POLITECNICO
MILANO 1863

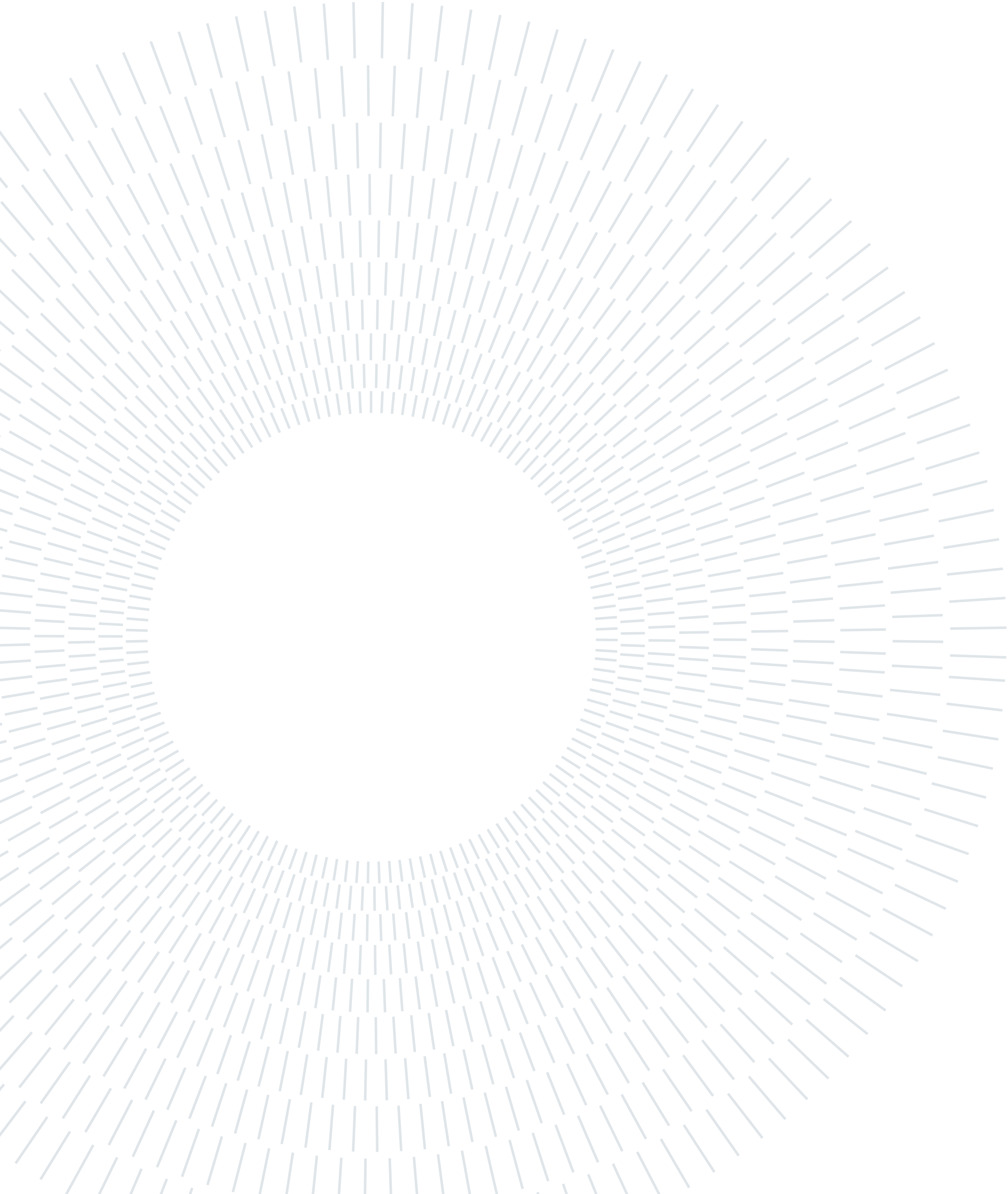
SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

# Differential Algebra Based Model Predictive Control for Spacecraft Rendezvous in Cislunar Space

Tesi di Laurea Magistrale in
Space Engineering - Ingegneria Spaziale

Author: **Michele Mapelli**

Student ID: 975859
Advisor: Prof. Mauro Massari
Co-advisors: Luca Giorcelli
Academic Year: 2024-25



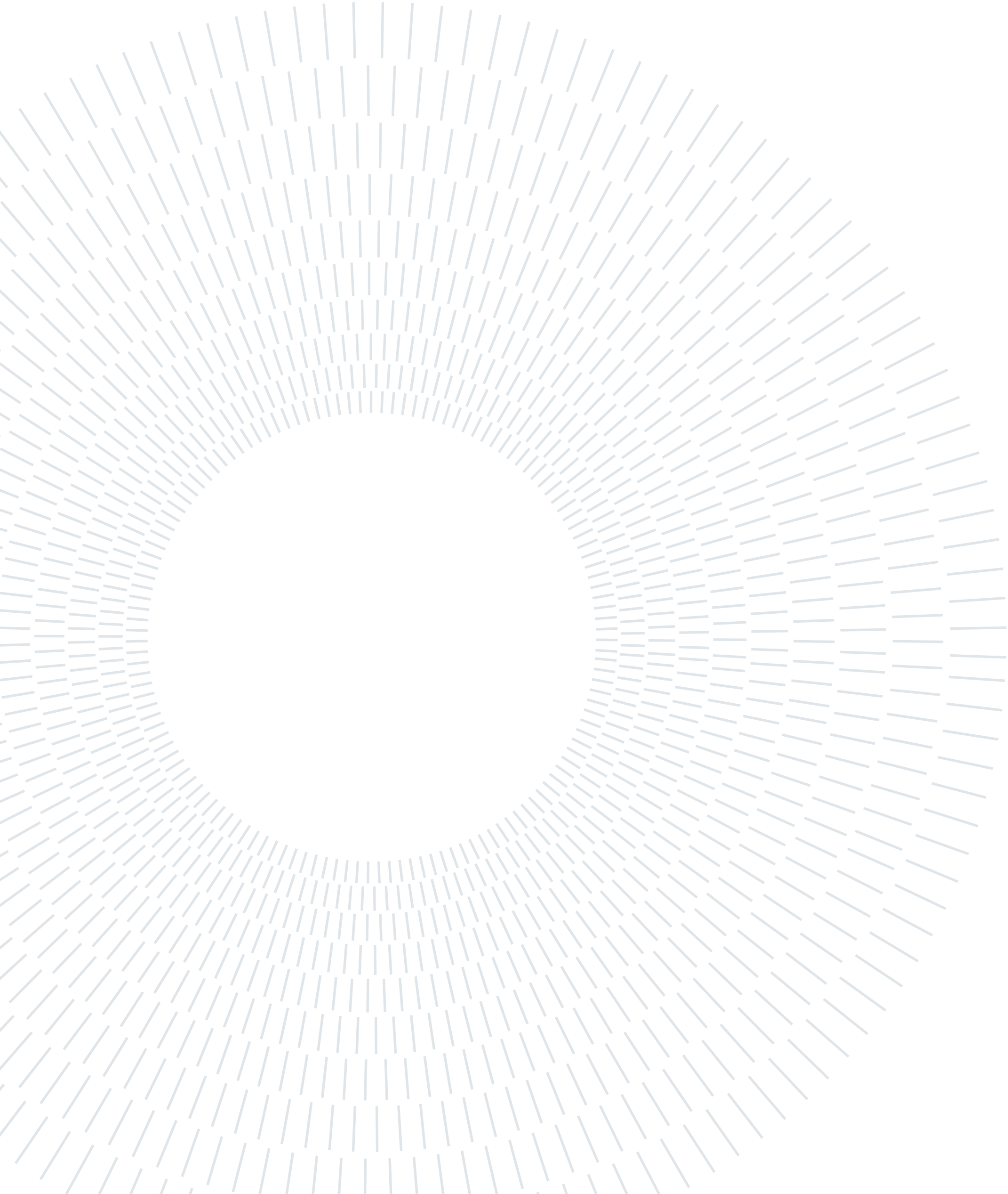
# Abstract

Through NASA's ARTEMIS program, the future of space exploration is increasingly focused on returning to the Moon, with the aim of establishing a sustained human presence in the cislunar space. One of the key components of the program will be the Lunar Gateway, a space station intended to serve as a staging point for missions to the surface of the natural satellite. Within this context, numerous rendezvous operations are expected to take place in the cislunar region, characterized by strongly nonlinear dynamics that pose significant challenges for traditional guidance and control algorithms, which typically rely on linearized models.

This thesis presents a Model Predictive Control (MPC) method based on Differential Algebra, applied to spacecraft rendezvous, aiming to effectively address the nonlinearities of the cislunar space. The proposed Differential Algebra MPC (DAMPC) is evaluated in a rendezvous scenario with a target spacecraft positioned on a Near Rectilinear Halo Orbit around the second Earth–Moon libration point, corresponding to the planned orbit of the Lunar Gateway. Three evaluation metrics are used to assess its performance: prediction error to determine model accuracy, computational time to evaluate real-time feasibility, and maneuver cost to quantify optimization effectiveness. The proposed DAMPC is then compared with a classical linear MPC across multiple initial conditions, including cases at both the aposelene and periselene.

The simulation results demonstrate that DAMPC more accurately captures the system dynamics leading to improved prediction accuracy compared to linear MPC. This enhanced fidelity comes at the cost of increased computational time but remains within a range suitable for potential online implementation. In some scenarios, the DAMPC also achieves a reduction in maneuver cost.

**Keywords:** Spacecraft Rendezvous, Model Predictive Control, Differential Algebra, Cislunar Space.



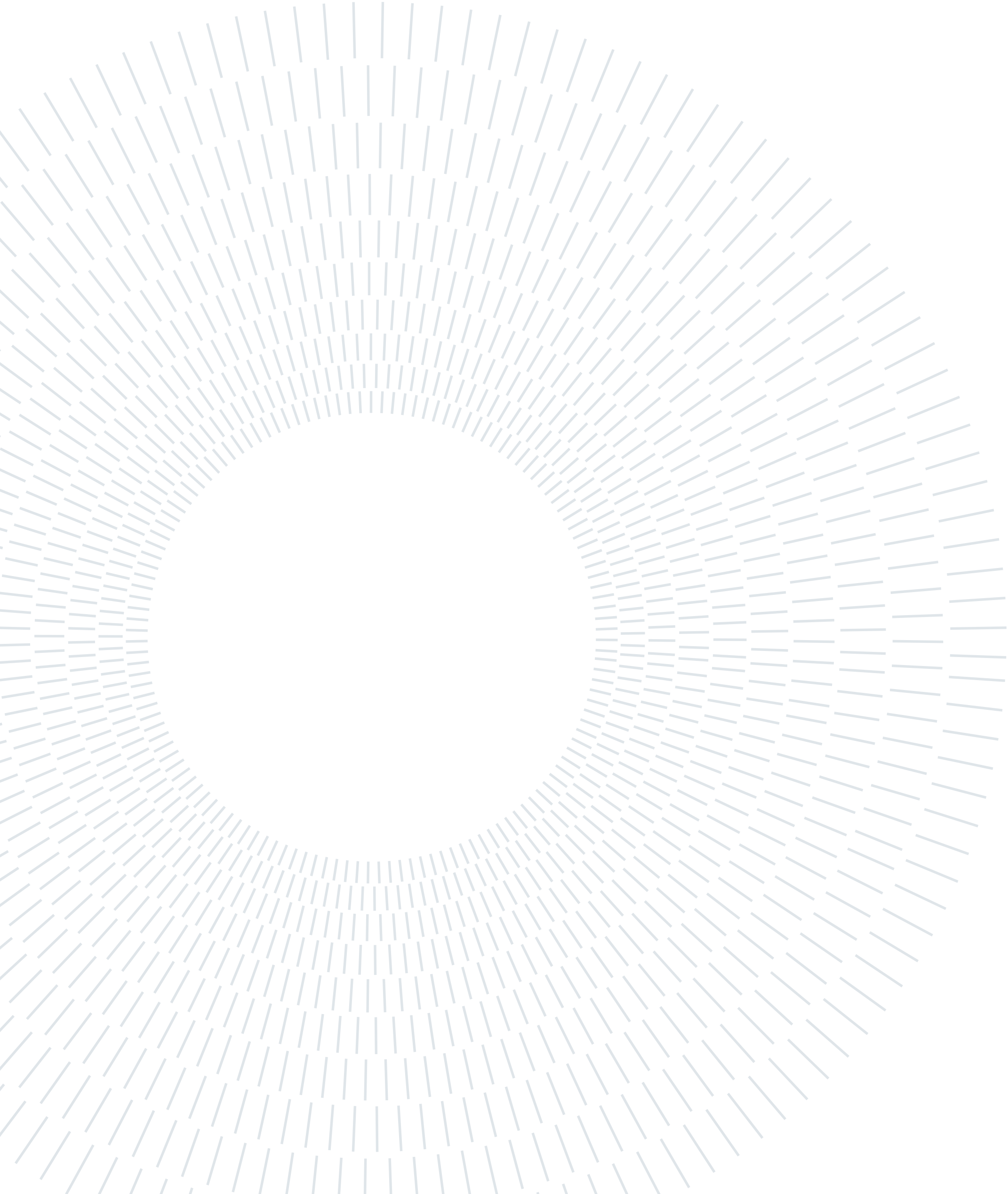
# Abstract in lingua italiana

Tramite il programma ARTEMIS della NASA, il futuro dell'esplorazione spaziale è sempre più orientato al ritorno sulla Luna, con l'obiettivo di stabilire una presenza umana duratura nello spazio cislunare. Uno degli elementi chiave del programma sarà il Lunar Gateway, una stazione spaziale concepita come punto di appoggio per le missioni dirette verso la superficie del satellite naturale. In questo contesto, numerose operazioni di rendezvous sono previste nella regione cislunare, caratterizzata da una dinamica fortemente nonlineare che pone sfide significative ai tradizionali algoritmi di guidance and control, i quali tipicamente si basano su modelli linearizzati.

