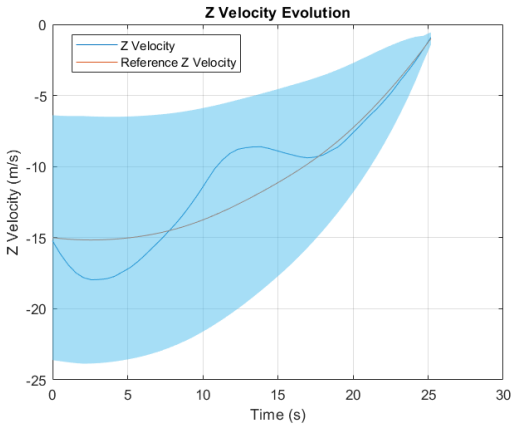

Figure 4.88: Evolution of the z-velocity

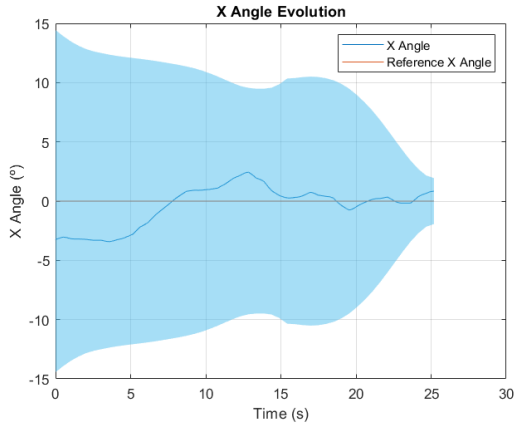

Figure 4.89: Evolution of the Euler angle relative to the x-axis

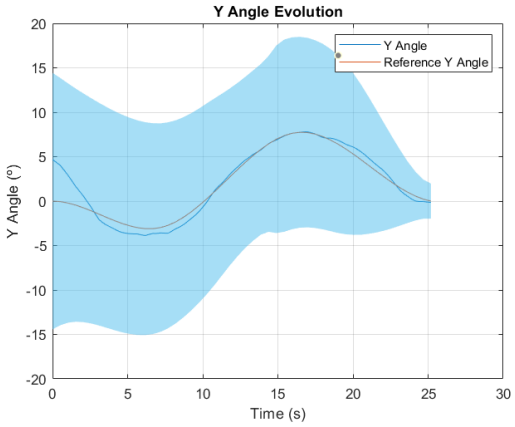

Figure 4.90: Evolution of the Euler angle relative to the y-axis

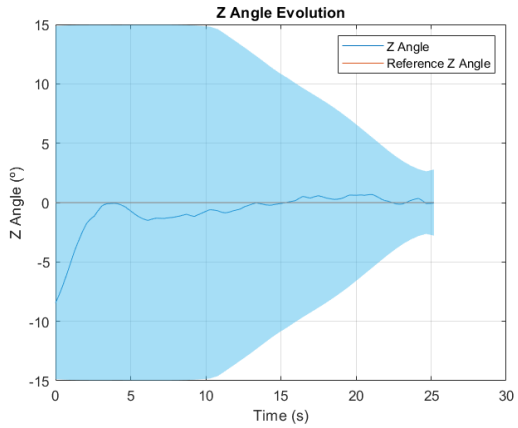

Figure 4.91: Evolution of the Euler angle relative to the z-axis

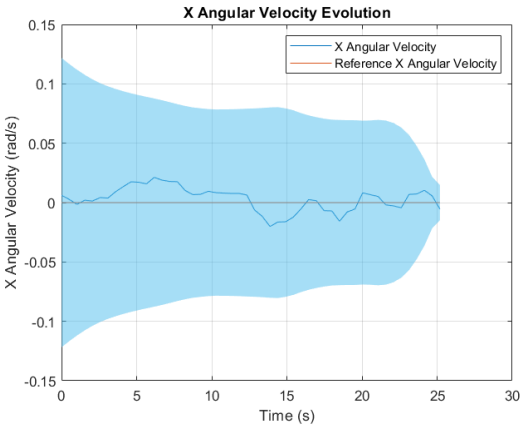

Figure 4.92: Evolution of the angular velocity relative to the x-axis

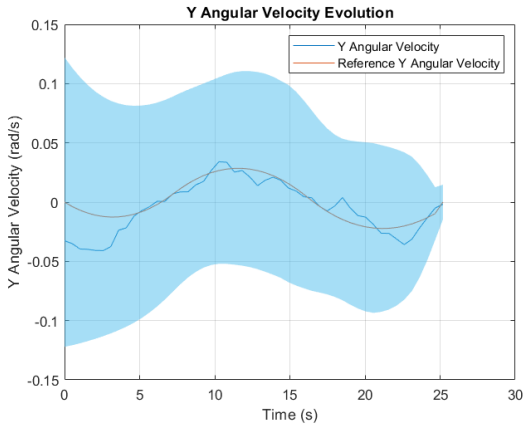

Figure 4.93: Evolution of the angular velocity relative to the y-axis

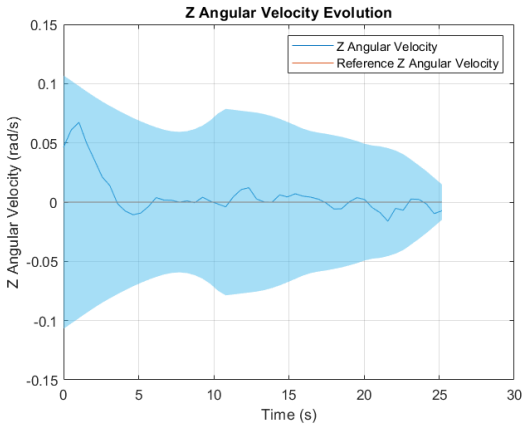

Figure 4.94: Evolution of the angular velocity relative to the z-axis

Another way to assess the performance of the MPC control strategy is to check how the control inputs evolve over time. As already mentioned above, in a real-world control system, the input values do not change too rapidly and suddenly, but rather progressively. That is the reason why the difference between two consecutive control input variables is penalized in the formulated optimization problem.

One can observe the evolution of the input values from Figures 4.95 to 4.100. The most predominant forces, which have a higher impact on the system's trajectory, are the thrust force applied in the z-direction and the torques around both the x- and y-axis. The corresponding plots show that these specific values have a smooth and gradual evolution. On the other hand, one can argue that the other values change too rapidly. However, such behavior can be explained by a simple factor. Firstly, the torque applied around the z-axis is much smaller than the others, and, consequently, such small variations are not concerning. Secondly, if one observes, for example, the thrust force applied in the y-direction, it goes from -2×10^4 to 2×10^4 in around 2.5 seconds. But, considering the total thrust force magnitude applied around that interval, that only represents a gimbal angle deviation of 8 degrees in 2.5 seconds, which is a feasible evolution. Hence, it is possible to verify that the evolution of the control input values is reasonable and realistic.

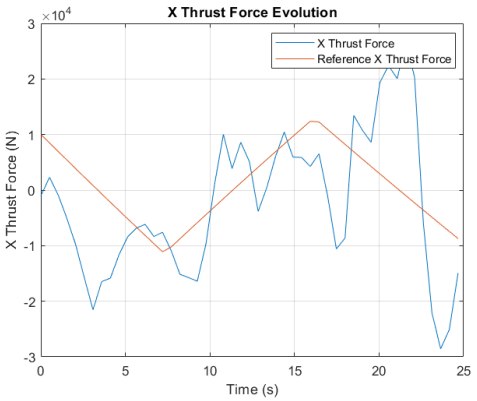

Figure 4.95: Evolution of the thrust force applied in the x-direction

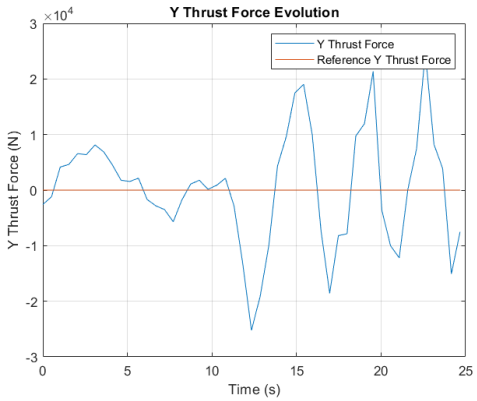

Figure 4.96: Evolution of the thrust force applied in the y-direction

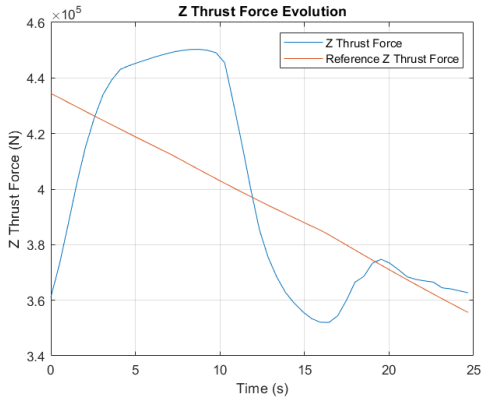

Figure 4.97: Evolution of the thrust force applied in the z-direction

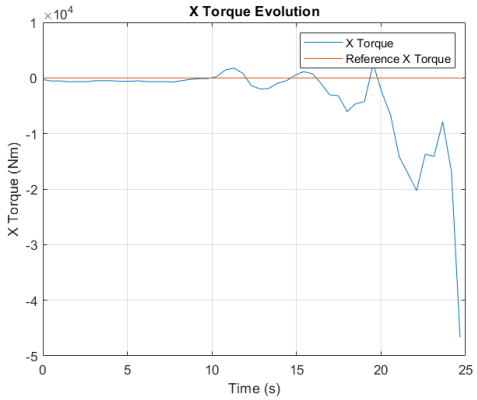

Figure 4.98: Evolution of the torque applied around the x-axis

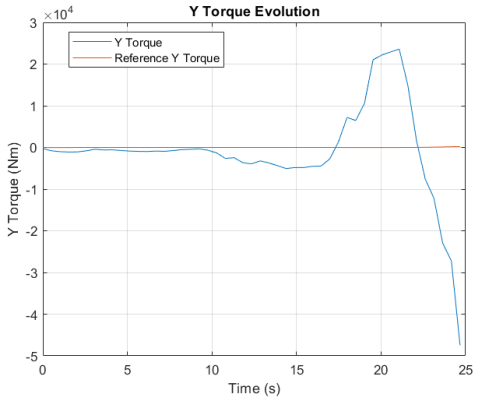

Figure 4.99: Evolution of the torque applied around the y-axis

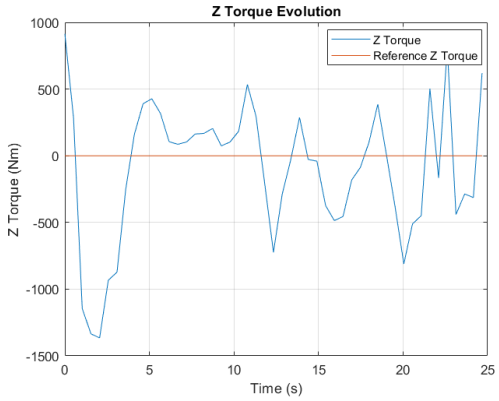

Figure 4.100: Evolution of the torque applied around the z-axis

However, as pointed out earlier, one can only take strong conclusions regarding the performance of the proposed MPC method, which leverages the funnel parameters to formulate a cost function and a terminal constraint, when comparing it with a standard MPC controller.