Questa tesi presenta un Model Predictive Control (MPC) basato sull'Algebra Differenziale, applicato alle manovre di rendezvous tra veicoli spaziali, con l'obiettivo di affrontare efficacemente le nonlinearità dello spazio cislunare. Il Differential Albegra MPC (DAMPC) proposto viene valutato in uno scenario di rendezvous con un veicolo target posto su un'orbita di tipo Near Rectilinear Halo Orbit attorno al secondo punto lagrangiano del sistema Terra–Luna, corrispondente all'orbita pianificata per il Lunar Gateway. Tre metriche vengono considerate per valutare le prestazioni: l'errore di previsione per determinare l'accuratezza del modello, il tempo computazionale per verificarne l'applicabilità in tempo reale, e il costo della manovra per quantificare l'efficacia dell'ottimizzazione. Il DAMPC viene quindi confrontato con un classico MPC lineare, considerando diverse condizioni iniziali, sia all'aposelenio sia al periselenio.

I risultati delle simulazioni dimostrano che il DAMPC riproduce con maggiore accuratezza la dinamica del sistema, garantendo previsioni più precise rispetto all'MPC lineare. Questa maggiore fedeltà comporta un incremento del tempo computazionale, che tuttavia rimane entro valori compatibili con una potenziale implementazione in tempo reale. In alcuni scenari, il DAMPC riesce inoltre a ridurre il costo della manovra.

**Parole chiave:** Rendezvous Spaziale, Model Predictive Control, Algebra Differenziale, Spazio Cislunare.



# Contents

# Introduction

Rendezvous and Proximity Operations (RPOs) refer to the maneuvers required for a spacecraft to approach, rendezvous and operate in close proximity to another space object; when these operations involve physical connection, they are called Rendezvous and Docking/Berthing (RVD/B). RPOs play a crucial role in various space activities such as in-orbit assembly, spacecraft and space station servicing, active debris removal and retrieval of landers from planetary surfaces. The future of space exploration is closely tied to the human capability to perform these operations, which are now essential for maintaining a continuous and safe astronaut presence in space.

Through the Gemini Rendezvous Program [1, 2], the U.S. National Aeronautics and Space Administration (NASA) extensively studied the problem of RVD/B proposing several strategies and approaches. The first successful docking was performed by Neil Armstrong and Dave Scott on March 16, 1966, when Gemini VIII rendezvoused and docked with the Agena Target Vehicle, marking a pivotal moment in space exploration; just a year later, on October 30, 1967, the Soviet Union achieved the first fully automated and unmanned docking in space with Kosmos 186 and Kosmos 188. Following these successes, many other missions were accomplished, with a notable example being the Space Shuttle, which conducted 57 missions over 22 years, each involving at least one rendezvous [3]; this further increased the knowledge and the capabilities in this kind of scenario, contributing significantly to the development of another pivotal achievement: the assembly of the International Space Station (ISS).

Figure 1: Left: Gemini VIII inspects the Agena Target Vehicle. Middle: Gemini VIII closes in for the docking. Right: Gemini VIII achieves the first docking.[1]

The future of space exploration is increasingly focused on returning to the Moon, with the goal of establishing a sustained human presence in the cislunar space; this goal is being pursued by NASA through the ARTEMIS program [4], which aims to lay the foundations for long-term lunar exploration. A pivotal component of this initiative is the Lunar Gateway, a space station designed to serve as a staging point for missions to the surface of the natural satellite and, eventually, as a stepping stone for human exploration of Mars. Several orbits were considered for the Gateway, with *R. Whitley and R. Martinez* [5] performing a trade-off analysis based on the following key factors:

- **Earth Access**: the orbit should be accessible within the $\Delta v$ limits of NASA's Space Launch System (SLS) and Orion capsule.
- **Lunar Access**: the orbit should minimize $\Delta v$ requirements for lunar surface missions, ensuring efficient two-way transfers.
- **Stationkeeping Requirements**: the orbit should be as stable as possible to reduce propellant consumption by minimizing the need for frequent adjustments.
- **Communication Availability**: the orbit should ensure continuous communication with both the Earth and the lunar surface to maintain operational efficiency and safety.
- **Thermal Environment**: the orbit should maintain a thermal environment within the operational limits of the radiators installed on the station.

Based on this analysis, the authors concluded that a Near Rectilinear Halo Orbit (NRHO) offers the best balance between these factors, making it the best compromise for the Lunar Gateway. Specifically, NASA proposes the selection of an NRHO of the $L_2$ southern family with an orbital period selected for a 9:2 Lunar Synodic Resonance [6].

The resulting orbit is optimal from an overall mission perspective but presents considerable challenges in designing RVD/B operations. Due to its non-Keplerian nature, the orbit exhibits highly nonlinear dynamics, which complicates the application of classical Guidance and Control (G&C) algorithms, as they are typically designed for linearized dynamical models. *G. Bucchioni and M. Innocenti* [7] investigated the challenges associated with modeling the dynamics of an NRHO, comparing the Elliptical Restricted Three Body Problem (ERTBP) and Circular Restricted Three Body Problem (CRTBP) with a high-fidelity ephemeris propagation; their analysis highlighted that the errors introduced by the simplified models are non-negligible, particularly in the vicinity of the periselene.

---

[1] Images courtesy of NASA, retrieved from `https://www.nasa.gov/history/55-years-ago-gemini-viii-the-first-docking-in-space/`.

Moreover, *G. Franzini and M. Innocenti* [8] developed a set of linearized equations of relative motion based on the restricted three-body problem and tested them on an NRHO. The linearization proved effective at the aposelene but introduced significant errors at the periselene, where nonlinearities are more pronounced. Additionally, the authors tested the traditional Clohessy–Wiltshire equations, highlighting the limitations of the latter due to their reliance on a simplified two-body framework.

Nowadays, when designing a G&C algorithm, one of the main features pursued is autonomy; in the context of RVD/B, this approach offers notable advantages in terms of efficiency, safety, cost and operational complexity. Autonomous G&C algorithms optimize trajectories to minimize propellant consumption, while autonomous Collision Avoidance Maneuvers (CAMs) enhance safety by responding to potential hazards in real time. Costs are lowered as these systems eliminate the need for onboard human operators, reducing astronaut training expenses. Additionally, autonomous systems excel in executing complex operations, such as formation flight, where multiple constraints must be managed simultaneously. In certain scenarios, autonomy is not just beneficial but essential, particularly for deep-space missions where communication delays make real-time ground control impractical. In this regard, besides the USSR's pioneering achievement in automated docking in 1967, two other notable cases are worth mentioning: the European Space Agency's (ESA) Automated Transfer Vehicle (ATV) [9], which successfully completed five servicing missions to the ISS between 2008 and 2015, and NASA's Demonstration for Autonomous Rendezvous Technology (DART) [10], the first U.S. attempt at fully automated RVD/B performed in 2005; although DART ended in a catastrophic collision, it provided valuable lessons for future missions.

When designing an autonomous G&C algorithm, several challenges must be addressed. Four key elements need to be considered:

- **Optimality**: control inputs should be designed to minimize propellant consumption, ensuring efficient and sustainable operations.
- **Robustness**: various sources of uncertainty exist, including simplifications in the dynamical model and inaccuracies in measurements; the control algorithm should be able to generate reliable solutions despite these uncertainties.
- **Computational Efficiency**: the algorithm should be computationally efficient and generate control commands in real time, enabling its use in a closed-loop system.
- **Constraints Enforcement**: safety is one of the most critical aspects of RVD/B operations to prevent catastrophic outcomes; it is typically ensured by imposing strict constraints on the trajectory, therefore, the algorithm should be capable of

handling multiple constraints effectively.

Achieving all these presents a significant challenge. For instance, optimality and safety can be ensured by formulating and solving an Optimal Control Problem (OCP), which provides an optimal solution while handling multiple constraints; however, the main limitation of OCPs lies in their computational complexity, as these methods often struggle to converge rapidly, making real-time implementation difficult. Conversely, simple and fast control approaches can be implemented in real time, but they may lack optimality and be unable to handle multiple constraints. As is often the case in engineering, a trade-off is necessary; below is a brief overview of commonly used methods.

Guidance and control are typically seen as two different tasks; the first aims at defining the trajectory to be followed, while the second generates the control inputs required to track the prescribed path. Some algorithms are listed here [11, 12]:

- **Guidance algorithms**: Proportional Navigation (PN), Artificial Potential Field (APF), Zero-Effort-Miss/Zero-Effort-Velocity (ZEM/ZEV), Model Predictive Control (MPC).
- **Control algorithms**: $H_\infty$, State Dependent Riccati Equation (SDRE), Linear Quadratic Regulator (LQR), Sliding Mode Control (SMC), Model Predictive Control (MPC).

As already mentioned, in the context of NRHOs the primary challenge is managing highly nonlinear dynamics; some of the aforementioned methodologies are well-suited for this scenario.

**SMC** is recognized as a robust and effective nonlinear control approach. An example of its application in combination with **APF** is presented by *M. Mancini et al.* [13], where these methods are combined to perform a rendezvous while ensuring collision avoidance in the presence of obstacles; APF is particularly effective for collision avoidance, whereas SMC excels in handling nonlinear control problems, making their integration a promising strategy. However, SMC has certain limitations: it does not inherently involve an optimization process, which can lead to increased propellant consumption, and it tends to suffer from chattering.

**SDRE** is another notable control technique capable of effectively addressing nonlinearities; it is generally considered a robust method that also yields an optimal solution. An example of its application in the context of NRHOs is presented by *M. Galullo et al.* [14]. There are also more sophisticated strategies that combine the aforementioned methodologies. *Lu Cao et al.* [15] propose an **integrated approach** where SMC, APF and SDRE are combined to perform a rendezvous with collision avoidance, while achieving a certain degree of optimality due to the contribution of SDRE.

**MPC** is one of the most promising control strategies, applied in various fields such as industrial automation and aerospace systems. The core concept involves solving an OCP over a finite, receding time horizon. At each step, the control input is computed to minimize a predefined objective function while satisfying a set of constraints; only the first control input of the optimal sequence is applied, and the process is repeated at the next time step. This results in an iterative optimization process, which can be computationally intensive; nevertheless, the flexibility and constraint handling capabilities of MPC have generated significant interest within the research and engineering communities. It can be considered both a guidance and a control algorithm, as solving an OCP yields not only an optimal trajectory but also the corresponding control inputs required to follow it. Several variants of MPC exist, depending on how the system dynamics, objective function and constraints are modeled [16]. One of the first applications of this control strategy in a RVD/B scenario was proposed by *S. Di Cairano et al.* [17], where a Linear MPC (LMPC) was implemented with fixed and rotating targets; the controller accounted for various constraints, including thrust limits, velocity bounds to ensure soft docking and line-of-sight requirements that constrained the trajectory within a defined approach cone. The study demonstrated the effectiveness of the method.

When disturbances and uncertainties are relevant, satisfaction of the constraints becomes more challenging, in such cases, robust MPC formulations are more appropriate. When working with linearized dynamics in the context of an NRHO, this approach is a viable solution for dealing with modeling uncertainties; some examples are provided below. One classical robust formulation is Tube-based Robust MPC (TRMPC); *M. Mammarella et al.* [18] implemented this robust controller and compared it to a standard LMPC with a quadratic cost function. The authors showed that TRMPC consistently ensured successful mission completion, whereas LMPC failed to meet docking constraints in certain cases, however, this robustness came at the cost of increased propellant consumption; additionally, the study demonstrated that TRMPC is suitable for real-time, online application. This approach was further extended to two orbital environments: a Geostationary Transfer Orbit and an NRHO [19]. In both scenarios, TRMPC satisfied all mission requirements, whereas LMPC again failed under uncertain conditions. The main drawback of TRMPC is that the maneuver requires significantly more time and, in the case of the NRHO simulation, also demands higher fuel consumption. Another robust formulation is Chance Constrained MPC (CCMPC), which ensures constraint satisfaction with a specified probability; in the work of *Julio C. Sanchez et al.* [20], CCMPC was applied to rendezvous operations in NRHO, effectively handling disturbances, however, this approach incurred a high computational cost.

For systems where linearization is not feasible or introduces significant errors, Nonlinear MPC (NMPC) is employed; it uses the full nonlinear dynamics of the system without approximations, although it incurs high computational cost, limiting its applicability in real time. *M. Pagone et al.* [21] applied a Pontryagin-based NMPC to an NRHO rendezvous, using the Pontryagin Minimum Principle to compute optimal trajectories; although the method proved effective, the authors did not report the computational cost, leaving its practical applicability for real-time onboard optimization uncertain. To reduce the computational burden, one approach is to embed the constraints into the cost function via penalty terms, rather than modeling them as hard constraints. This strategy was successfully tested, again by *M. Pagone et al.* [22], although only in the context of an obstacle avoidance maneuver and not a complete RVD/B scenario. Nevertheless, the reported computational cost suggests that the penalty-based NMPC could be a viable approach.

MPC represents a highly promising G&C algorithm for performing RVD/B on an NRHO. However, addressing the uncertainties in the modeling of system dynamics remains a significant challenge; whether through a linearization approach combined with a robust formulation, or by directly retaining all nonlinearities using a NMPC, both methods exhibit limitations that may restrict their applicability.

**Differential Algebra (DA)** emerged from the desire to address analytical problems using algebraic methods. One of the first mathematicians to work in this direction was *J. Liouville* [23], who studied the problem of integrating functions and differential equations in finite terms, and in a more modern era, *J. F. Ritt* [24, 25] built upon these early efforts by developing a complete algebraic theory for a specific class of differential equations. Since then, many others have contributed to the development of DA; one of the most significant contributions was made by *M. Berz* [26], who applied these techniques in the field of beam physics, aiming to obtain a high-order Taylor expansion of the flow of differential equations with respect to the initial conditions. The properties of DA are nowadays exploited in various fields, including aerospace engineering; some examples of these applications are provided below.

The core idea behind DA is that functions can be described in more detail than just their values at specific points, specifically, through their full Taylor expansions around a given location, up to any desired order. In this framework, algebraic operations like addition and scalar multiplication are defined to operate on these polynomials, defining a structure which qualifies as a true algebra; furthermore, the inclusion of differentiation and integration operators transforms this algebra into a differential algebra, enabling the propagation of systems of Ordinary Differential Equations (ODEs). As a result, a dy-

namical system can be easily integrated, and the resulting flow can be represented as a simple polynomial expression. Although generating high-order polynomial expansions can be computationally demanding, especially for complex systems or higher orders, the major benefit emerges after this initial step; once the expansions are available, subsequent evaluations are significantly faster and more efficient, involving only the evaluation of polynomials. This approach can provide great benefits in those problems where iterative processes are required, such as OCPs or Monte Carlo analysis.
An interesting application of DA is provided by *P. di Lizia et al.* [27], where the authors address a Two Point Boundary Value Problem (TPBVP) using high-order expansions. The key advantage of this approach lies in its ability to generate new solutions, within a relatively large neighborhood of the reference one, through simple polynomial evaluations; the effectiveness of the method is demonstrated through its application to astrodynamics problems such as halo orbit generation and aerocapture maneuvers. The approach was later extended [28, 29] and adapted to OCPs. After computing a reference optimal trajectory and control, the solution was expanded by applying DA to the associated TPBVP; this allows the optimal solution to be efficiently recomputed for varying initial conditions through a simple evaluation of the resulting polynomial expansion. Due to its speed and low computational cost after the initial setup, the method is particularly well-suited for on-board applications.
*M. Valli et al.* [30] applied DA to enable efficient nonlinear uncertainty propagation with significantly reduced computational cost. They proposed a differential algebra-based Monte Carlo method, in which point-wise integration is replaced by evaluating a polynomial expansion obtained via DA, thereby decreasing the computational time by several orders of magnitude; the accuracy of the method can be adjusted by selecting the appropriate expansion order. Although focused on a different application, *A. Witting et al.* [31] built on the same idea of exploiting DA for propagation. They present a fast and accurate method for propagating large sets of initial conditions along with their associated probability density functions; the method is validated by propagating 10 000 random samples, completing the simulation in just a few seconds. Another interesting application is proposed by *F. Cavenago et al.* [32], where the authors applied DA for a real-time relative pose estimation and attitude prediction of a tumbling target spacecraft through a high-order numerical extended Kalman filter; the proposed filter outperforms the classical first-order counterpart and the real-time implementation is demonstrated through a hardware-in-the-loop experiment that simulates the computational limitations typically found in onboard processors.

The objective of this thesis is to propose an autonomous G&C algorithm suitable for

RVD/B operations in a highly nonlinear environment, by embedding DA into a classical MPC framework. DA is exploited to compute high-order expansions of the system flow, enabling more accurate propagation of the dynamics compared to a linearized model, while keeping the computational cost of solving the OCP within practical limits for a real-time application. The resulting **Differential Algebra MPC (DAMPC)** is evaluated against a classical LMPC in a RVD/B scenario on an NRHO, considering multiple initial conditions located in both aposelene and periselene; the target is assumed to be cooperative, and only the three translational degrees of freedom are modeled. The comparison is conducted based on three primary metrics: prediction error to determine model accuracy, computational time to evaluate real-time feasibility, and maneuver cost to quantify optimization effectiveness.

The thesis will be structured as follows:

1. **Theoretical Background**: this chapter provides the basic theoretical background required to develop the DAMPC. First, an overview of the main procedures, regions and constraints involved in a RVD/B scenario is presented. Then, the reference frames and dynamical models implemented are described to offer a comprehensive view of the dynamics governing the problem. Finally, the fundamental theory underlying both MPC and DA is outlined.

2. **Model Predictive Control**: this chapter explains how the LMPC and DAMPC have been implemented. First, the formulation of the linear controller is presented, clearly addressing the objective function, the constraints considered and the numerical optimizer used to solve the OCP. Then, the DAMPC is introduced, specifying the role of the DA and the differences with respect to the LMPC; a dedicated section details the strategy for generating the polynomial maps and selecting the expansion order.

3. **Simulations and Results**: in this chapter, the simulations are carried out and the results are presented. A preliminary section outlines the initial setup: the orbit is generated, the simulation environment is introduced, and a trade-off analysis is performed to select the expansion order. Subsequently, two sets of simulations are conducted depending on how the sampling time of the controller is managed.

# 1 | Theoretical Background

This chapter provides the basic theoretical background required to develop the DAMPC. First, an overview of the main procedures, regions and constraints involved in a RVD/B scenario is presented. Then, the reference frames and dynamical models implemented are described to offer a comprehensive view of the dynamics governing the problem. Finally, the fundamental theory underlying both MPC and DA is outlined.

## 1.1. Rendezvous and Proximity Operations

Building on the historical successes, the understanding of RVD/B has evolved, leading to the identification of distinct operational phases. According to *W. Fehse* [33], a RVD/B mission begins with the *launch*, whose objective is to position the chaser vehicle into a stable orbit within the orbital plane of the target. Then, a *phasing* maneuver is performed, whose purpose is to reduce the phase angle between the two spacecraft, typically using open-loop control since high precision is not required at this stage of the mission. The next phase is the *far-range rendezvous*, which starts when relative navigation becomes possible, usually at a distance of a few tens of kilometers, and focuses on minimizing trajectory dispersion while further closing the gap; at this stage, closed-loop control is introduced to enhance accuracy. As the distance shrinks to within a few kilometers, the *close-range rendezvous* starts; this phase is typically divided into two sub-phases: closing, a preparatory phase that brings the chaser into the final approach corridor, and final approach, whose objective is to reach the mating conditions. Finally, the *mating phase* establishes a physical connection between the two spacecraft, completing the docking or berthing process.

Each phase of the mission typically involves passing through one or more hold points, locations in space where the chaser can, in principle, remain stationary relative to the target without requiring active maneuvering; however, in practice, passive safety criteria are applied to ensure that, in the event of a malfunction, the chaser will naturally drift away from the target, minimizing the risk of collision. At each hold point, all required operations can be safely carried out before proceeding to the next mission phase; these procedures may include establishing a communication link with the target, as well as switching navigation systems or adjusting control logic to meet the evolving accuracy requirements.

There are several critical safety constraints that must be respected during RVD/B operations, which are strongly dependent on the target characteristics. In general, the approach to the target should respect several control zones standardized by the International Rendezvous System Interoperability Standards (IRSIS) [34]; four distinct regions are defined, and the chaser must receive a 'GO' signal from the ground station before entering any of them:

- *Rendezvous Sphere (RS)*: a 10 km radius sphere centered at the target.
- *Approach Sphere (AS)*: a 1 km radius sphere centered at the target.
- *Keep-out Sphere (KOS)*: a 200 m radius sphere centered at the target.
- *Approach/Departure Corridors*: conical corridors of ±10°, aligned with the docking port axis within the KOS.

Depending on the peculiar characteristics of the target, these regions can assume different shapes. In the case of an approach to the ISS, the AS, also known as the Approach Ellipsoid, is an ellipsoidal region surrounding the station, extending ±2 km along the V-bar and ±1 km along the R-bar; before entering this space, the ISS Control Center assumes full authority (the meaning of V-bar and R-bar will be clarified in Chapter 1.2). The KOS, also known as the Keep-Out Zone, on the other hand, is defined according to the IRSIS standard, and contains three approach corridors.

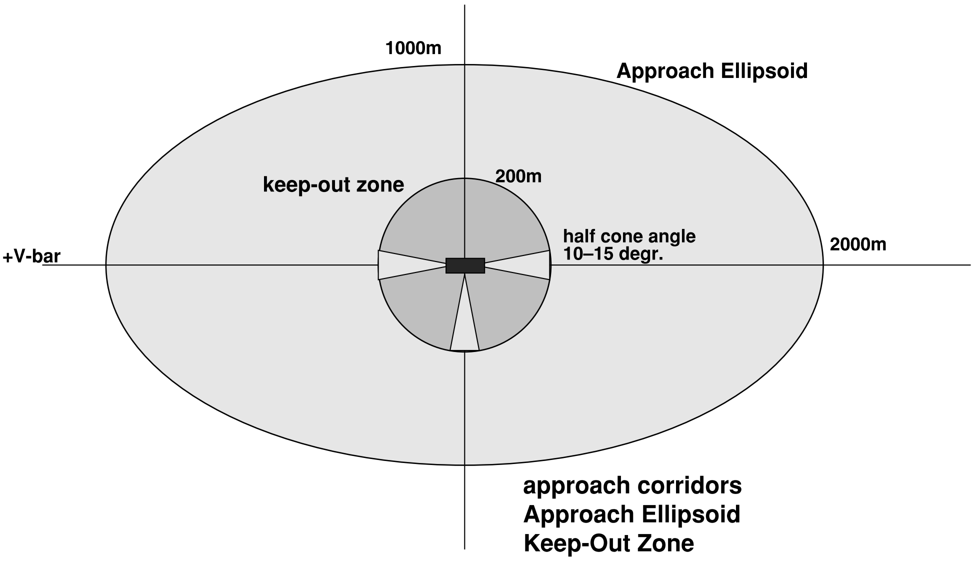


Figure 1.1: Control zones of the ISS [33].

Another standard worth mentioning is the International Docking System Standard (IDSS) [35], which establishes a standardized docking interface for the *mating phase*. Although the docking system itself is not part of the proposed thesis, the IDSS defines a set of initial conditions that must be satisfied prior to the initiation of the docking procedure. This process can be divided into two phases: the first, performed by the Soft Capture System (SCS), aims to establish an initial latching between the two vehicles; the second, carried out by the Hard Capture System (HCS), finalizes the physical connection by securing the structural interface and ensuring proper sealing. The initial contact conditions required by the SCS are summarized in Table 1.1.

| **Parameter** | **Value** |
|---|---|
| Closing (axial) rate | 0.05 to 0.10 m/s |
| Lateral (radial) rate | 0.04 m/s |
| Pitch/Yaw rate | 0.20°/s (vector sum of pitch/yaw rate) |
| Roll rate | 0.20°/s |
| Lateral (radial) misalignment | 0.10 m |
| Pitch/Yaw misalignment | 4.0°(vector sum of pitch/yaw) |
| Roll misalignment | 4.0° |

Table 1.1: Initial contact conditions required by the SCS [35].

## 1.2. Reference Frames

The previously mentioned work by *G. Franzini and M. Innocenti* [8] is used to define the reference frames and derive the dynamical models; for the full detailed procedure refer to their work, only the final results are here presented. Moreover, in the following chapters, a specific notation will be adopted to represent the derivatives. To eliminate ambiguity, each derivative will explicitly indicate the frame in which it is computed; for example, the notation $[\dot{\boldsymbol{q}}]_{\mathcal{F}}$ denotes the time derivative of the generic quantity $\boldsymbol{q}$ with respect to the generic frame $\mathcal{F}$.

First, an *inertial reference frame*, centered at point $\boldsymbol{O}$ and defined by the orthonormal unit vectors $\hat{\mathbf{I}}$, $\hat{\mathbf{J}}$ and $\hat{\mathbf{K}}$, is introduced and denoted as $\mathcal{I} : \{\boldsymbol{O}; \hat{\mathbf{I}}, \hat{\mathbf{J}}, \hat{\mathbf{K}}\}$; the classical J2000 reference frame is adopted for this purpose. The initial conditions used to compute the trajectory of the target will be extracted in this frame.