For the following results, the control horizon is set to $N = 10$. From Figures 4.101 to 4.112, it is possible to observe a comparison between the performance of the funnel feedback controller, the proposed MPC strategy and the standard MPC approach. First of all, the main conclusion is that the new MPC strategy has a much better performance than the feedback controller computed through the funnel synthesis optimization problem, which once again highlights the relevance of the proposed formulation to control systems with additive disturbances. Then, although it may seem that the standard MPC controller is more accurate in terms of rotational motion, in the end, it leads to larger position and velocity errors. This may be related with the fact that, even though it is possible to compute a control invariant set in each iteration, the optimal control problem would become infeasible towards the last seconds of simulation, possibly due to the terminal constraint.

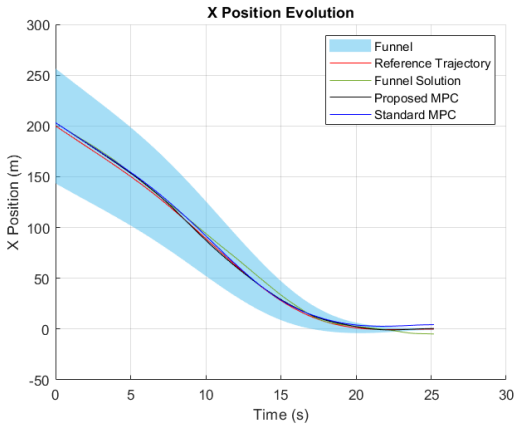

Figure 4.101: Evolution of the x-position

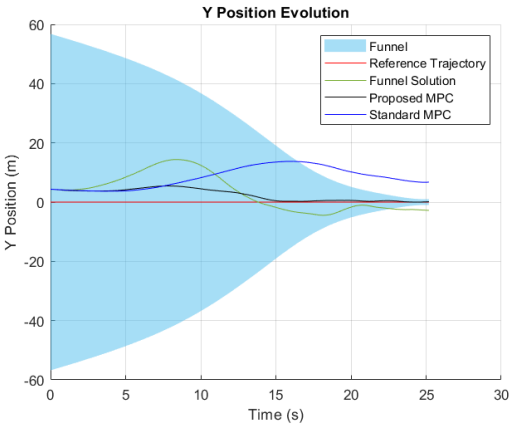

Figure 4.102: Evolution of the y-position

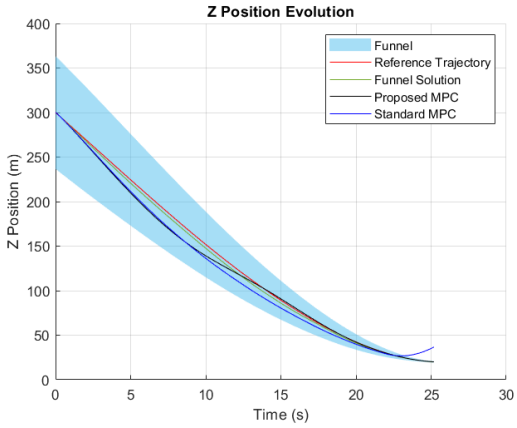

Figure 4.103: Evolution of the z-position

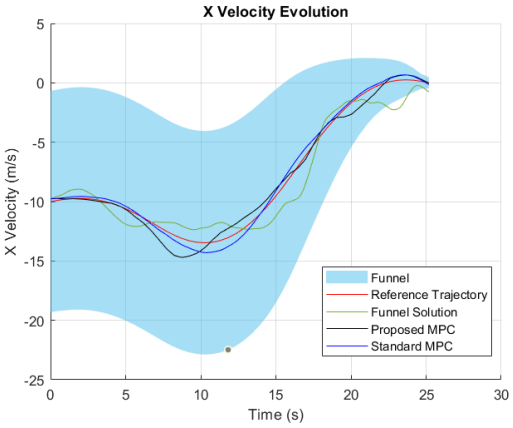

Figure 4.104: Evolution of the x-velocity

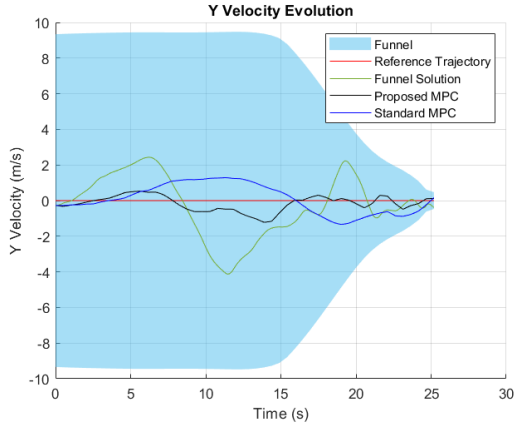

Figure 4.105: Evolution of the y-velocity

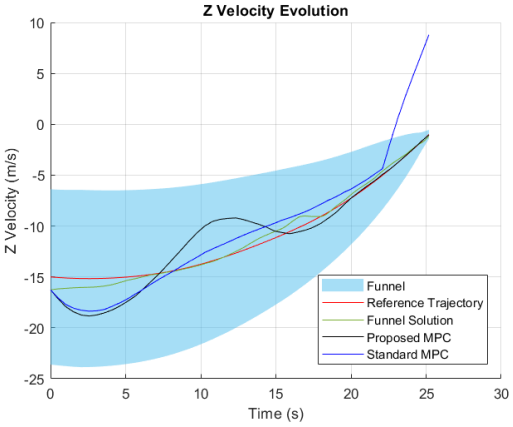

Figure 4.106: Evolution of the z-velocity

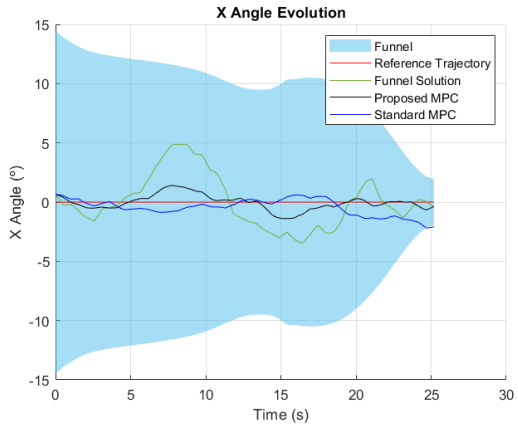

Figure 4.107: Evolution of the Euler angle relative to the x-axis

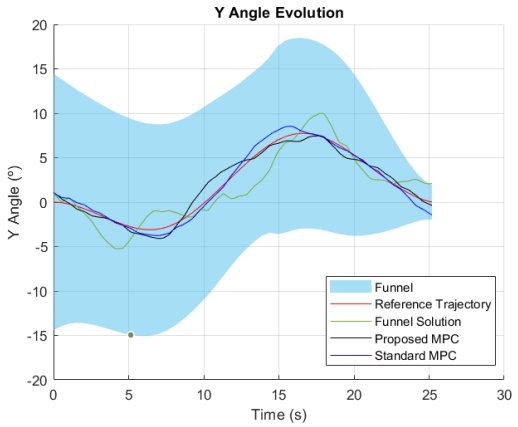

Figure 4.108: Evolution of the Euler angle relative to the y-axis

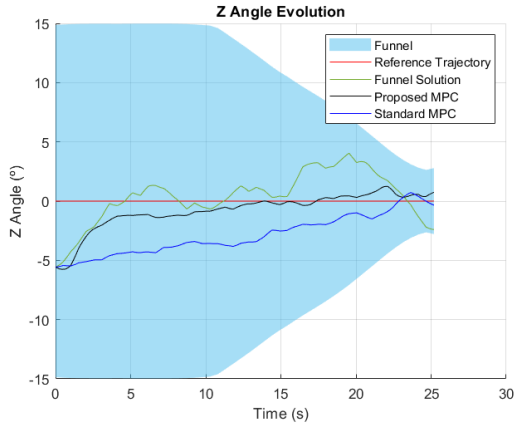

Figure 4.109: Evolution of the Euler angle relative to the z-axis

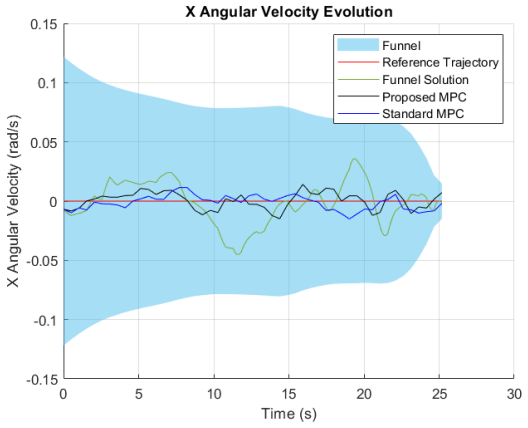

Figure 4.110: Evolution of the angular velocity relative to the x-axis

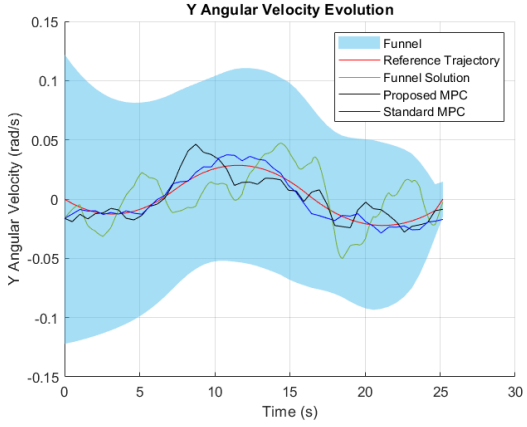

Figure 4.111: Evolution of the angular velocity relative to the y-axis

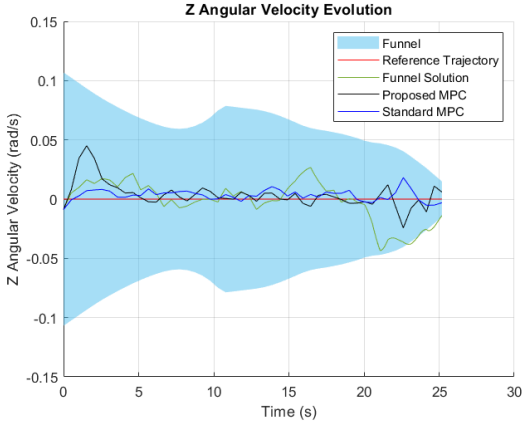

Figure 4.112: Evolution of the angular velocity relative to the z-axis

Hence, from the above scenarios, one can understand the importance of leveraging the funnel parameters to formulate the MPC optimal control problem, and successfully and accurately control a spacecraft throughout the Powered Descent Guidance (PDG) phase. The final state matches almost perfectly

the goal state, the landing procedure is smoother, the system trajectory evolves very close to the reference and there are no rough variations in the control input values' evolution. As supported above, its performance is better than both the funnel solution and the standard MPC controller.

The proposed formulation has two more advantages. Firstly, as concluded from several simulations, the standard MPC controller requires a control horizon of around 8 to 12 temporal nodes to lead to reasonable and acceptable results. Horizons of 3 to 6 temporal nodes would make the trajectory deviate from the reference right from the first seconds of simulation and, in some cases, lead to infeasible scenarios. On the other hand, the proposed MPC method shows a promising performance with small control horizons, of around 3 to 5 temporal nodes. Secondly, a standard MPC controller often requires a tedious fine-tuning process to perform reasonably well. The proposed approach does not suffer from this issue, as the parameters in the optimal control problem are determined by the funnel synthesis method.

4.4 Real-Time Recomputation Algorithm

For the real-time trajectory and funnel recomputation algorithm, the parameter N_1, which represents the number of temporal nodes between each recomputation step, is equal to 4. Regarding computation times, it was found that, for a trajectory with a total of 30 temporal nodes, the SCvx algorithm converges to an accurate solution in less than 0.3 seconds, which eventually decreases as the end of the landing procedure gets closer. On the other hand, each funnel synthesis optimization problem is solved in around 0.8 seconds.

On top of that, it is established that the last reference trajectory recomputation would occur when the last 8 temporal nodes are reached. This modification is implemented because it was found that, beyond that number, the reference trajectory would be less accurate and the control actions would change too suddenly in order to correct small errors.

The results of the real-time trajectory and funnel recomputation are shown from Figures 4.113 to 4.124. As observed, this strategy leads to a smooth trajectory, with less oscillations. In the end of the landing procedure, there are some small deviations in the velocity components but, overall, the landing position is reached.