The second frame that needs to be introduced is the *synodic reference frame*, denoted as

$\mathcal{M} : \{\boldsymbol{R_m}; \hat{\boldsymbol{\imath}}_{\boldsymbol{m}}, \hat{\boldsymbol{\jmath}}_{\boldsymbol{m}}, \hat{\boldsymbol{k}}_{\boldsymbol{m}}\}$. It is centered at the Moon, whose position in $\mathcal{I}$ is given by the vector $\boldsymbol{R_m}$, and is defined by the following orthonormal basis:

$$\hat{\boldsymbol{\imath}}_{\boldsymbol{m}} = -\frac{\boldsymbol{r_{em}}}{r_{em}}, \qquad \hat{\boldsymbol{\jmath}}_{\boldsymbol{m}} = \hat{\boldsymbol{k}}_{\boldsymbol{m}} \times \hat{\boldsymbol{\imath}}_{\boldsymbol{m}}, \qquad \hat{\boldsymbol{k}}_{\boldsymbol{m}} = \frac{\boldsymbol{h_{m/e}}}{h_{m/e}}, \tag{1.1}$$

where $\boldsymbol{r_{em}}$ denotes the relative position of the Moon with respect to the Earth, and $\boldsymbol{h_{m/e}} = \boldsymbol{r_{em}} \times [\dot{\boldsymbol{r}}_{\boldsymbol{em}}]_{\mathcal{I}}$ is the corresponding specific angular momentum. In this frame, $\hat{\boldsymbol{\imath}}_{\boldsymbol{m}}$ points along the line connecting the two primaries, $\hat{\boldsymbol{k}}_{\boldsymbol{m}}$ is normal to their orbital plane, and $\hat{\boldsymbol{\jmath}}_{\boldsymbol{m}}$ completes the right-handed orthonormal triad. This frame will be used to propagate the trajectory of the target within the framework of the CRTBP.

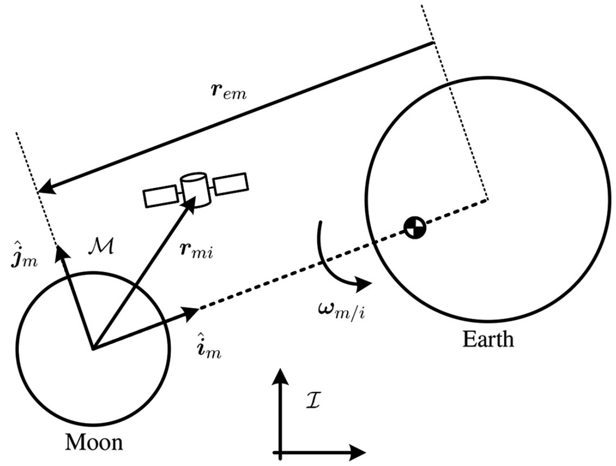


Figure 1.2: Synodic reference frame [8].

The final reference frame introduced is the *Local Vertical Local Horizontal (LVLH) reference frame*, denoted as $\mathcal{L} : \{\boldsymbol{r}; \hat{\boldsymbol{\imath}}, \hat{\boldsymbol{\jmath}}, \hat{\boldsymbol{k}}\}$. It is centered at the target, whose position relative to the Moon is given by the vector $\boldsymbol{r}$, and is defined by the following orthonormal basis:

$$\hat{\boldsymbol{\imath}} = \hat{\boldsymbol{\jmath}} \times \hat{\boldsymbol{k}}, \qquad \hat{\boldsymbol{\jmath}} = -\frac{\boldsymbol{h}}{h}, \qquad \hat{\boldsymbol{k}} = -\frac{\boldsymbol{r}}{r}, \tag{1.2}$$

where $\boldsymbol{h} = \boldsymbol{r} \times [\dot{\boldsymbol{r}}]_{\mathcal{M}}$ denotes the specific angular momentum of the target with respect to the Moon. In rendezvous literature, the unit vectors $\hat{\boldsymbol{\imath}}$, $\hat{\boldsymbol{\jmath}}$ and $\hat{\boldsymbol{k}}$ are commonly referred to as the *V-bar*, *H-bar* and *R-bar*, respectively. This frame will be employed to formulate both the nonlinear and linearized relative dynamics equations of motion and will be directly used by the controller to compute the control inputs, eliminating the need for onboard reference frame transformations.

# 1.3. Dynamical Models

## 1.3.1. Absolute Dynamics

To derive the equations of motion of a massless point subject to the gravitational attraction of two primary bodies, the Restricted Three Body Problem (RTBP) is typically employed. This model is well suited to describe the trajectory of a spacecraft of mass $m_{sc}$ under the gravitational influence of the Earth and the Moon, whose respective masses are $M_e$ and $M_m$; it is straightforward to verify that $m_{sc} \ll M_e, M_m$, thereby confirming the validity of the RTBP for the proposed scenario. Since no assumptions are introduced about the motion of the two primaries, the resulting equations of motion can be somewhat involved, requiring a careful description of the time evolution of the relative position of the two attractors. A common simplifying hypothesis is to consider the primaries to follow circular orbits; under this assumption, the distance between the two points and their angular velocity remain constant over time, which leads to a further simplification of the equations of motion and results in the CRTBP. This hypotesis is reasonable in the context of the Earth–Moon system, as the Moon follows an approximately circular orbit around the Earth. Moreover, if the distance between the two primaries is assumed to remain constant over time, the synodic reference frame $\mathcal{M}$ provides an ideal setting for representing the motion; in this rotating frame, the two planets become fixed points aligned along the $\hat{\boldsymbol{i}}_{\boldsymbol{m}}$ axis, and the motion of the spacecraft can be described and visualized more easily. For this reason, the equations of motion in the context of the CRTBP are typically derived in $\mathcal{M}$, as will be done in the present work. A final step typically involves deriving the equations of motion in their dimensionless form, in order to mitigate numerical issues that may arise when working with quantities of vastly different orders of magnitude. Distances are expressed in units of the Moon's orbit semi-major axis, the time unit is defined as the inverse of the Earth-Moon mean angular velocity and the mass unit is defined as the sum of the masses of the two primaries; finally the mass ratio parameter $\mu$ is defined as:

$$\mu = \frac{\mu_m}{\mu_e + \mu_m} = \frac{M_m}{M_e + M_m} \tag{1.3}$$

where $\mu_e$ and $\mu_m$ are the gravitational parameters of the Earth and the Moon. After normalization, the total mass is scaled such that $M_e + M_m = 1$, consequently, the gravitational parameters become $\mu_e = 1 - \mu$ and $\mu_m = \mu$. The value of $\mu$, along with the dimensional units used in the simulation, is reported in Table 1.2.

| Parameter | Value |
|---|---|
| $\mu$ | $1.21530 \cdot 10^{-2}$ |
| Distance Unit (DU) | $3.84400 \cdot 10^{8}$ m |
| Time Unit (TU) | $3.75699 \cdot 10^{5}$ s |
| Mass Unit (MU) | $6.04709 \cdot 10^{24}$ kg |

Table 1.2: Dimensional unit for the Earth-Moon system [36].

Finally, the dimensionless equations of motion for the CRTBP, expressed in $\mathcal{M}$, are given by:

$$\begin{cases} \ddot{x} = 2\,\dot{y} + x - \mu\dfrac{x}{r_m^3} - (1-\mu)\left(\dfrac{x-1}{r_e^3} + 1\right) \\ \ddot{y} = -2\,\dot{x} + y - \mu\dfrac{y}{r_m^3} - (1-\mu)\dfrac{y}{r_e^3} \\ \ddot{z} = -\mu\dfrac{z}{r_m^3} - (1-\mu)\dfrac{z}{r_e^3} \end{cases} \tag{1.4}$$

with $r_e$ and $r_m$ representing the distance of the spacecraft from the Earth and the Moon, computed as:

$$r_e = \sqrt{(x-1)^2 + y^2 + z^2}, \qquad r_m = \sqrt{x^2 + y^2 + z^2} \tag{1.5}$$

### 1.3.2. Relative Dynamics

To present the equations of relative motion, some details on the notation must first be provided. The position vectors of the target and the chaser with respect to the Moon are denoted by $\boldsymbol{r}$ and $\boldsymbol{r_c}$, respectively, while the relative position of the chaser with respect to the target is defined as $\boldsymbol{\rho} = \boldsymbol{r_c} - \boldsymbol{r}$. A graphical representation of these quantities is provided in Figure 1.3.

The formulation of the equations of motion requires the *skew symmetric matrices* associated with the angular velocity and angular acceleration of $\mathcal{L}$ with respect to $\mathcal{I}$, namely $\boldsymbol{\omega_{l/i}}$ and $[\dot{\boldsymbol{\omega}}_{l/i}]_{\mathcal{L}}$:

$$\boldsymbol{\Omega}_{l/i} = \begin{bmatrix} 0 & -\omega^z_{l/i} & \omega^y_{l/i} \\ \omega^z_{l/i} & 0 & -\omega^x_{l/i} \\ -\omega^y_{l/i} & \omega^x_{l/i} & 0 \end{bmatrix}, \qquad [\dot{\boldsymbol{\Omega}}_{l/i}]_{\mathcal{L}} = \begin{bmatrix} 0 & -\dot{\omega}^z_{l/i} & \dot{\omega}^y_{l/i} \\ \dot{\omega}^z_{l/i} & 0 & -\dot{\omega}^x_{l/i} \\ -\dot{\omega}^y_{l/i} & \dot{\omega}^x_{l/i} & 0 \end{bmatrix} \tag{1.6}$$

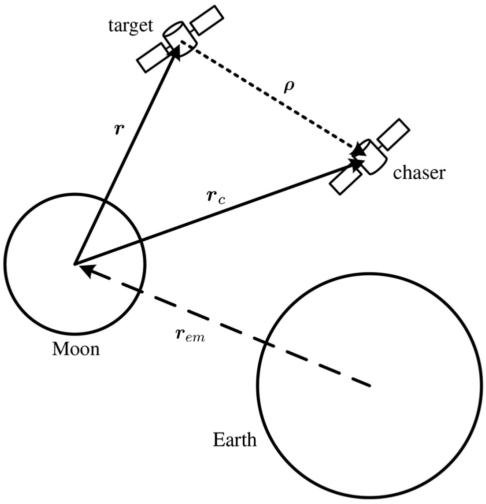


Figure 1.3: Target and chaser in the three-body system [8].

The angular velocity can be computed as the composition of the angular velocity vectors:

$$\boldsymbol{\omega}_{l/i} = \boldsymbol{\omega}_{l/m} + \boldsymbol{\omega}_{m/i} \tag{1.7}$$

where $\boldsymbol{\omega}_{l/m}$ and $\boldsymbol{\omega}_{m/i}$ are the angular velocities of $\mathcal{L}$ with respect to $\mathcal{M}$, and of $\mathcal{M}$ with respect to $\mathcal{I}$, respectively. The angular acceleration is obtained by differentiation:

$$[\dot{\boldsymbol{\omega}}_{l/i}]_{\mathcal{L}} = [\dot{\boldsymbol{\omega}}_{l/m}]_{\mathcal{L}} + [\dot{\boldsymbol{\omega}}_{m/i}]_{\mathcal{M}} - \boldsymbol{\omega}_{l/m} \times \boldsymbol{\omega}_{m/i} \tag{1.8}$$

Under the assumption that the primaries revolve in circular orbits, the equations are specialized for the context of the CRTBP. This hypothesis leads to the following simplifications:

$$\boldsymbol{r}_{em} = -\hat{\boldsymbol{i}}_m, \qquad [\dot{\boldsymbol{r}}_{em}]_{\mathcal{M}} = \boldsymbol{0}, \qquad \boldsymbol{\omega}_{m/i} = \hat{\boldsymbol{k}}_m, \qquad [\dot{\boldsymbol{\omega}}_{m/i}]_{\mathcal{M}} = [\ddot{\boldsymbol{\omega}}_{m/i}]_{\mathcal{M}} = \boldsymbol{0}. \tag{1.9}$$

and the angular velocity and acceleration involved in the computation of the skew symmetric matrices are reduced to:

$$\boldsymbol{\omega}_{l/i} = \boldsymbol{\omega}_{l/m} + \hat{\boldsymbol{k}}_m, \qquad [\dot{\boldsymbol{\omega}}_{l/i}]_{\mathcal{L}} = [\dot{\boldsymbol{\omega}}_{l/m}]_{\mathcal{L}} - \boldsymbol{\omega}_{l/m} \times \hat{\boldsymbol{k}}_m \tag{1.10}$$

The computation of the angular velocity $\boldsymbol{\omega}_{l/m}$ and acceleration $[\dot{\boldsymbol{\omega}}_{l/m}]_{\mathcal{L}}$ is quite involved, the reader can refer to Appendix A.1 for the full equations.

With these preliminaries established, the Circular Nonlinear Equations of Relative Motion

(CNERM) expressed in $\mathcal{L}$ are defined as follows:

$$\begin{aligned}[\ddot{\boldsymbol{\rho}}]_{\mathcal{L}} = &- 2\,\boldsymbol{\Omega}_{l/i}\,[\dot{\boldsymbol{\rho}}]_{\mathcal{L}} - ([\dot{\boldsymbol{\Omega}}_{l/i}]_{\mathcal{L}} + \boldsymbol{\Omega}_{l/i}^2)\,\boldsymbol{\rho} + \mu\left(\frac{\boldsymbol{r}}{r^3} - \frac{\boldsymbol{r}+\boldsymbol{\rho}}{\|\boldsymbol{r}+\boldsymbol{\rho}\|^3}\right)\\ &+ (1-\mu)\left(\frac{\boldsymbol{r}+\boldsymbol{r}_{em}}{\|\boldsymbol{r}+\boldsymbol{r}_{em}\|^3} - \frac{\boldsymbol{r}+\boldsymbol{r}_{em}+\boldsymbol{\rho}}{\|\boldsymbol{r}+\boldsymbol{r}_{em}+\boldsymbol{\rho}\|^3}\right)\end{aligned} \tag{1.11}$$

If the chaser acceleration is controllable by means of the control vector $\boldsymbol{u}(t)$, Equation 1.11 can be written as a nonlinear system affine in the control:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x},t) + \tilde{\boldsymbol{B}}\boldsymbol{u}, \qquad \boldsymbol{x} = \begin{bmatrix}\boldsymbol{\rho}\\ [\dot{\boldsymbol{\rho}}]_{\mathcal{L}}\end{bmatrix}, \qquad \tilde{\boldsymbol{B}} = \begin{bmatrix}\boldsymbol{0}_{3\times3}\\ \boldsymbol{I}_3\end{bmatrix}, \tag{1.12}$$

where $\boldsymbol{f}(\boldsymbol{x},t)$ compactly represents the right-hand side of Equation 1.11, $\tilde{\boldsymbol{B}}$ is the control input matrix, and $\boldsymbol{I}_3$ denotes the $3\times3$ identity matrix.

### 1.3.3. Linearized Relative Dynamics

The CNERM described in Equation 1.11 can be linearized to obtain the Circular Linear Equations of Relative Motion (CLERM). The key idea is to consider the gravitational acceleration acting on the chaser due to the two primaries and to perform a first-order Taylor expansion of these accelerations around the target position. As a result, the system dynamics can be reduced to a Linear Time Variant (LTV) system as follows:

$$\dot{\boldsymbol{x}} = \tilde{\boldsymbol{A}}(t)\boldsymbol{x} + \tilde{\boldsymbol{B}}\boldsymbol{u}, \qquad \tilde{\boldsymbol{A}}(t) = \begin{bmatrix}\boldsymbol{0}_{3\times3} & \boldsymbol{I}_3\\ \boldsymbol{A}_{\dot{\rho}\rho}(t) & -2\,\boldsymbol{\Omega}_{l/i}(t)\end{bmatrix} \tag{1.13}$$

with:

$$\begin{aligned}\boldsymbol{A}_{\dot{\rho}\rho}(t) = &- [\dot{\boldsymbol{\Omega}}_{l/i}]_{\mathcal{L}} - \boldsymbol{\Omega}_{l/i}^2 - \frac{\mu}{r^3}\left(\boldsymbol{I} - 3\frac{\boldsymbol{r}\boldsymbol{r}^T}{r^2}\right)\\ &- \frac{1-\mu}{\|\boldsymbol{r}+\boldsymbol{r}_{em}\|^3}\left(\boldsymbol{I} - 3\frac{(\boldsymbol{r}+\boldsymbol{r}_{em})(\boldsymbol{r}+\boldsymbol{r}_{em})^T}{\|\boldsymbol{r}+\boldsymbol{r}_{em}\|^2}\right)\end{aligned} \tag{1.14}$$

To implement the linearized dynamics within an MPC framework, it is necessary to discretize the system; let $T_s$ denote the sampling interval, and $q_k$ the value of a generic quantity $q$ at time instant $t_k$. The discrete-time dynamics then take the form:

$$\boldsymbol{x}_{k+1} = \boldsymbol{A}_k\boldsymbol{x}_k + \boldsymbol{B}_k\boldsymbol{u}_k \tag{1.15}$$

A widely used technique for deriving the matrices $\boldsymbol{A}_k$ and $\boldsymbol{B}_k$ is the Zero Order Hold (ZOH), which assumes the dynamics and the control inputs to remain constant between two successive sampling instants; under this assumption, the discretized matrices are obtained as:

$$\boldsymbol{A}_k = e^{\tilde{\boldsymbol{A}} T_s}, \qquad \boldsymbol{B}_k = \int_0^{T_s} e^{\tilde{\boldsymbol{A}} \tau} \tilde{\boldsymbol{B}} \, d\tau \tag{1.16}$$

When the system is a Linear Time Invariant (LTI) and the control input is piecewise constant, ZOH provides an exact discretization. In the context of MPC, the piecewise-constant input assumption is valid, however, the linearized model introduced here cannot be strictly represented by an LTI system due to its time-varying dynamics; still, if the time variation in system dynamics is slow and the sampling interval is sufficiently small, the system can be approximated as an LTI within each sampling period making the ZOH an effective method for discretization.

## 1.4. Model Predictive Control

The following section draws heavily from the exposition provided by *A. Botelho et al.* [37]; for further details, refer to their book.

MPC is a control technique based on iterative online optimization, where future system states are predicted over a finite prediction horizon. At each discrete time step, the controller formulates and solves an OCP using the most recent state measurement, or estimate, as the initial condition for the prediction. Solving the OCP yields a sequence of control inputs over the prediction horizon; however, only the first control input from this sequence is applied to the plant, and the optimization is repeated at the next time step with the updated state and a horizon shifted forward. This continuous shifting of the prediction window is the reason why the MPC is also commonly referred to as Receding-Horizon Control or Moving-Horizon Control. A graphical representation of this strategy is given in Figure 1.4.

Unlike classical control methods, MPC is based on solving an OCP, which allows it to explicitly handle complex constraints on both states and inputs while optimizing the performance; this capability makes MPC particularly effective in scenarios where maintaining safety or meeting operational limits is critical, such as in spacecraft RVD/B or other high-precision, constrained maneuvering tasks. Its main drawback is the computational effort required to solve the OCP in real time; however, advancements in computational resources and clever problem formulations help to keep the computational cost manageable, making MPC one of the most promising strategies for complex, constrained control problems.

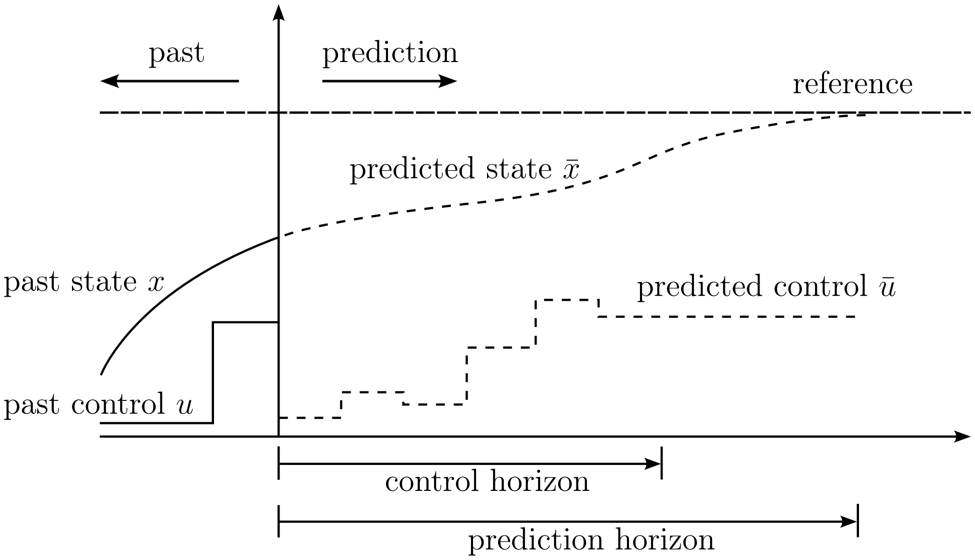


Figure 1.4: Model Predictive Control strategy [37].

Assume that an estimate of the current state, denoted by $\boldsymbol{x}(t_k)$, is available at time step $t_k$. Let $N$ be the number of time steps representing the prediction horizon over which the optimization is carried out, and denote by $\bar{\boldsymbol{x}}_i$ and $\bar{\boldsymbol{u}}_i$ the predicted state and control inputs at a given time instant $t_{k+i}$. The classical formulation for an MPC is then given by:

$$\min_{\substack{\bar{\boldsymbol{x}}_0,\dots,\bar{\boldsymbol{x}}_N \\ \bar{\boldsymbol{u}}_0,\dots,\bar{\boldsymbol{u}}_{N-1}}} \quad J = \sum_{i=0}^{N-1} \ell(\bar{\boldsymbol{x}}_i, \bar{\boldsymbol{u}}_i, t_{k+i}) + J_N(\bar{\boldsymbol{x}}_N, t_{k+N}) \tag{1.17a}$$

$$\text{s.t.} \quad \bar{\boldsymbol{x}}_0 = \boldsymbol{x}(t_k) \tag{1.17b}$$

$$\bar{\boldsymbol{x}}_{i+1} = f(\bar{\boldsymbol{x}}_i, \bar{\boldsymbol{u}}_i, t_{k+i}) \qquad \forall i \in 0, \dots, N-1 \tag{1.17c}$$

$$\bar{\boldsymbol{x}}_i \in \mathcal{X}_i \qquad \forall i \in 0, \dots, N \tag{1.17d}$$

$$\bar{\boldsymbol{u}}_i \in \mathcal{U}_i \qquad \forall i \in 0, \dots, N-1 \tag{1.17e}$$

Equation 1.17a defines the cost function, which is the quantity to be minimized; it consists of two components: the stage cost $\ell(\bar{\boldsymbol{x}}_i, \bar{\boldsymbol{u}}_i, t_{k+i})$, also known as the Lagrangian cost, and the terminal cost $J_N(\bar{\boldsymbol{x}}_N, t_{k+N})$, also referred to as the Mayer cost. Equation 1.17b enforces the initial conditions, ensuring that the predicted trajectory starts from the estimated state, while Equation 1.17c imposes the system dynamics over the prediction horizon. Finally, Equation 1.17d and Equation 1.17e represent state and input constraints, respectively, where $\mathcal{X}_i$ and $\mathcal{U}_i$ denote the admissible sets at each time step.

The behavior and computational cost of the controller are strongly influenced by the formulation of the cost function, the system dynamics and the constraints. In particular, the cost function should ideally be designed to ensure convexity, thereby guaranteeing the existence of a unique global minimum; this prevents the optimization algorithm from getting trapped in local minima, which could lead to suboptimal performance or increased computational effort. A common approach to achieve this is to employ a quadratic cost function, which not only promotes convexity but also typically results in smooth closed-loop transients and exhibits desirable robustness properties [16]:

$$\begin{aligned} &\ell(\bar{\boldsymbol{x}}_i, \bar{\boldsymbol{u}}_i, t_{k+i}) = \|\bar{\boldsymbol{x}}_i - \boldsymbol{x}_{ref}\|^2_{\boldsymbol{Q}} + \|\bar{\boldsymbol{u}}_i - \boldsymbol{u}_{ref}\|^2_{\boldsymbol{R}} \qquad \forall i \in 0, \dots, N-1 \\ &J_N(\bar{\boldsymbol{x}}_N, t_{k+N}) = \|\bar{\boldsymbol{x}}_N - \boldsymbol{x}_{ref}\|^2_{\boldsymbol{Q_f}} \end{aligned} \tag{1.18}$$

where $\boldsymbol{Q}$ and $\boldsymbol{Q_f}$ are positive semidefinite matrices, while $\boldsymbol{R}$ is positive definite; the terms $\boldsymbol{x}_{ref}$ and $\boldsymbol{u}_{ref}$ represent the reference signals to be tracked, if the target is at the origin and no control reference is provided, they are simply set to zero. An alternative formulation is the $\ell_1$-Norm cost function, which tends to produce sparser control inputs, often resulting in a bang-bang behaviour. A further extension is the *LASSO* cost function, which combines the two previously discussed solutions, effectively merging the sparsity of the $\ell_1$-Norm with the smoothness and robustness of a quadratic cost.

The next step is to define how the dynamics and constraints are treated. In general, linearity enables faster optimization, making online implementation more practical. A classical formulation for the Linear MPC relies on Quadratic Programming (QP), which requires a quadratic cost function, along with linear system dynamics and constraints. This allows for very fast computation, and highly efficient solvers exist for this class of OCPs. However, in many practical applications, both the system dynamics and constraints are inherently nonlinear, making linearization necessary. This introduces a trade-off between model accuracy and computational efficiency: one must assess whether the linear approximation is sufficiently accurate to justify a linear formulation, or other strategies should be employed.