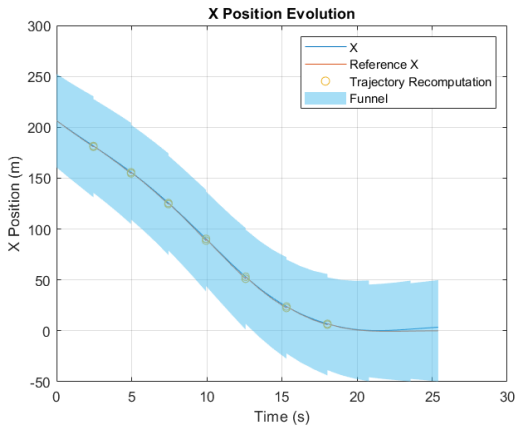

Figure 4.113: Evolution of the x-position

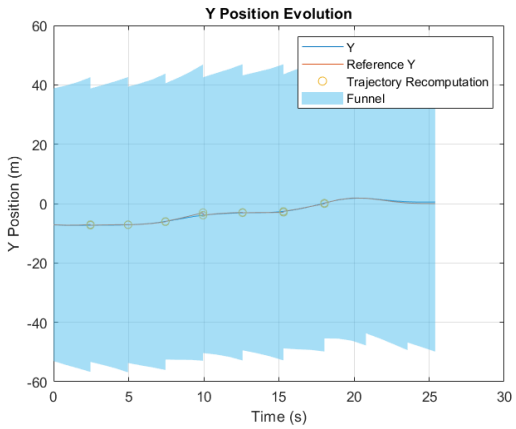

Figure 4.114: Evolution of the y-position

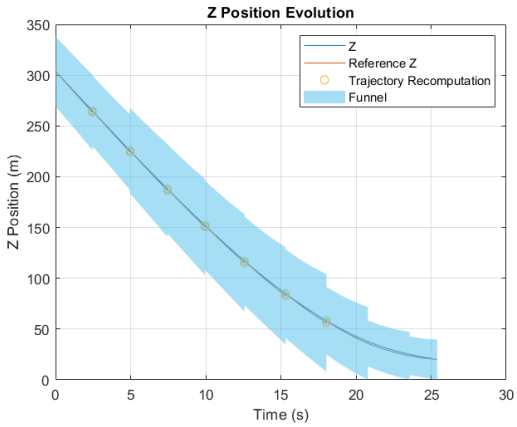

Figure 4.115: Evolution of the z-position

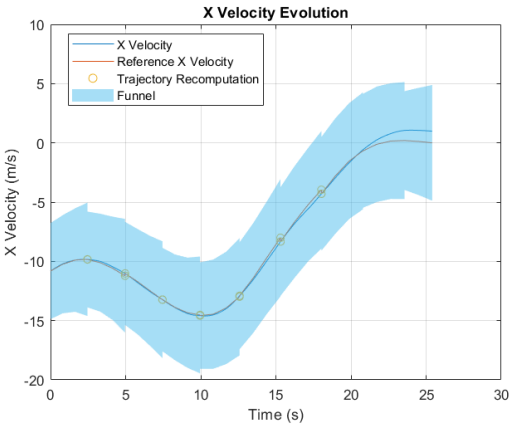

Figure 4.116: Evolution of the x-velocity

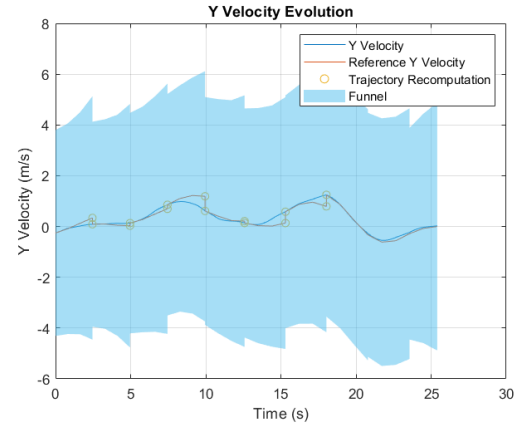

Figure 4.117: Evolution of the y-velocity

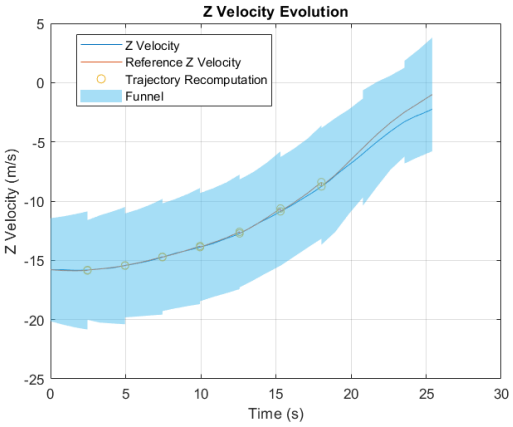

Figure 4.118: Evolution of the z-velocity

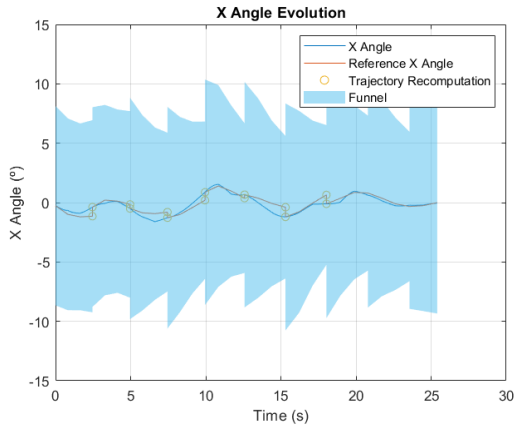

Figure 4.119: Evolution of the Euler angle relative to the x-axis

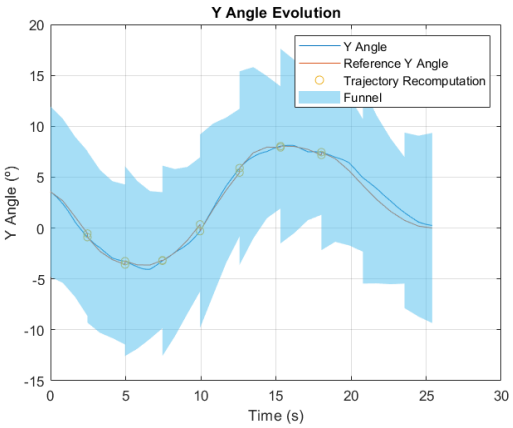

Figure 4.120: Evolution of the Euler angle relative to the y-axis

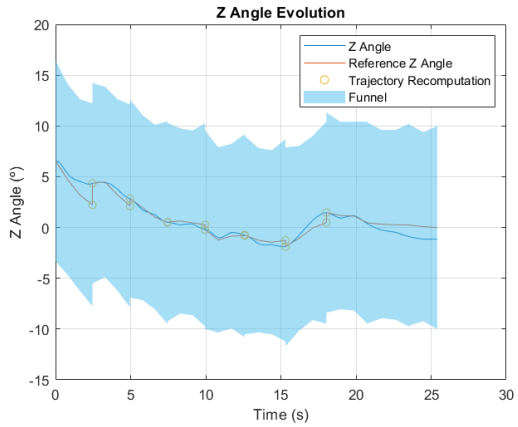

Figure 4.121: Evolution of the Euler angle relative to the z-axis

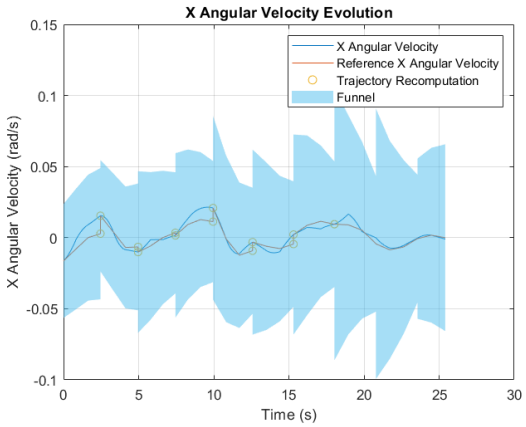

Figure 4.122: Evolution of the angular velocity relative to the x-axis

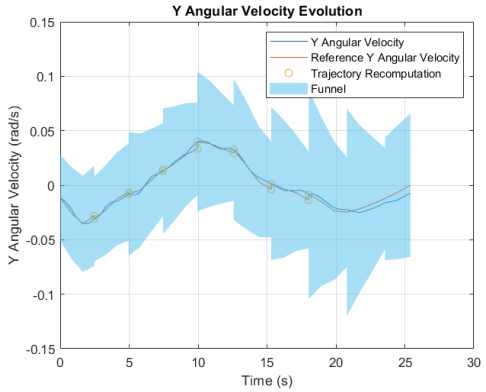

Figure 4.123: Evolution of the angular velocity relative to the y-axis

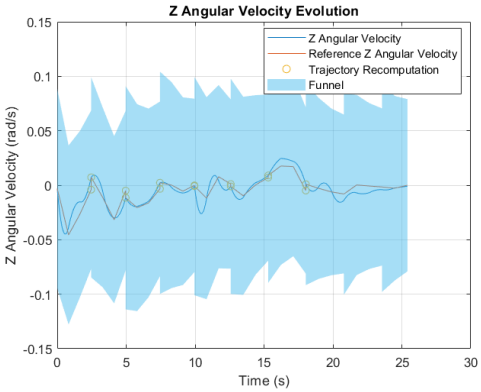

Figure 4.124: Evolution of the angular velocity relative to the z-axis

4.5 Final Performance Comparison

However, to truly evaluate the performance of the proposed methods, one must compare the three different approaches seen until now: the funnel and the feedback controller computed through an offline funnel synthesis optimization problem; the MPC algorithm which uses the funnel parameters to formulate a cost function and a terminal constraint; and the real-time trajectory and funnel recomputation procedure. This comparison can be seen from Figures 4.125 to 4.136.

First of all, it is possible to conclude that both of the proposed methods perform better than the funnel solution, the current state-of-the-art 6-DOF Powered Descend Guidance control method. Then, one can argue that both the MPC algorithm and the recomputation method have its strengths and limitations. The former is much more accurate, with the corresponding final state matching perfectly the goal state. However, in order to guarantee that the control input signal is smooth, some constraints are added that make the system's trajectory overshoot and not converge immediately to the reference. On the other hand, the real-time trajectory and funnel recomputation method leads to a smoother trajectory, which converges very quickly to the reference, but, in the end, some minor deviations from the goal position

and velocity values are observed. Both proposed methods actually perform very similarly when it comes to rotational motion.