Nonlinear MPC is designed to handle systems governed by nonlinear dynamics without relying on linearization. By incorporating a nonlinear prediction model, often implemented as a numerical integrator, NMPC enables more accurate predictions and allows operation closer to the boundaries of the feasible region. However, this improved model fidelity comes at the cost of increased computational complexity, as NMPC requires solving nonlinear and non-convex optimization problems.

An effective and straightforward approach for reducing the computational cost is known as *move blocking*. The core idea is to predict $N$ future states, but only optimize the first $M$ control inputs while the remaining $N-M$ can be determined using various alternative strategies, such as setting them to zero, assigning them the value of the last control available, or employing more sophisticated methods. This approach leads to faster optimizations since fewer decision variables are involved; however, it is important to note that this reduction in computational effort generally results in a loss of optimality, as fewer control inputs are actively optimized. If the remaining $N-M$ control inputs are set to zero and the cost function is quadratic, the stage cost can be divided into two contributions:

$$\begin{aligned} \ell_1(\bar{\boldsymbol{x}}_i, \bar{\boldsymbol{u}}_i, t_{k+i}) &= \|\bar{\boldsymbol{x}}_i - \boldsymbol{x}_{ref}\|^2_{\boldsymbol{Q}} + \|\bar{\boldsymbol{u}}_i - \boldsymbol{u}_{ref}\|^2_{\boldsymbol{R}} & \quad \forall i \in 0, \dots, M-1 \\ \ell_2(\bar{\boldsymbol{x}}_i, \boldsymbol{0}, t_{k+i}) &= \|\bar{\boldsymbol{x}}_i - \boldsymbol{x}_{ref}\|^2_{\boldsymbol{Q}} & \quad \forall i \in M, \dots, N-1 \end{aligned} \tag{1.19}$$

Regarding stability, according to standard theory [38], the key idea is to define the terminal cost function $J_N$ such that the overall cost function $J$ serves as a Lyapunov function $V$, therefore $J$ is often denoted as $V$. This objective can be achieved in several ways, but the most common approach in the context of a QP problem is to define the terminal cost using the solution of the algebraic Riccati equation, so the terminal weight $\boldsymbol{Q_f}$ is often denoted as $\boldsymbol{P}$; additionally a positively invariant set can be introduced as a terminal constraint. When dealing with NMPC, ensuring stability is more complex and requires a more careful and specific formulation.

## 1.5. Differential Algebra

The following section is largely based on the theoretical framework presented by *M. Berz* [39], and will outline how Differential Algebra is used to compute the $n^{\text{th}}$-order Taylor expansions of the flow of a set of ODEs with respect to the initial conditions.

Traditional numerical methods are designed to work with numbers, treating functions as mere tools to extract values at specific points. In contrast, DA offers a fundamentally different perspective by recognizing that functions inherently contain more information than just their point-wise evaluations. The underlying principle behind DA is to enable operations on functions within a computer environment in a manner analogous to how real numbers are handled through Floating-Point (FP) arithmetic. Since computers can store only a finite number of digits, real numbers must be approximated by FP numbers, which provide a practical means for performing numerical computations while managing the truncation errors; for this approach to work correctly, every operator defined in one domain must have a counterpart in the other domain, and their application must commute. To illustrate this concept, consider two real numbers $a$ and $b$, and a transformation $\mathcal{T}^{FP}$ that maps them to their FP representations, $\bar{a}$ and $\bar{b}$. Given a generic operator $*$ defined in the real domain, an associated adjoint operation $\circledast$ is defined in the FP domain such that the diagram in Figure 1.5 commutes. In practice, this means that when the real numbers $a$ and $b$ are converted into their FP representations and an operation is performed within the FP domain, the result matches what would be obtained by first performing the operation in the real domain and then converting the result to its FP counterpart.

$$\begin{tikzcd}
a, b \in \mathbb{R} \arrow[r, "\mathcal{T}^{FP}"] \arrow[d, "*"'] & \bar{a}, \bar{b} \in \mathbb{FP} \arrow[d, "\circledast"] \\
a * b \in \mathbb{R} \arrow[r, "\mathcal{T}^{FP}"] & \bar{a} \circledast \bar{b} \in \mathbb{FP}
\end{tikzcd}$$

Figure 1.5: Floating-Point equivalence diagram (adapted from [39]).

A similar approach is adopted in DA. Just as real numbers cannot be directly manipulated in a computer and must be approximated using FP arithmetic, mathematical functions also need to be represented in a form more suitable for digital representation and manipulation. To this end, functions are approximated by their Taylor expansions, resulting in polynomial representations that are easier to manipulate due to their algebraic simplicity; DA provides the algebraic framework to operate on these polynomials in a systematic and structured way. Consequently, the diagram shown in Figure 1.5 can be mirrored in the context of DA, leading to the diagram in Figure 1.6. Let $f$ and $g$ be two sufficiently differentiable functions, and let $\mathcal{T}^{DA}$ be the transformation that maps them to their corresponding $n^{\text{th}}$-order Taylor expansions, $F$ and $G$; note that the space of Taylor polynomials is often denoted ${}_nD_v$, with $n$ representing the expansion order while $v$ the number of variables. For a generic operator $*$ defined in the function space, an adjoint operator $\circledast$ is defined in the space of Taylor polynomials such that the diagram commutes. This ensures that operating on $F$ and $G$ with $\circledast$ yields the same result as applying $*$ to $f$ and $g$, and then extracting the Taylor expansion of the results.

$$\begin{tikzcd}
f, g \arrow[r, "\mathcal{T}^{DA}"] \arrow[d, "*"'] & F, G \arrow[d, "\circledast"] \\
f * g \arrow[r, "\mathcal{T}^{DA}"] & F \circledast G
\end{tikzcd}$$

Figure 1.6: Differential Algebra equivalence diagram (adapted from [39]).

Based on this concept, classical operators such as addition, multiplication and scalar multiplication are defined, giving rise to an algebra known as Truncated Power Series Algebra (TPSA). Eventually, differential and integral operators can also be introduced, resulting in a Differential Algebra and completing the DA framework.

### 1.5.1. The Minimal Differential Algebra ${}_1D_1$

As a first step in defining Differential Algebra, the fundamental operations of addition, scalar multiplication and vector multiplication can be introduced. Consider the set of ordered pairs $(q_0, q_1)$, with $q_0$ and $q_1$ real numbers; the following operations are defined:

$$\begin{aligned} (q_0, q_1) + (r_0, r_1) &= (q_0 + r_0, q_1 + r_1) \\ t \cdot (q_0, q_1) &= (t \cdot q_0, t \cdot q_1) \\ (q_0, q_1) \cdot (r_0, r_1) &= (q_0 \cdot r_0, q_0 \cdot r_1 + q_1 \cdot r_0) \end{aligned} \tag{1.20}$$

The set of ordered pairs, equipped with these operations, is denoted as ${}_1D_1$. This structure already satisfies the properties of an algebra: multiplication is commutative, associative and distributive with respect to addition, and the element $(1, 0)$ acts as the multiplicative identity. Moreover, any real number $q$ can be represented as $(q, 0)$; thus, similarly to how complex numbers extend the real numbers, ${}_1D_1$ constitutes an extension of $\mathbb{R}$, since it holds that $(q, 0) + (r, 0) = (q + r, 0)$ and $(q, 0) \cdot (r, 0) = (q \cdot r, 0)$. This is the only similarity shared with complex numbers; indeed, a fundamental distinction arises in the behavior of the quantity $(0, 1)$, which in $\mathbb{C}$ corresponds to the imaginary unit. In $\mathbb{C}$, the square of the imaginary unit yields $-1$, meaning that $(0, 1) \cdot (0, 1) = (-1, 0)$; in contrast, in ${}_1D_1$ this is not the case, using the definition of multiplication from Equation 1.20, the following is obtained: $(0, 1) \cdot (0, 1) = (0, 0)$. The element $(0, 1)$ plays a central role in DA and is denoted by $d$; its square being zero is a peculiar property that will be further explored and generalized later.

The set ${}_1D_1$ also admits a multiplicative inverse and a square root:

$$\begin{aligned} (q_0, q_1)^{-1} &= \left( \frac{1}{q_0}, -\frac{q_1}{q_0^2} \right) \qquad \forall\ q_0 \neq 0 \\ \sqrt{(q_0, q_1)} &= \left( \sqrt{q_0}, \frac{q_1}{2\sqrt{q_0}} \right) \qquad \forall\ q_0 > 0 \end{aligned} \tag{1.21}$$

Another key difference compared to complex algebra is that the structure ${}_1D_1$ can be endowed with an order relation that is compatible with its algebraic operations. Specifically, the following can be defined:

$$\begin{aligned} (q_0, q_1) < (r_0, r_1) &\quad \text{if} \quad q_0 < r_0 \vee (q_0 = r_0 \wedge q_1 < r_1) \\ (q_0, q_1) > (r_0, r_1) &\quad \text{if} \quad (r_0, r_1) < (q_0, q_1) \\ (q_0, q_1) = (r_0, r_1) &\quad \text{if} \quad q_0 = r_0 \wedge q_1 = r_1 \end{aligned} \tag{1.22}$$

From this order definition it follows that for every couple $(q_0, q_1)$ and $(r_0, r_1)$ only one of the previous definitions holds, hence, the order is said to be total. Moreover, compatibility of the order with addition and multiplication means that for all $(q_0, q_1)$, $(r_0, r_1)$ and $(s_0, s_1)$ in ${}_1D_1$, the following properties hold:

$$\begin{aligned} &\text{if } (q_0, q_1) < (r_0, r_1) && \Rightarrow (q_0, q_1) + (s_0, s_1) < (r_0, r_1) + (s_0, s_1) \\ &\text{if } (q_0, q_1) < (r_0, r_1) \wedge (s_0, s_1) > (0, 0) && \Rightarrow (q_0, q_1) \cdot (s_0, s_1) < (r_0, r_1) \cdot (s_0, s_1) \end{aligned} \tag{1.23}$$

At this point, having defined the order, it is possible to extend the properties of $d = (0, 1)$. As previously mentioned, its square is zero; furthermore, by exploiting the relation in Equation 1.22, it has the special property of being positive yet smaller than any real number $r$; that is, the inequality chain $(0, 0) < (0, 1) < (r, 0)$ holds. This implies that $d$ is infinitely small and therefore is called an *infinitesimal* or *differential*. Consequently, any generic pair $(q_0, q_1) \in {}_1D_1$ can be represented as $(q_0, 0) + d \cdot (0, q_1) = q_0 + d \cdot q_1$, where the first term is the *real part* and the second is the *differential part*.

A map $\partial$ from ${}_1D_1$ into itself can be introduced, which will act as a derivation and will endow the algebra ${}_1D_1$ with the structure of a differential algebra. The map $\partial : {}_1D_1 \to {}_1D_1$ is defined as:

$$\partial(q_0, q_1) = (0, q_1) \tag{1.24}$$

And it can be verified that:

$$\begin{aligned} \partial\{(q_0, q_1) + (r_0, r_1)\} &= \partial(q_0 + r_0, q_1 + r_1) = (0, q_1 + r_1) \\ &= (0, q_1) + (0, r_1) = \partial(q_0, q_1) + \partial(r_0, r_1) \end{aligned} \tag{1.25}$$

and

$$\begin{aligned} \partial\{(q_0, q_1) \cdot (r_0, r_1)\} &= \partial(q_0 \cdot r_0, q_0 \cdot r_1 + q_1 \cdot r_0) = (0, q_0 \cdot r_1 + q_1 \cdot r_0) \\ &= (0, q_1) \cdot (r_0, r_1) + (q_0, q_1) \cdot (0, r_1) \\ &= \{\partial(q_0, q_1)\} \cdot (r_0, r_1) + (q_0, q_1) \cdot \{\partial(r_0, r_1)\} \end{aligned} \tag{1.26}$$

These properties hold for all $(q_0, q_1), (r_0, r_1) \in {}_1D_1$. Therefore, $\partial$ satisfies the conditions of a derivation, and the pair $({}_1D_1, \partial)$ defines a differential algebra.

At this point, the most important properties of $({}_1D_1)$ can be introduced: DA enables automatic computation of derivatives. Assume that the values and derivatives of two functions, $f$ and $g$, are known at the origin; these can be organized in $({}_1D_1)$ by introducing the operator $[\,]$, defined as follows:

$$[f] = (f(0), f'(0)), \qquad [g] = (g(0), g'(0)) \tag{1.27}$$

It is straightforward to verify that the sum of $[f]$ and $[g]$ results in an element of ${}_1D_1$ whose real component is the sum of the function values, and whose differential component is the derivative of the sum; the same holds for the product. This enables automatic derivative computation, which is achieved by applying the basic algebraic operations defined in Equation 1.20; therefore, when the derivative of a sum or a product is required, the expressions in Equation 1.28 can be applied, the differential part will contain the desired result.

$$[f+g] = [f] + [g], \qquad [f \cdot g] = [f] \cdot [g] \tag{1.28}$$

### 1.5.2. The Differential Algebra ${}_nD_v$

The differential algebra ${}_1D_1$ can be extended, with some effort, to compute derivatives up to order $n$ for functions of $v$ variables, yielding the ${}_nD_v$ algebra. On the space representing the collection of functions that are $n$ times continuously differentiable in $v$ real variables, denoted by $\mathcal{C}^n(\mathbb{R}^v)$, an equivalence relation can be defined. Given two functions $f$ and $g \in \mathcal{C}^n(\mathbb{R}^v)$, the equivalence $f =_n g$ holds if and only if $f(0) = g(0)$, and all partial derivatives up to order $n$ coincide at the origin. The relation $=_n$ satisfies:

$$\begin{aligned}
&f =_n f && \forall\, f \in \mathcal{C}^n(\mathbb{R}^v) \\
&f =_n g \Rightarrow g =_n f && \forall\, f, g \in \mathcal{C}^n(\mathbb{R}^v) \\
&f =_n g \wedge g =_n h \Rightarrow f =_n h && \forall\, f, g, h \in \mathcal{C}^n(\mathbb{R}^v)
\end{aligned} \tag{1.29}$$

At this point, $=_n$ is an equivalence relation, and all elements related to a function $f$ can be grouped into a set, namely the equivalence class $[f]$ of the function $f$. Each class, often referred to as a *DA vector* or *DA number*, is uniquely specified by a collection of partial derivatives in all $v$ variables, up to order $n$; the set of all such classes is denoted by ${}_nD_v$. It can be observed that a function $f$ belongs to the same class as its Taylor expansion $T_n$

of order $n$ around the origin; therefore, $[f] = [T_n]$.

Starting from this principle, the construction of the differential algebra in ${}_1D_1$ can be generalized; as a result, the same properties hold for ${}_nD_v$, allowing the differential algebra to be extended to operate on functions in $\mathcal{C}^n(\mathbb{R}^v)$.

The algebra described above has been implemented in several tools. The most notable is COSY INFINITY, developed by *M. Berz and K. Makino* [40], which has been extensively applied in beam physics. In the context of aerospace applications, an equivalent implementation is provided by the Differential Algebra Core Engine (DACE) library, introduced by *M. Massari et al.* [41], which is currently in its second iteration.

### 1.5.3. High-Order Expansion of the Flow

The properties of DA can be exploited when numerically integrating a set of ODEs using an arbitrary integration scheme. Consider the following initial value problem:

$$\begin{cases} \dot{x} = f(x) \\ x(t_0) = x_0 \end{cases} \tag{1.30}$$

Any numerical integration scheme is typically based on simple algebraic operations and requires the evaluation of the ODE right-hand side $f(x)$ at multiple integration points; since a differential algebra is available, and the diagram in Figure 1.6 commutes, any propagation scheme can be replicated within the DA framework, enabling the expansion of the flow up to an arbitrary order $n$. This concept can be clarified by considering the first step of the forward Euler scheme:

$$x_1 = x_0 + \Delta t \cdot f(x_0) \tag{1.31}$$

Now consider a displaced initial point $x_0 + \delta x_0$, which can be obtained by substituting the initial point $x_0$ with the Taylor expansion of its identity function, $[x_0] = x_0 + \delta x_0$. If the first step of the forward Euler scheme is replicated within the DA framework, the result is the Taylor expansion of order $n$ of the flow, computed around $x_0$, and evaluated at $t = t_1$:

$$[x_1] = [x_0] + \Delta t \cdot f([x_0]) \tag{1.32}$$

By adopting the formalism $[x_1] = \mathcal{M}_{x_1}(\delta x_0)$, it is made explicit how $[x_1]$ is function of the displacement $\delta x_0$ with respect the nominal point $x_0$ around which the Taylor expansion is computed. The next step can be done by exploiting again the scheme previously introduced:

$$[x_2] = [x_1] + \Delta t \cdot f([x_1]) = \mathcal{M}_{x_1}(\delta x_0) + f(\mathcal{M}_{x_1}(\delta x_0)) = \mathcal{M}_{x_2}(\delta x_0) \tag{1.33}$$

With this approach, the expansion of the flow can be computed for any point up to the final time $t = t_f$, resulting in the map $\mathcal{M}_{x_f}(\delta x_0)$. Evaluating this map for a displaced initial condition $\delta x_0$ yields a $n^{\text{th}}$-order Taylor expansion approximation of the solution at $t = t_f$; in other words, the full propagation is replaced by a simple evaluation of a polynomial.

This method is particularly effective in scenarios requiring repeated integration of the dynamics, such as uncertainty propagation via Monte Carlo analysis or, as in the proposed thesis, in the context of OCPs. Although the initial computation of the polynomials may be more time-consuming than standard integration, especially for high-order maps, once the polynomials are available, subsequent propagations are straightforward, allowing the generation of a large number of samples with minimal computational effort. The accuracy of the results depends on the expansion order $n$, as highly nonlinear dynamics require higher-order maps to be accurately represented; moreover, it also depends on the magnitude of the displacement $\delta x_0$, since the polynomials are local approximations and the maps are therefore valid only within a neighborhood of the expansion point $x_0$.

# 2 | Model Predictive Control

This chapter explains how the LMPC and DAMPC have been implemented. First, the formulation of the linear controller is presented, clearly addressing the objective function, the constraints considered and the numerical optimizer used to solve the OCP. Then, the DAMPC is introduced, specifying the role of DA and the differences with respect to the LMPC; a dedicated section details the strategy for generating the polynomial maps and selecting the expansion order.

## 2.1. Linear Model Predictive Control

### 2.1.1. Preliminaries

The LMPC is formulated by leveraging the structure of a QP optimization, which consists of a quadratic cost function subject to a set of linear constraints; the classical formulation of such OCPs typically has the following structure:

$$\min_{\boldsymbol{z}} \quad J = \frac{1}{2}\boldsymbol{z}^T \boldsymbol{Q}_{cost}\, \boldsymbol{z} + \boldsymbol{b}_{cost}^T\, \boldsymbol{z} + \boldsymbol{c}_{cost} \tag{2.1a}$$

$$\text{s.t.} \quad \boldsymbol{z}_l \leq \boldsymbol{z} \leq \boldsymbol{z}_u \tag{2.1b}$$

$$\boldsymbol{A}_{eq} \cdot \boldsymbol{z} = \boldsymbol{b}_{eq} \tag{2.1c}$$

$$\boldsymbol{A}_{ineq} \cdot \boldsymbol{z} \leq \boldsymbol{b}_{ineq} \tag{2.1d}$$

Here, Equation 2.1a defines the most general form of a quadratic cost function, while Equation 2.1b imposes upper and lower bounds on the optimization variables; the linear equality and inequality constraints are expressed in Equation 2.1c and Equation 2.1d, respectively. This formulation will be specialized to the context of the RVD/B problem. To reduce the computational complexity, a move blocking strategy is adopted, in which only the first $M$ control inputs are optimized, whereas the remaining $N - M$ inputs are set to zero; the predicted states $\bar{\boldsymbol{x}}_i$ and control inputs $\bar{\boldsymbol{u}}_i$ are then collected into a single vector of the form:

$$\boldsymbol{z} = [\bar{\boldsymbol{x}}_0, \ldots, \bar{\boldsymbol{x}}_N \ , \ \bar{\boldsymbol{u}}_0, \ldots, \bar{\boldsymbol{u}}_{M-1}] \tag{2.2}$$

It is assumed that the target vehicle is equipped with a docking port oriented to enable an approach along the negative V-bar direction (i.e., from behind). As constraints, the following will be enforced:

- initial conditions and system dynamics;
- approach cone limitations;
- maximum control effort bounds;
- soft docking conditions.

A final remark concerns the choice of the optimization algorithm used to solve the OCP. The LMPC formulation results in a QP problem, for which very efficient dedicated solvers exist; however, the DAMPC, as will be shown later, is nonlinear, and thus cannot be solved using the same class of solvers. To ensure a fair comparison, a general nonlinear optimizer was adopted for both controllers. While this choice implies a higher computational burden for the LMPC, as the peculiar structure of a QP is not exploited, it guarantees that any difference in the maneuver cost arises only from the MPC formulation rather than from discrepancies in the optimization method. To this purpose, the Interior Point Optimizer (IPOPT) algorithm was selected through the use of the Python library *cyipopt* [42]. This kind of algorithm requires both the gradient and the Hessian of the objective function and the constraints, if these are not provided, it approximates them numerically, which can be computationally expensive; therefore, it is good practice to supply analytical expressions for these quantities, for this reason, the gradient and Hessian of both the objective function and the constraints will be explicitly specified.

### 2.1.2. Cost Function

To define the cost function, a quadratic formulation with a move blocking strategy is employed, as expressed in Equation 1.19; the target is assumed to be located at the origin and no reference control is provided, so both $\boldsymbol{x}_{ref}$ and $\boldsymbol{u}_{ref}$ are set to zero. The cost function is then expressed as:

$$J = \sum_{i=0}^{N-1} \|\bar{\boldsymbol{x}}_i\|^2_{\boldsymbol{Q}} + \sum_{i=0}^{M-1} \|\bar{\boldsymbol{u}}_i\|^2_{\boldsymbol{R}} + \|\bar{\boldsymbol{x}}_N\|^2_{\boldsymbol{Q_f}} \tag{2.3}$$

The weighting matrices $\boldsymbol{Q}$ and $\boldsymbol{R}$ are assumed to be diagonal and positive definite; the weights applied to the position and velocity states are denoted by $Q_{pos}$ and $Q_{vel}$, respectively, yielding the following structure:

$$\boldsymbol{Q} = \begin{bmatrix} Q_{pos} & & & & & \\ & Q_{pos} & & & & \\ & & Q_{pos} & & & \\ & & & Q_{vel} & & \\ & & & & Q_{vel} & \\ & & & & & Q_{vel} \end{bmatrix}, \qquad \boldsymbol{R} = \begin{bmatrix} R & & \\ & R & \\ & & R \end{bmatrix} \tag{2.4}$$

For the terminal cost $\boldsymbol{Q_f}$, the solution of the discrete algebraic Riccati equation, denoted by $\boldsymbol{P}$, is employed in accordance with standard stability theory. These weighting matrices are then assembled into a single block-diagonal matrix $\boldsymbol{Q}_{cost}$, which allows the cost function to be expressed in a compact form as:

$$J = \boldsymbol{z}^T \boldsymbol{Q}_{cost}\, \boldsymbol{z}, \qquad \boldsymbol{Q}_{cost} = \begin{bmatrix} \boldsymbol{Q} & & & & & & \\ & \ddots & & & & & \\ & & \boldsymbol{Q} & & & & \\ & & & \boldsymbol{P} & & & \\ & & & & \boldsymbol{R} & & \\ & & & & & \ddots & \\ & & & & & & \boldsymbol{R} \end{bmatrix} \tag{2.5}$$

The computation of the gradient and Hessian for this class of quadratic functions is straightforward, yielding the following expressions:

$$\boldsymbol{\nabla} J = (\boldsymbol{Q}_{cost}^T + \boldsymbol{Q}_{cost}) \cdot \boldsymbol{z}, \qquad \boldsymbol{\nabla}^2 J = \boldsymbol{Q}_{cost}^T + \boldsymbol{Q}_{cost} \tag{2.6}$$

### 2.1.3. Initial conditions and system dynamics

To propagate the system dynamics, a linearized model must be employed so that it can be imposed as a linear equality constraint; in the proposed work, the one introduced in Chapter 1.3 is adopted. Although the system is inherently LTV, it is treated as LTI within the prediction horizon in order to simplify the propagation. Specifically, at each time instant $t_k$, when the optimization problem is solved, the dynamics is frozen, and a single set of matrices, $\boldsymbol{A}_k$ and $\boldsymbol{B}_k$, is computed and used throughout the entire prediction horizon; once the current optimization is complete, at the next time step $t_{k+1}$, the dynamics is re-linearized and a new set of matrices, $\boldsymbol{A}_{k+1}$ and $\boldsymbol{B}_{k+1}$, is generated for the updated problem. This approximation is considered acceptable under the assumption that the system dynamics exhibit only minor variations over time, particularly when a sufficiently small sampling time is employed.