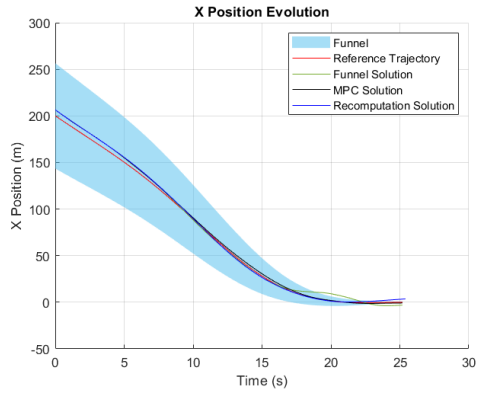

Figure 4.125: Evolution of the x-position

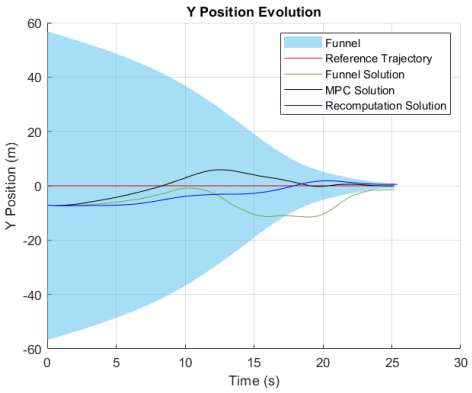

Figure 4.126: Evolution of the y-position

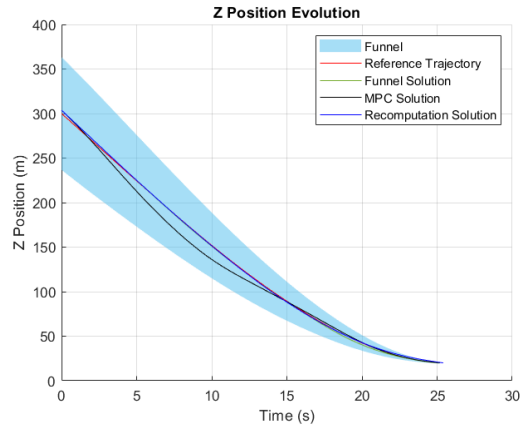

Figure 4.127: Evolution of the z-position

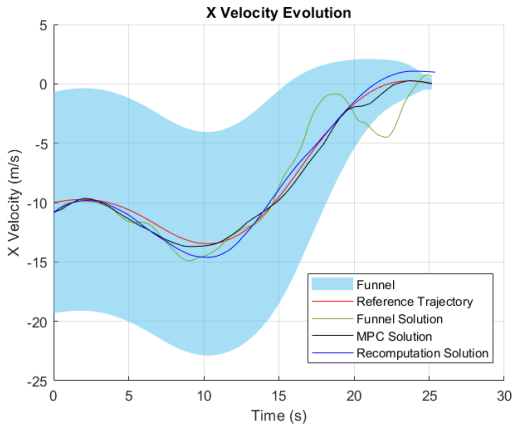

Figure 4.128: Evolution of the x-velocity

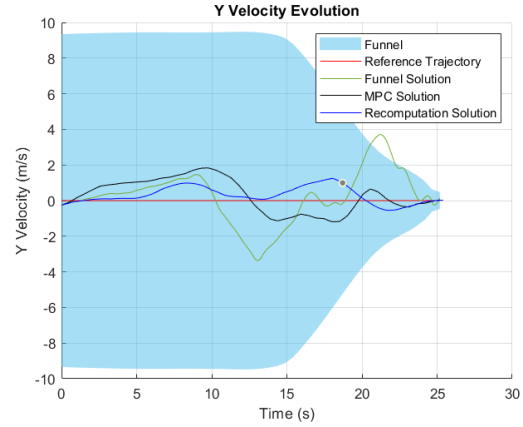

Figure 4.129: Evolution of the y-velocity

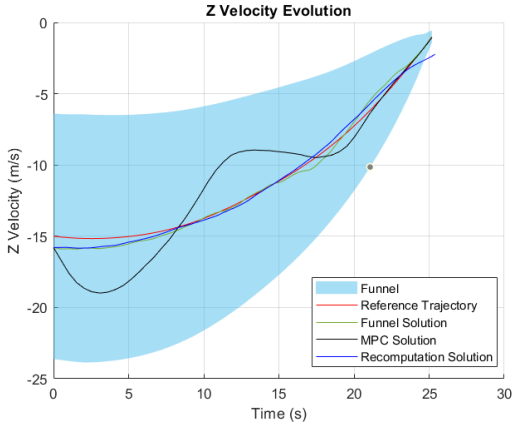

Figure 4.130: Evolution of the z-velocity

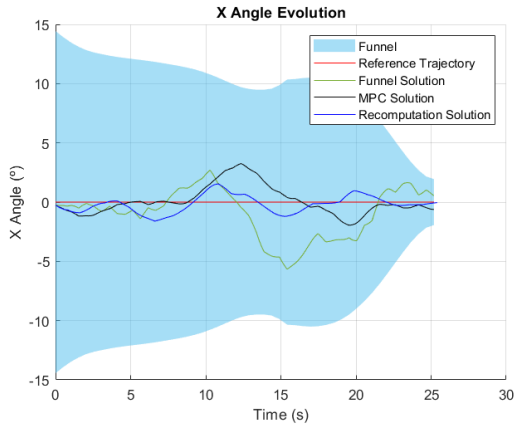

Figure 4.131: Evolution of the Euler angle relative to the x-axis

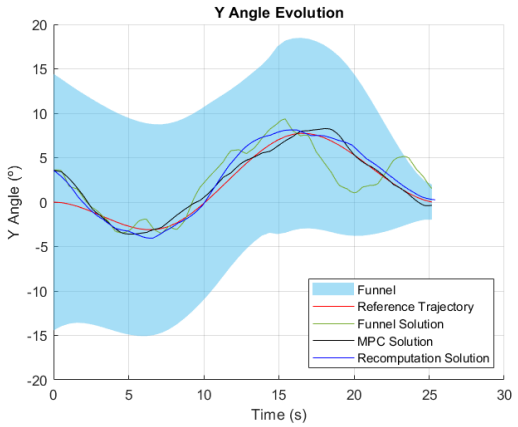

Figure 4.132: Evolution of the Euler angle relative to the y-axis

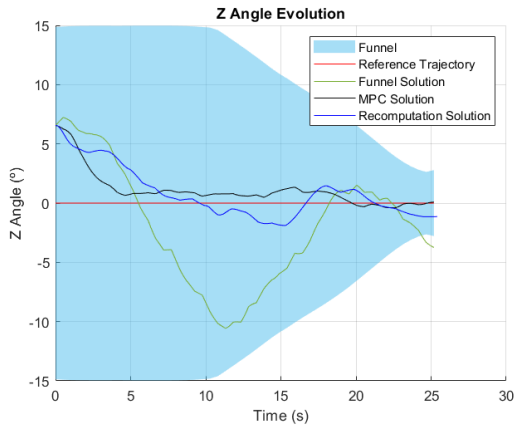

Figure 4.133: Evolution of the Euler angle relative to the z-axis

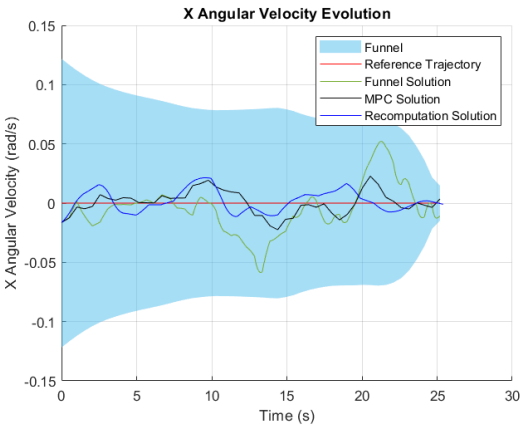

Figure 4.134: Evolution of the angular velocity relative to the x-axis

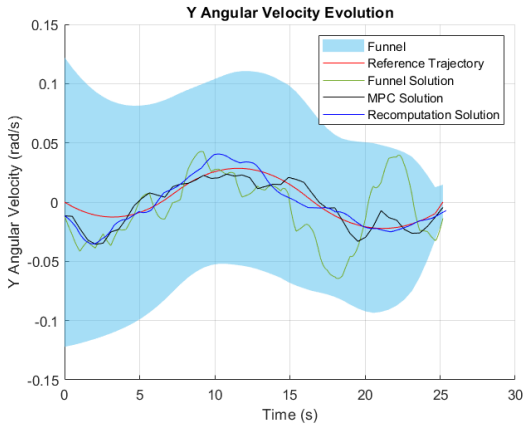

Figure 4.135: Evolution of the angular velocity relative to the y-axis

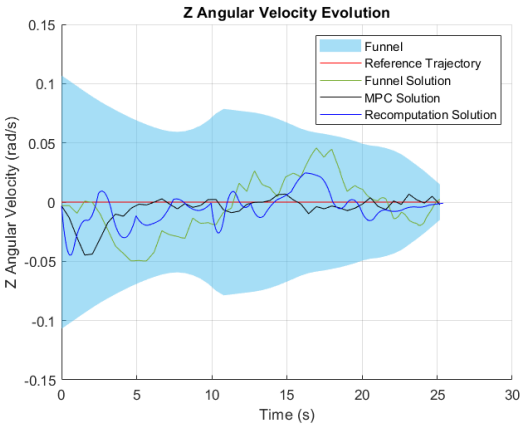

Figure 4.136: Evolution of the angular velocity relative to the z-axis

On top of that, to understand the overall landing accuracy of each control method, a additional simulation test is designed. Four approaches are assessed: the initial funnel synthesis method; the proposed funnel MPC controller; the standard MPC controller; and, finally, the proposed real-time trajectory and

funnel recomputation algorithm. For each of them, 20 randomly initialized trajectories are simulated using the dynamical model where additive disturbances affect the system's evolution. The disturbance upper bounds remain the same as in the previous scenarios.

At the end of the simulation, each trajectory is labelled either a successful or an unsuccessful landing. A successful landing occurs when the spacecraft's final position is within 4 meters of the goal position, each final velocity component is no further than 1 m/s from the corresponding references, the final Euler angles reach, at maximum, 2 degrees and the final angular velocities are in less than 0.02 rad/s from the corresponding references. If these requirements are not met, the trajectory is considered to be an unsuccessful landing.

A summary of the results can be seen in Table 4.3. The proposed MPC approach, where funnel parameters are used within the optimization problem, is the one that shows, by far, the highest landing accuracy. The real-time trajectory and funnel recomputation algorithm is not as accurate because, as seen above, it always leads to small final deviations regarding the position coordinates and velocity components. Often, these deviations are too large and, consequently, result in trajectories labelled as unsuccessful landings. The advantages of the proposed recomputation procedure are more related with the minimization of rough variations and the fast convergence to the reference. The standard MPC controller never leads to a successful landing because the infeasibility of the optimization problem towards the last seconds of the simulation results in large errors in some position coordinates or velocity components.

Table 4.3: Results of the landing accuracy test

Control approach	Number of successful landings
Funnel synthesis method	3
Proposed MPC approach	18
Standard MPC approach	0
Real-time trajectory and funnel recomputation algorithm	8

Chapter 5

Conclusions

5.1 Achievements

The main aim of this thesis project was to study the state-of-the-art control methods that had been proposed for the 6-DOF Powered Descent Guidance (PDG) problem and understand their limitations. More specifically, we were rather interested in the robustness aspect of these algorithms and in how well they perform in scenarios where disturbances and uncertainty are considered. For that purpose, after completing an in-depth Literature Review, possible areas of improvement were identified and specific thesis objectives were defined.

To control a spacecraft throughout its landing procedure, one must, firstly, generate a reference trajectory that connects the initial and the goal states. To achieve such goal, the Successive Convexification (SCvx) algorithm has been proposed in the past. Due to its unique principle of iteratively solving convex relaxations of the original nonlinear optimization problem and keep updating the corresponding solution, SCvx has repeatedly shown its usefulness and its ability to find an optimal and feasible trajectory rather quickly. Its lower computational complexity allows it to run in real-time and onboard the spacecraft computers.

After a reference trajectory has been optimized, one should focus on real-time trajectory tracking. For that purpose, funnel synthesis is the current state-of-the-art method. Besides computing a feedback controller that is able to correct deviations from the reference trajectory, the funnel synthesis optimization problem maximizes a control invariant set. This set mathematically guarantees that, if a system's initial condition is inside the corresponding entry, the trajectory will remain within the given set for all future times and actually converge to the reference. This introduces a robustness component, since the goal state is guaranteed to be reached for a wide set of distinct initial conditions.

However, after the implementation of the two aforementioned methods, it was found that their performance considerably decreases when a simulation model with additive disturbances is considered. The funnel is no longer control invariant, rough and sudden state variations are introduced and the final state is not as close to the goal state as one would expect.

For that reason, different control approaches should be exploited. Firstly, two modifications to the

funnel synthesis problem formulation were introduced: the funnel entry was maximized while the remaining funnel size was minimized; and the control invariance conditions of the funnel were derived based on a dynamical model which already considered additive disturbances beforehand. Nevertheless, even though they would lead to trajectories which evolved closer to the reference, the performance improvement was not significant and the effects of the external perturbations were still noticeable.

Therefore, two novel approaches were proposed. Their main difference with the state-of-the-art methods encountered throughout the literature is that they take into account the real-time evolution of the system's trajectory to readjust the corresponding control law and further mitigate the disturbances. Since the proposed algorithms are intended to run in real-time, they are designed to be much less computationally complex.

The first proposed approach is a Model Predictive Control (MPC) algorithm which leverages the parameters of a funnel computed offline to formulate a cost function and a terminal constraint. The corresponding performance was shown to be very promising. The trajectories are smoother, the system quickly converges to the reference and, consequently, the landing accuracy increases to a great extent.

To further outline the relevance of the proposed approach, its performance was compared to a standard MPC controller. Even though the results were very similar when it came to rotational motion, the standard MPC controller would lead to large errors in some position coordinates or velocity components at the end of the simulation. On top of that, two additional advantages should be mentioned. Firstly, it was found that the proposed MPC approach does not require a large control horizon to be accurate. Predicting the system evolution to 3 to 5 temporal nodes ahead would be sufficient to achieve a noteworthy performance, whereas the standard MPC controller would only lead to reasonable results with a control horizon of more than 8 nodes. Secondly, the typical MPC approach requires a tedious fine-tuning process to achieve an acceptable performance, while the proposed approach does not consider any user-defined parameters. It is only dependent on the result of the funnel synthesis optimization problem.

The second proposed approach incorporates real-time trajectory and funnel recomputation steps. From time to time, and given the actual system evolution, a new reference trajectory from the current state to the goal state is computed, around which a new funnel is constructed and, consequently, a new time-varying feedback controller is designed. This algorithm introduced a large improvement in performance when compared to the initial funnel synthesis solution. The trajectories are smoother, the effects of the disturbances are less visible and the system's trajectory closely tracks the reference.

Finally, the performance of the funnel synthesis method and the two proposed approaches were compared. Both of the new approaches showed a major improvement regarding real-time reference tracking, even with dynamical models with additive disturbances. In terms of landing accuracy, the proposed MPC method proved to be, by far, the most reliable. The real-time trajectory and funnel recomputation algorithm still has a higher landing accuracy than the original funnel synthesis method, but it sometimes leads to undesired deviations from the final goal state.

Hence, this thesis project was able to, in the end, introduce two new approaches that deliver promising results when applied to such a complex control problem such as a 6-DOF PDG landing procedure

which is directly affected by disturbances. These methods would benefit from a more careful formal analysis to its properties but we were able to take an important step towards the further advancement of research on the topic of robust PDG methods.

5.2 Future Work

Regarding future work, three main areas should be discussed. Firstly, an in-depth formal analysis to the stability of the proposed real-time trajectory and funnel recomputation algorithm should be conducted. The conclusions drawn above are only based on the empirical observation of the outcome of different simulations, but, nevertheless, it is important to mathematically prove the stability of this approach. Such research task may even point towards modifications in the algorithm formulation that consequently lead to better results.

Then, and most importantly, one should investigate if other real-time control strategies, based on the funnel synthesis method, can be formulated and if they contribute to performance improvement. For example, an idea would be to include a prediction of the system evolution inside the funnel synthesis optimization problem. Such modification would certainly lead to a nonlinear and nonconvex problem, but it might be possible to apply a technique to decrease the complexity of such formulation. A second idea would be to derive a simpler parameterization of the funnel and the feedback controller. This procedure would be able to preserve the essential information of the offline solution, but, simultaneously, allow for quick adjustments and recomputations in real-time. This could lead to a more conservative and robust feedback controller. Other paths can also be explored, the reality is that the topic of real-time funnel recomputation still has plenty of room for improvement.

Finally, one could also study the possibility of applying classical robust control techniques, such as $\mathcal{H}_\infty$ controller synthesis, to the 6-DOF PDG control problem. The funnel synthesis approach could provide an initial idea of a suitable structure for the time-varying feedback controller, but a method such as $\mathcal{H}_\infty$ loop-shaping could be used to analyze the sensitivity of the given controller and, then, further improve the robust performance.

Bibliography

[1] L. Blackmore. Autonomous Precision Landing of Space Rockets. In *Frontiers of Engineering: Reports on Leading-Edge Engineering from the 2016 Symposium*, volume 46, pages 15–20. The Bridge Washington, DC, 2016.

[2] R. Sostaric and J. Rea. Powered Descent Guidance Methods For The Moon and Mars. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, 2005. doi: 10.2514/6.2005-6287. URL https://arc.aiaa.org/doi/abs/10.2514/6.2005-6287.

[3] M. Szmuk, T. P. Reynolds, and B. Açıkmeşe. Successive Convexification for Real-Time Six-Degree-of-Freedom Powered Descent Guidance with State-Triggered Constraints. *Journal of Guidance, Control, and Dynamics*, 43(8):1399–1413, 2020. doi: 10.2514/1.G004549. URL https://doi.org/10.2514/1.G004549.

[4] D. S. Hochbaum. Complexity and Algorithms for Nonlinear Optimization Problems. *Ann. Oper. Res.*, 153(1):257–296, Sept. 2007. ISSN 1572-9338. doi: 10.1007/s10479-007-0172-6.

[5] M. Ragab and F. M. Cheatwood. Launch vehicle recovery and reuse. In *AIAA SPACE 2015 Conference and Exposition*, page 4490. doi: 10.2514/6.2015-4490. URL https://arc.aiaa.org/doi/abs/10.2514/6.2015-4490.

[6] D. Malyuta, Y. Yu, P. Elango, and B. Açıkmeşe. Advances in Trajectory Optimization for Space Vehicle Control. *Annual Reviews in Control*, 52:282–315, 2021. ISSN 1367-5788. doi: https://doi.org/10.1016/j.arcontrol.2021.04.013. URL https://www.sciencedirect.com/science/article/pii/S1367578821000377.

[7] A. Majumdar and R. Tedrake. Funnel Libraries for Real-Time Robust Feedback Motion Planning. *The International Journal of Robotics Research*, 36(8):947–982, 2017. doi: 10.1177/0278364917712421. URL https://doi.org/10.1177/0278364917712421.

[8] T. Reynolds, D. Malyuta, M. Mesbahi, B. Acikmese, and J. M. Carson. Funnel Synthesis for the 6-DOF Powered Descent Guidance Problem. In *AIAA Scitech 2021 Forum*, page 0504, 2021. doi: 10.2514/6.2021-0504. URL https://arc.aiaa.org/doi/abs/10.2514/6.2021-0504.

[9] J. Nevins. Man-Machine Design for the Apollo Navigation, Guidance, and Control System – Revisited: Apollo, A Transition in the Art of Piloting a Vehicle. *IFAC Proceedings Volumes*, 3

(1):1–27, 1970. ISSN 1474-6670. doi: https://doi.org/10.1016/S1474-6670(17)68763-9. URL https://www.sciencedirect.com/science/article/pii/S1474667017687639. 3rd International IFAC Conference on Automatic Control in Space, Toulouse, France, March 2-6, 1970.

[10] J. A. Saponaro and S. L. Copps. Operations and Functions of the Apollo Guidance Computer During Rendezvous. *IFAC Proceedings Volumes*, 3(1):116–135, 1970. ISSN 1474-6670. doi: https://doi.org/10.1016/S1474-6670(17)68772-X. URL https://www.sciencedirect.com/science/article/pii/S147466701768772X. 3rd International IFAC Conference on Automatic Control in Space, Toulouse, France, March 2-6, 1970.

[11] A. R. Klumpp. Apollo Lunar Descent Guidance. *Automatica*, 10(2):133–146, 1974. ISSN 0005-1098. doi: https://doi.org/10.1016/0005-1098(74)90019-3. URL https://www.sciencedirect.com/science/article/pii/0005109874900193.

[12] G. Mendeck and L. Craig. Entry Guidance for the 2011 Mars Science Laboratory Mission. In *AIAA Atmospheric Flight Mechanics Conference*, 2011. doi: 10.2514/6.2011-6639. URL https://arc.aiaa.org/doi/abs/10.2514/6.2011-6639.

[13] S. N. D. Souza and N. Sarigul-Klijn. Survey of Planetary Entry Guidance Algorithms. *Progress in Aerospace Sciences*, 68:64–74, 2014. ISSN 0376-0421. doi: https://doi.org/10.1016/j.paerosci.2014.01.002. URL https://www.sciencedirect.com/science/article/pii/S0376042114000293.

[14] C. T. Chomel and R. H. Bishop. Analytical Lunar Descent Guidance Algorithm. *Journal of Guidance, Control, and Dynamics*, 32(3):915–926, 2009. doi: 10.2514/1.37700. URL https://doi.org/10.2514/1.37700.

[15] D. Lawden. *Optimal Trajectories for Space Navigation*. 1963. URL https://books.google.pt/books?id=5c1QcgAACAAJ.

[16] J. Meditch. On the Problem of Optimal Thrust Programming for a Lunar Soft Landing. *IEEE Transactions on Automatic Control*, 9(4):477–484, 1964. doi: 10.1109/TAC.1964.1105758.

[17] J. V. Breakwell and J. F. Dixon. Minimum-Fuel Rocket Trajectories Involving Intermediate-Thrust Arcs. *J. Optim. Theory Appl.*, 17(5):465–479, Dec. 1975. ISSN 1573-2878. doi: 10.1007/BF00932784.

[18] A. G. Azizov and N. A. Korshunova. On an Analytical Solution of the Optimum Trajectory Problem in a Gravitational Field. *Celestial Mechanics*, 38(4):297–306, Apr. 1986. ISSN 1572-9478. doi: 10.1007/BF01238922.

[19] C. D'Souza. An Optimal Guidance Law for Planetary Landing. In *AIAA Guidance, Navigation, and Control Conference*, page 3709, 1997. doi: 10.2514/6.1997-3709. URL https://arc.aiaa.org/doi/abs/10.2514/6.1997-3709.

[20] U. Topcu, J. Casoliva, and K. D. Mease. Minimum-Fuel Powered Descent for Mars Pinpoint Landing. *Journal of Spacecraft and Rockets*, 44(2):324–331, 2007. doi: 10.2514/1.25023. URL https://doi.org/10.2514/1.25023.

[21] P. Lu. Propellant-Optimal Powered Descent Guidance. *Journal of Guidance, Control, and Dynamics*, 41(4):813–826, 2018. doi: 10.2514/1.G003243. URL https://doi.org/10.2514/1.G003243.

[22] J. Rea and R. Bishop. Analytical Dimensional Reduction of a Fuel Optimal Powered Descent Subproblem. In *AIAA Guidance, Navigation, and Control Conference*, page 8026, 2010. doi: 10.2514/6.2010-8026. URL https://arc.aiaa.org/doi/abs/10.2514/6.2010-8026.

[23] P. Lu, S. Forbes, and M. Baldwin. A Versatile Powered Guidance Algorithm. In *AIAA Guidance, Navigation, and Control Conference*, page 4843, 2012. doi: 10.2514/6.2012-4843. URL https://arc.aiaa.org/doi/abs/10.2514/6.2012-4843.

[24] B. A. Steinfeldt, M. J. Grant, D. A. Matz, R. D. Braun, and G. H. Barton. Guidance, Navigation, and Control System Performance Trades for Mars Pinpoint Landing. *Journal of Spacecraft and Rockets*, 47(1):188–198, 2010. doi: 10.2514/1.45779. URL https://doi.org/10.2514/1.45779.

[25] B. Acikmese and S. Ploen. A Powered Descent Guidance Algorithm for Mars Pinpoint Landing. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, page 6288, 2005. doi: 10.2514/6.2005-6288. URL https://arc.aiaa.org/doi/abs/10.2514/6.2005-6288.

[26] S. Ploen, B. Acikmese, and A. Wolf. A Comparison of Powered Descent Guidance Laws for Mars Pinpoint Landing. In *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, page 6676, 2006. doi: 10.2514/6.2006-6676. URL https://arc.aiaa.org/doi/abs/10.2514/6.2006-6676.

[27] B. Acikmese and S. R. Ploen. Convex Programming Approach to Powered Descent Guidance for Mars Landing. *Journal of Guidance, Control, and Dynamics*, 30(5):1353–1366, 2007. doi: 10.2514/1.27553. URL https://doi.org/10.2514/1.27553.

[28] J. M. Carson, B. Açikmeşe, and L. Blackmore. Lossless Convexification of Powered-Descent Guidance with Non-Convex Thrust Bound and Pointing Constraints. In *Proceedings of the 2011 American Control Conference*, pages 2651–2656, 2011. doi: 10.1109/ACC.2011.5990959.

[29] B. Açıkmeşe, J. M. Carson, and L. Blackmore. Lossless Convexification of Nonconvex Control Bound and Pointing Constraints of the Soft Landing Optimal Control Problem. *IEEE Transactions on Control Systems Technology*, 21(6):2104–2113, 2013. doi: 10.1109/TCST.2012.2237346.

[30] L. Blackmore, B. Açikmeşe, and D. P. Scharf. Minimum-Landing-Error Powered-Descent Guidance for Mars Landing Using Convex Optimization. *Journal of Guidance, Control, and Dynamics*, 33(4):1161–1171, 2010. doi: 10.2514/1.47202. URL https://doi.org/10.2514/1.47202.

[31] D. Scharf, M. Regehr, G. Vaughan, J. Benito, H. Ansari, M. Aung, A. Johnson, J. Casoliva, S. Mohan, D. Dueri, B. Açıkmeşe, D. Masten, and S. Nietfeld. ADAPT Demonstrations of Onboard Large-

Divert Guidance with a VTVL Rocket. pages 1–18, 03 2014. ISBN 978-1-4799-1622-1. doi: 10.1109/AERO.2014.6836462.

[32] D. P. Scharf, B. Açıkmeşe, D. Dueri, J. Benito, and J. Casoliva. Implementation and Experimental Demonstration of Onboard Powered-Descent Guidance. *Journal of Guidance, Control, and Dynamics*, 40(2):213–229, 2017. doi: 10.2514/1.G000399. URL https://doi.org/10.2514/1.G000399.

[33] X. Liu and P. Lu. Solving Nonconvex Optimal Control Problems by Convex Optimization. *Journal of Guidance, Control, and Dynamics*, 37(3):750–765, 2014. doi: 10.2514/1.62110. URL https://doi.org/10.2514/1.62110.

[34] M. Szmuk, B. Acikmese, and A. W. Berning. Successive Convexification for Fuel-Optimal Powered Landing with Aerodynamic Drag and Non-Convex Constraints. In *AIAA Guidance, Navigation, and Control Conference*, page 378, 2016. doi: 10.2514/6.2016-0378. URL https://arc.aiaa.org/doi/abs/10.2514/6.2016-0378.

[35] A. M. Dwyer-Cianciolo, C. D. Karlgaard, D. Woffinden, R. A. Lugo, J. Tynis, R. R. Sostaric, S. Striepe, R. Powell, and J. M. Carson. Defining Navigation Requirements for Future Missions. In *AIAA Scitech 2019 Forum*, page 0661. doi: 10.2514/6.2019-0661. URL https://arc.aiaa.org/doi/abs/10.2514/6.2019-0661.

[36] U. Lee and M. Mesbahi. Dual Quaternions, Rigid Body Mechanics, and Powered-Descent Guidance. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, pages 3386–3391, 2012. doi: 10.1109/CDC.2012.6425996.

[37] U. Lee and M. Mesbahi. Constrained Autonomous Precision Landing via Dual Quaternions and Model Predictive Control. *Journal of Guidance, Control, and Dynamics*, 40(2):292–308, 2017. doi: 10.2514/1.G001879. URL https://doi.org/10.2514/1.G001879.

[38] M. Szmuk, U. Eren, and B. Acikmese. Successive Convexification for Mars 6-DoF Powered Descent Landing Guidance. In *AIAA Guidance, Navigation, and Control Conference*, page 1500, 2017. doi: 10.2514/6.2017-1500. URL https://arc.aiaa.org/doi/abs/10.2514/6.2017-1500.

[39] M. Szmuk and B. Acikmese. Successive Convexification for 6-DoF Mars Rocket Powered Landing with Free-Final-Time. In *2018 AIAA Guidance, Navigation, and Control Conference*, page 0617, 2018. doi: 10.2514/6.2018-0617. URL https://arc.aiaa.org/doi/abs/10.2514/6.2018-0617.

[40] T. Reynolds, M. Szmuk, D. Malyuta, M. Mesbahi, B. Acikmese, and J. M. Carson. A State-Triggered Line of Sight Constraint for 6-DoF Powered Descent Guidance Problems. In *AIAA Scitech 2019 Forum*, page 0924, 2019. doi: 10.2514/6.2019-0924. URL https://arc.aiaa.org/doi/abs/10.2514/6.2019-0924.

[41] M. Szmuk, T. Reynolds, B. Acikmese, M. Mesbahi, and J. M. Carson. Successive Convexification for 6-DoF Powered Descent Guidance with Compound State-Triggered Constraints. In *AIAA Scitech*

2019 Forum, page 0926, 2019. doi: 10.2514/6.2019-0926. URL https://arc.aiaa.org/doi/abs/10.2514/6.2019-0926.

[42] T. P. Reynolds, M. Szmuk, D. Malyuta, M. Mesbahi, B. Açıkmeşe, and J. M. Carson. Dual Quaternion-Based Powered Descent Guidance with State-Triggered Constraints. *Journal of Guidance, Control, and Dynamics*, 43(9):1584–1599, 2020. doi: 10.2514/1.G004536. URL https://doi.org/10.2514/1.G004536.

[43] P. Elango, A. G. Kamath, Y. Yu, B. Acikmese, M. Mesbahi, and J. M. Carson. A Customized First-Order Solver for Real-Time Powered-Descent Guidance. In *AIAA SCITECH 2022 Forum*, page 0951, 2022. doi: 10.2514/6.2022-0951. URL https://arc.aiaa.org/doi/abs/10.2514/6.2022-0951.

[44] A. G. Kamath, P. Elango, T. Kim, S. Mceowen, Y. Yu, J. M. Carson, M. Mesbahi, and B. Acikmese. Customized Real-Time First-Order Methods for Onboard Dual Quaternion-based 6-DoF Powered-Descent Guidance. In *AIAA SCITECH 2023 Forum*, 2023. doi: 10.2514/6.2023-2003. URL https://arc.aiaa.org/doi/abs/10.2514/6.2023-2003.

[45] D. Rutishauser, G. Mendeck, R. Ramadorai, J. Prothro, T. Fleming, and P. Fidelman. NASA and Blue Origin's Flight Assessment of Precision Landing Algorithms Computing Performance. In *AIAA SCITECH 2022 Forum*, page 1832, 2022. doi: 10.2514/6.2022-1832. URL https://arc.aiaa.org/doi/abs/10.2514/6.2022-1832.

[46] M. Fritz, J. Doll, K. C. Ward, G. Mendeck, R. R. Sostaric, S. Pedrotty, C. Kuhl, B. Acikmese, S. R. Bieniawski, L. Strohl, and A. W. Berning. Post-Flight Performance Analysis of Navigation and Advanced Guidance Algorithms on a Terrestrial Suborbital Rocket Flight. In *AIAA SCITECH 2022 Forum*, page 0765, 2022. doi: 10.2514/6.2022-0765. URL https://arc.aiaa.org/doi/abs/10.2514/6.2022-0765.

[47] L. Strohl, J. Doll, M. Fritz, A. W. Berning, S. White, S. R. Bieniawski, J. M. Carson, and B. Acikmese. Implementation of a Six Degree of Freedom Precision Lunar Landing Algorithm Using Dual Quaternion Representation. In *AIAA SCITECH 2022 Forum*, page 1831, 2022. doi: 10.2514/6.2022-1831. URL https://arc.aiaa.org/doi/abs/10.2514/6.2022-1831.

[48] P. Simplício, A. Marcos, and S. Bennani. Reusable Launchers: Development of a Coupled Flight Mechanics, Guidance, and Control Benchmark. *Journal of Spacecraft and Rockets*, 57(1):74–89, 2020. doi: 10.2514/1.A34429. URL https://doi.org/10.2514/1.A34429.

[49] P. Simplício, A. Marcos, and S. Bennani. Guidance of Reusable Launchers: Improving Descent and Landing Performance. *Journal of Guidance, Control, and Dynamics*, 42(10):2206–2219, 2019. doi: 10.2514/1.G004155. URL https://doi.org/10.2514/1.G004155.

[50] E. Dumont, S. Ishimoto, P. Tatiossian, J. Klevanski, B. Reimann, T. Ecker, L. Witte, J. Riehmer, M. Sagliano, S. G. Vincenzino, et al. Callisto: A Demonstrator for Reusable Launcher Key Tech-

nologies. *Transactions of the Japan Society for Aeronautical and Space Sciences, Aerospace Technology Japan*, 19(1):106–115, 2021. doi: 10.2322/tastj.19.106.

[51] X. Liu, Z. Shen, and P. Lu. Entry Trajectory Optimization by Second-Order Cone Programming. *Journal of Guidance, Control, and Dynamics*, 39(2):227–241, 2016. doi: 10.2514/1.G001210. URL https://doi.org/10.2514/1.G001210.

[52] M. Sagliano. Pseudospectral Convex Optimization for Powered Descent and Landing. *Journal of Guidance, Control, and Dynamics*, 41(2):320–334, 2018. doi: 10.2514/1.G002818. URL https://doi.org/10.2514/1.G002818.

[53] Z. Wang and M. J. Grant. Constrained Trajectory Optimization for Planetary Entry via Sequential Convex Programming. *Journal of Guidance, Control, and Dynamics*, 40(10):2603–2615, 2017. doi: 10.2514/1.G002150. URL https://doi.org/10.2514/1.G002150.

[54] Z. Wang and Y. Lu. Improved Sequential Convex Programming Algorithms for Entry Trajectory Optimization. *Journal of Spacecraft and Rockets*, 57(6):1373–1386, 2020. doi: 10.2514/1.A34640. URL https://doi.org/10.2514/1.A34640.

[55] T. Reynolds, D. Malyuta, M. Mesbahi, B. Acikmese, and J. M. Carson. A Real-Time Algorithm for Non-Convex Powered Descent Guidance. In *AIAA Scitech 2020 Forum*, page 0844, 2020. doi: 10.2514/6.2020-0844. URL https://arc.aiaa.org/doi/abs/10.2514/6.2020-0844.

[56] J. V. Breakwell, J. L. Speyer, and A. E. Bryson. Optimization and Control of Nonlinear Systems Using the Second Variation. *Journal of the Society for Industrial and Applied Mathematics Series A Control*, 1(2):193–223, 1963. doi: 10.1137/0301011. URL https://doi.org/10.1137/0301011.

[57] H. Kelley. An Optimal Guidance Approximation Theory. *IEEE Transactions on Automatic Control*, 9 (4):375–380, 1964. doi: 10.1109/TAC.1964.1105747.

[58] H. Seywald and E. M. Cliff. Neighboring Optimal Control Based Feedback Law for the Advanced Launch System. *Journal of Guidance, Control, and Dynamics*, 17(6):1154–1162, 1994. doi: 10.2514/3.21327. URL https://doi.org/10.2514/3.21327.

[59] H. Yan, F. Fahroo, and I. M. Ross. Real-time Computation of Neighboring Optimal Control Laws. In *AIAA Guidance, Navigation, and Control Conference and Exhibit*, page 4657, 2002. doi: 10.2514/6.2002-4657. URL https://arc.aiaa.org/doi/abs/10.2514/6.2002-4657.

[60] M. R. Jardin and A. E. Bryson. Neighboring Optimal Aircraft Guidance in Winds. *Journal of Guidance, Control, and Dynamics*, 24(4):710–715, 2001. doi: 10.2514/2.4798. URL https://doi.org/10.2514/2.4798.

[61] M. S. Miah and W. Gueaieb. RFID-Based Mobile Robot Trajectory Tracking and Point Stabilization Through On-line Neighboring Optimal Control. *Journal of Intelligent Robotic Systems*, 78:377–399, 2014. doi: 10.1007/s10846-014-0048-3.

[62] M. Mason. The Mechanics of Manipulation. In *Proceedings. 1985 IEEE International Conference on Robotics and Automation*, volume 2, pages 544–548, 1985. doi: 10.1109/ROBOT.1985.1087242.

[63] R. R. Burridge, A. A. Rizzi, and D. E. Koditschek. Sequential Composition of Dynamically Dexterous Robot Behaviors. *The International Journal of Robotics Research*, 18(6):534–555, 1999. doi: 10.1177/02783649922066385. URL https://doi.org/10.1177/02783649922066385.

[64] A. A. Julius and G. J. Pappas. Trajectory Based Verification Using Local Finite-Time Invariance. In *Hybrid Systems: Computation and Control*, pages 223–236. Springer, 2009. doi: 10.1007/978-3-642-00602-9_16.

[65] R. Tedrake, I. R. Manchester, M. Tobenkin, and J. W. Roberts. LQR-trees: Feedback Motion Planning via Sums-of-Squares Verification. *The International Journal of Robotics Research*, 29(8):1038–1052, 2010. doi: 10.1177/0278364910369189. URL https://doi.org/10.1177/0278364910369189.

[66] M. M. Tobenkin, I. R. Manchester, and R. Tedrake. Invariant Funnels around Trajectories using Sum-of-Squares Programming. *IFAC Proceedings Volumes*, 44(1):9218–9223, 2011. doi: https://doi.org/10.3182/20110828-6-IT-1002.03098. URL https://www.sciencedirect.com/science/article/pii/S1474667016450925.

[67] P. P. Khargonekar, I. R. Petersen, and K. Zhou. Robust Stabilization of Uncertain Linear Systems: Quadratic Stabilizability and H_∞ Control Theory. *IEEE Transactions on Automatic Control*, 35 (3):356–361, 1990. doi: 10.1109/9.50357. URL https://api.semanticscholar.org/CorpusID:121814168.

[68] M. A. Rotea and P. P. Khargonekar. Stabilization of Uncertain Systems with Norm Bounded Uncertainty—A Control Lyapunov Function Approach. *SIAM Journal on Control and Optimization*, 27(6):1462–1476, 1989. doi: 10.1137/0327075. URL https://doi.org/10.1137/0327075.

[69] M. V. Kothare, V. Balakrishnan, and M. Morari. Robust Constrained Model Predictive Control Using Linear Matrix Inequalities. *Automatica*, 32(10):1361–1379, 1996. doi: https://doi.org/10.1016/0005-1098(96)00063-5. URL https://www.sciencedirect.com/science/article/pii/0005109896000635.

[70] B. Açıkmeşe, J. M. Carson III, and D. S. Bayard. A Robust Model Predictive Control Algorithm for Incrementally Conic Uncertain/Nonlinear Systems. *International Journal of Robust and Nonlinear Control*, 21(5):563–590, 2011. doi: https://doi.org/10.1002/rnc.1613. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/rnc.1613.

[71] M. Ono, M. Pavone, Y. Kuwata, and J. Balaram. Chance-Constrained Dynamic Programming with Application to Risk-Aware Robotic Space Exploration. *Autonomous Robots*, 39:555–571, 12 2015. doi: 10.1007/s10514-015-9467-7.

[72] J. Ridderhof and P. Tsiotras. Uncertainty Quantication and Control During Mars Powered Descent and Landing using Covariance Steering. In *2018 AIAA Guidance, Navigation, and Control Conference*, page 0611, 2018. doi: 10.2514/6.2018-0611. URL https://arc.aiaa.org/doi/abs/10.2514/6.2018-0611.

[73] J. Ridderhof and P. Tsiotras. Minimum-fuel Powered Descent in the Presence of Random Disturbances. In *AIAA Scitech 2019 Forum*, page 0646, 2019. doi: 10.2514/6.2019-0646. URL https://arc.aiaa.org/doi/abs/10.2514/6.2019-0646.

[74] L. Cheng, H. Wen, and D. Jin. Uncertain Parameters Analysis of Powered-Descent Guidance based on Chebyshev Interval Method. *Acta Astronautica*, 162:581–588, 2019. doi: https://doi.org/10.1016/j.actaastro.2019.05.040. URL https://www.sciencedirect.com/science/article/pii/S0094576518318290.

[75] M. Szmuk. *Successive Convexification & High Performance Feedback Control for Agile Flight*. PhD thesis, 2019. URL https://digital.lib.washington.edu/researchworks/handle/1773/44017.

[76] T. P. Reynolds. *Computational Guidance and Control for Aerospace Systems*. PhD thesis, 2020. URL https://digital.lib.washington.edu/researchworks/handle/1773/46722.

[77] J. C. Butcher. *Numerical Methods for Ordinary Differential Equations*. John Wiley & Sons, 2016.

[78] J. Nocedal and S. J. Wright. *Numerical Optimization*. Springer, 1999.

[79] L. D'Alto and M. J. Corless. Incremental Quadratic Stability. *Numerical Algebra, Control and Optimization*, 3:175–201, 03 2013. doi: 10.3934/naco.2013.3.175. URL https://api.semanticscholar.org/CorpusID:124642915.

[80] B. Açıkmeşe. Stability Analysis with Quadratic Lyapunov Functions: A Necessary and Sufficient Multiplier Condition. *Systems Control Letters - SYST CONTROL LETT*, 57, 01 2008. doi: 10.1016/j.sysconle.2007.06.018.

[81] B. Açıkmeşe, J. M. Carson III, and D. S. Bayard. A Robust Model Predictive Control Algorithm for Incrementally Conic Uncertain/Nonlinear Systems. *International Journal of Robust and Nonlinear Control*, 21(5):563–590, 2011. doi: https://doi.org/10.1002/rnc.1613. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/rnc.1613.

[82] S. Boyd, L. El Ghaoui, E. Feron, and V. Balakrishnan. *Linear Matrix Inequalities in System and Control Theory*. Society for Industrial and Applied Mathematics, 1994. doi: 10.1137/1.9781611970777. URL https://epubs.siam.org/doi/abs/10.1137/1.9781611970777.

[83] D. Kleinman. On an Iterative Technique for Riccati Equation Computations. *IEEE Transactions on Automatic Control*, 13(1):114–115, 1968. doi: 10.1109/TAC.1968.1098829.

[84] G. R. Wood and B. P. Zhang. Estimation of the Lipschitz Constant of a Function. *J. Global Optim.*, 8(1):91–103, Jan. 1996. ISSN 1573-2916. doi: 10.1007/BF00229304.

[85] T. P. Reynolds, D. Malyuta, M. Mesbahi, and B. Acikmese. Temporally-Interpolated Funnel Synthesis for Nonlinear Systems. *2nd RSS Workshop on Robust Autonomy: Tools for Safety in Real-World Uncertain Environments (RSS 2020)*, 2020. URL https://openreview.net/forum?id=cMoxvmU22Zg.

[86] J. Fejlek and S. Ratschan. Computing Funnels Using Numerical Optimization Based Falsifiers. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 4318–4324, 2022. doi: 10.1109/ICRA46639.2022.9811730.

[87] T. Kim, P. Elango, and B. Acikmese. Joint Synthesis of Trajectory and Controlled Invariant Funnel for Discrete-time Systems with Locally Lipschitz Nonlinearities. 2023. URL https://api.semanticscholar.org/CorpusID:252118592.

[88] H. Seo and C. Son. Fast Funnel Computation Using Multivariate Bernstein Polynomial. *IEEE Robotics and Automation Letters*, PP:1–1, 02 2021. doi: 10.1109/LRA.2021.3057569.

[89] T. Kim, P. Elango, T. P. Reynolds, B. Açıkmeşe, and M. Mesbahi. Optimization-Based Constrained Funnel Synthesis for Systems With Lipschitz Nonlinearities via Numerical Optimal Control. *IEEE Control Systems Letters*, 7:2875–2880, 2023. doi: 10.1109/LCSYS.2023.3290229.

[90] S. Singh, A. Majumdar, J.-J. Slotine, and M. Pavone. Robust Online Motion Planning via Contraction Theory and Convex Optimization. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5883–5890, 2017. doi: 10.1109/ICRA.2017.7989693.

[91] S. V. Rakovic, A. R. Teel, D. Q. Mayne, and A. Astolfi. Simple Robust Control Invariant Tubes for Some Classes of Nonlinear Discrete Time Systems. In *Proceedings of the 45th IEEE Conference on Decision and Control*, pages 6397–6402, 2006. doi: 10.1109/CDC.2006.377551.