Having provided this brief introduction to explain how the dynamics will be treated, the initial conditions are enforced; at each time instant $t_k$, the current state of the spacecraft, denoted by $\boldsymbol{x}(t_k)$, is assumed to be known, and the following initial condition is imposed:

$$\bar{\boldsymbol{x}}_0 = \boldsymbol{x}(t_k) \tag{2.7}$$

Concerning the dynamics, the associated constraints ensure that each predicted state $\bar{\boldsymbol{x}}_{i+1}$ is consistent with the propagation of the preceding state $\bar{\boldsymbol{x}}_i$ under the assumed LTI dynamics; this is enforced through the following equalities:

$$\begin{aligned} &\bar{\boldsymbol{x}}_{i+1} - (\boldsymbol{A}_k\, \bar{\boldsymbol{x}}_i + \boldsymbol{B}_k\, \bar{\boldsymbol{u}}_i) = \boldsymbol{0} && \forall\, i = 0, \dots, M-1 \\ &\bar{\boldsymbol{x}}_{i+1} - (\boldsymbol{A}_k\, \bar{\boldsymbol{x}}_i) = \boldsymbol{0} && \forall\, i = M, \dots, N-1 \end{aligned} \tag{2.8}$$

The initial conditions and system dynamics can be enforced together in matrix form as follows:

$$\left[\begin{array}{cccccc|ccc} \boldsymbol{I} & & & & & & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ -\boldsymbol{A}_k & \boldsymbol{I} & & & & & -\boldsymbol{B}_k & & \\ & \ddots & \ddots & & & & & \ddots & \\ & & \ddots & \ddots & & & & & -\boldsymbol{B}_k \\ & & & \ddots & \ddots & & \boldsymbol{0} & \cdots & \boldsymbol{0} \\ & & & & \ddots & \ddots & \vdots & & \vdots \\ & & & & -\boldsymbol{A}_k & \boldsymbol{I} & \boldsymbol{0} & \cdots & \boldsymbol{0} \end{array}\right] \cdot \boldsymbol{z} = \begin{bmatrix} \boldsymbol{x}(t_k) \\ \boldsymbol{0} \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \boldsymbol{0} \end{bmatrix} \tag{2.9}$$

The first row enforces the initial condition, while the others impose the system dynamics across the prediction horizon; the matrix $\boldsymbol{B}_k$ appears only in the first $M$ block rows, in accordance with the move blocking strategy. The complete set of equality constraints can thus be expressed compactly and the gradient and Hessian are straightforward:

$$\boldsymbol{A}_{eq} \cdot \boldsymbol{z} = \boldsymbol{b}_{eq}, \qquad \boldsymbol{\nabla}(\boldsymbol{A}_{eq} \cdot \boldsymbol{z}) = \boldsymbol{A}_{eq}, \qquad \boldsymbol{\nabla}^2(\boldsymbol{A}_{eq} \cdot \boldsymbol{z}) = \boldsymbol{0} \tag{2.10}$$

### 2.1.4. Approach cone

According to the IRSIS standard, access to the KOS is allowed only through predefined conical corridors of ±10°, centered on the docking port axis. To facilitate the implementation of this constraint, two simplifying assumptions are adopted; first, the cone is assumed to extend indefinitely and is not limited to the KOS, thereby avoiding the need to model the various regions involved in a rendezvous, which would be outside the scope of this thesis; second, since the exact mathematical representation of a conical surface is nonlinear, the cone is approximated by a set of planar surfaces, allowing it to be enforced through linear inequality constraints. Using four planes, the following mathematical representation is obtained:

$$\begin{cases} \bar{y}_i + \bar{x}_i \tan\gamma \leq c_y \\ -\bar{y}_i + \bar{x}_i \tan\gamma \leq c_y \\ \bar{z}_i + \bar{x}_i \tan\gamma \leq c_z \\ -\bar{z}_i + \bar{x}_i \tan\gamma \leq c_z \end{cases} \tag{2.11}$$

where $\bar{x}_i$, $\bar{y}_i$ and $\bar{z}_i$ denote the components of the position vector of the predicted state $\bar{\boldsymbol{x}}_i$, while the parameter $\gamma$ represents the semi-aperture angle of the approach cone, set to 10°. The constants $c_y$ and $c_z$ define the offsets of the planar boundaries at the tip of the cone, corresponding to the aperture of the docking port; these offsets are determined based on the maximum allowable radial misalignment for the initial conditions required by the SCS, following the IDSS standard (Table 1.1), with a factor of $\sqrt{2}$ included to account for the worst-case misalignment scenario:

$$c_y = c_z = \frac{0.10}{\sqrt{2}}\,\mathrm{m} \approx 0.07\,\mathrm{m} \tag{2.12}$$

The inequalities expressed in Equation 2.11 can be written in matrix form as:

$$\begin{bmatrix} \tan\gamma & 1 & 0 & 0 & 0 & 0 \\ \tan\gamma & -1 & 0 & 0 & 0 & 0 \\ \tan\gamma & 0 & 1 & 0 & 0 & 0 \\ \tan\gamma & 0 & -1 & 0 & 0 & 0 \end{bmatrix} \cdot \bar{\boldsymbol{x}}_i \leq \begin{bmatrix} c_y \\ c_y \\ c_z \\ c_z \end{bmatrix} \qquad \forall\, i = 0, \dots, N \tag{2.13}$$

Finally, this structure can be used to assemble a block diagonal matrix $\boldsymbol{A}_{ineq}$ and obtain the classical linear inequality constraints, along with their gradient and Hessian:

$$\boldsymbol{A}_{ineq} \cdot \boldsymbol{z} \leq \boldsymbol{b}_{ineq}, \qquad \boldsymbol{\nabla}(\boldsymbol{A}_{ineq} \cdot \boldsymbol{z}) = \boldsymbol{A}_{ineq}, \qquad \boldsymbol{\nabla}^2(\boldsymbol{A}_{ineq} \cdot \boldsymbol{z}) = \boldsymbol{0} \tag{2.14}$$

### 2.1.5. Maximum control effort

The maximum control effort corresponds to the maximum acceleration deliverable given the limitations of the propulsion system, and is typically imposed as upper and lower bounds on the control input. Let $T_{max}$ denote the maximum available thrust and $m_{chaser}$ the mass of the chaser. Although a slight reduction of the latter is expected during the operations, this variation is neglected since the maneuvers are not very intensive and the mass is expected to change only slightly; therefore, $m_{chaser}$ is assumed constant. The maximum acceleration is then given by $T_{max}/m_{chaser}$; however, given a control $\bar{\boldsymbol{u}}_i$, strictly verifying this constraint would require computing the norm $\|\bar{\boldsymbol{u}}_i\|$, which is a nonlinear operation and cannot be imposed as simple bounds. To overcome this problem, a classical approach is to define the maximum control effort by introducing a factor of $\sqrt{3}$, obtaining $u_{max} = T_{max}/(\sqrt{3}\, m_{chaser})$; this maximum value is then imposed as upper and lower bounds on every component of $\bar{\boldsymbol{u}}_i$, resulting in the following set of inequalities:

$$\begin{bmatrix} -u_{max} \\ -u_{max} \\ -u_{max} \end{bmatrix} \leq \begin{bmatrix} \bar{u}_i^x \\ \bar{u}_i^y \\ \bar{u}_i^z \end{bmatrix} \leq \begin{bmatrix} u_{max} \\ u_{max} \\ u_{max} \end{bmatrix} \qquad \forall\, i = 0, \dots, M-1 \tag{2.15}$$

This constraint can be compactly expressed as:

$$\boldsymbol{z}_l \leq \boldsymbol{z} \leq \boldsymbol{z}_u \tag{2.16}$$

where $\boldsymbol{z}_l$ and $\boldsymbol{z}_u$ denote the lower and upper bounds applied to $\boldsymbol{z}$. The first $N+1$ components correspond to the initial and predicted states and, since no explicit bounds are imposed on them, these entries are set to $\pm\infty$; the remaining $M$ components refer to the control inputs and are bounded by $\pm u_{max}$, as derived above. Notice that for this class of constraints no gradient or Hessian is required.

### 2.1.6. Soft docking

The soft docking constraints are introduced to ensure that, at the docking instant, the velocity is sufficiently low to allow a safe and soft contact between the chaser and the target; to define these constraints, the initial conditions required by the SCS in accordance with the IDSS standard (Table 1.1) are taken as reference, leading to the definition of a terminal state $\boldsymbol{x}_{term}$ as follows:

$$\boldsymbol{x}_{term} = \begin{bmatrix} x \\ y \\ z \\ \dot{x} \\ \dot{y} \\ \dot{z} \end{bmatrix} = \begin{bmatrix} 0.05\ \mathrm{m} \\ 0.10/\sqrt{2}\,\mathrm{m} \\ 0.10/\sqrt{2}\,\mathrm{m} \\ 0.05\,\mathrm{m/s} \\ 0.04/\sqrt{2}\,\mathrm{m/s} \\ 0.04/\sqrt{2}\,\mathrm{m/s} \end{bmatrix} \approx \begin{bmatrix} 0.05\,\mathrm{m} \\ 0.07\,\mathrm{m} \\ 0.07\,\mathrm{m} \\ 0.05\,\mathrm{m/s} \\ 0.03\,\mathrm{m/s} \\ 0.03\,\mathrm{m/s} \end{bmatrix} \tag{2.17}$$

Again, the terminal radial misalignment and velocity are distributed along the two directions by introducing a factor of $\sqrt{2}$, thereby avoiding the need to compute their norm; moreover, it is important to note that the IDSS standard does not explicitly define a maximum allowable axial misalignment, and therefore a threshold of 0.05 m is set as an arbitrary value.

The soft docking constraints are not directly enforced within the OCP, as their implementation would be complex and computationally demanding, instead, they are used as terminal conditions to evaluate the success of a simulation; before starting any optimization, given the actual state $\boldsymbol{x}(t_k)$, the following is verified:

$$-\boldsymbol{x}_{term} \leq \boldsymbol{x}(t_k) \leq \boldsymbol{x}_{term} \tag{2.18}$$

If the actual state is bounded within $\pm\boldsymbol{x}_{term}$, the simulation is considered successful, otherwise, a new OCP is solved; moreover, if the chaser is close enough to the target for docking but its velocity is excessive, the gains are manually redefined to satisfy the terminal conditions.

## 2.2. Differential Algebra Model Predictive Control

### 2.2.1. Role of Differential Algebra

Given a generic nonlinear system dynamics, DA is exploited to propagate it and compute a high-order expansion of the flow, which is then used to enforce the dynamics as equality constraints resulting in the DAMPC; as a consequence, the predicted states involved in the optimization are generated with higher accuracy compared to those from a linearized model. The DAMPC follows the same formulation used for the LMPC, except for the equality constraints, which are nonlinear and expressed as a general function $g(\boldsymbol{z})$; the formulation in Equation 2.1 is modified as follows:

$$\min_{\boldsymbol{z}} \quad J = \frac{1}{2}\boldsymbol{z}^T \boldsymbol{Q}_{cost}\, \boldsymbol{z} + \boldsymbol{b}_{cost}^T\, \boldsymbol{z} + \boldsymbol{c}_{cost} \tag{2.19a}$$

$$\text{s.t.} \quad \boldsymbol{z}_l \leq \boldsymbol{z} \leq \boldsymbol{z}_u \tag{2.19b}$$

$$\boldsymbol{g}(\boldsymbol{z}) = \boldsymbol{b}_{eq} \tag{2.19c}$$

$$\boldsymbol{A}_{ineq} \cdot \boldsymbol{z} \leq \boldsymbol{b}_{ineq} \tag{2.19d}$$

Assume that a system dynamics affine in the control, as described in Equation 1.12, is available:

$$\dot{\boldsymbol{x}} = \boldsymbol{f}(\boldsymbol{x}, t) + \tilde{\boldsymbol{B}}\boldsymbol{u} \tag{2.20}$$

The dynamics is split into two contributions: the *free dynamics* $\boldsymbol{f}(\boldsymbol{x}, t)$ and the *control term* $\tilde{\boldsymbol{B}}\,\boldsymbol{u}$; while the latter can be accurately represented by a linear formulation, the free dynamics is typically the component that contains the nonlinearities, therefore, DA is employed to provide a more precise representation of this contribution. The linear propagation used in the LMPC, introduced in Equation 1.15, can be modified by replacing the term $\boldsymbol{A}_k\,\boldsymbol{x}_k$ with the high-order expansion of the flow from $t_k$ to $t_{k+1}$, computed around $\boldsymbol{x}_k$ and obtained propagating the free dynamics; consequently, the following expression is obtained:

$$\boldsymbol{x}_{k+1} = \mathcal{M}_{\boldsymbol{x}_{k+1}}(\delta\boldsymbol{x}_k) + \boldsymbol{B}_k\,\boldsymbol{u}_k \tag{2.21}$$

At this stage, this propagation scheme is applied to the MPC prediction horizon. Assume that, for each predicted state $\bar{\boldsymbol{x}}_i$, a polynomial map $\mathcal{M}_{\bar{\boldsymbol{x}}_{i+1}}(\delta\bar{\boldsymbol{x}}_i)$ is available to propagate the free dynamics to the next time instant; the terms $\delta\bar{\boldsymbol{x}}_i$ represent the displacement of the predicted state $\bar{\boldsymbol{x}}_i$ relative to the nominal point around which the map was computed. The control input is introduced under the same assumption as in the LMPC, where a single matrix $\boldsymbol{B}_k$ is employed throughout the entire prediction horizon. As in the linear case, the dynamics constraints ensure that each predicted state $\bar{\boldsymbol{x}}_{i+1}$ is consistent with the propagation of the preceding state $\bar{\boldsymbol{x}}_i$; this consistency is enforced through the following equalities:

$$\begin{aligned} &\bar{\boldsymbol{x}}_{i+1} - \left[\mathcal{M}_{\bar{\boldsymbol{x}}_{i+1}}(\delta\bar{\boldsymbol{x}}_i) + \boldsymbol{B}_k\,\bar{\boldsymbol{u}}_i\right] = \boldsymbol{0} \qquad && \forall\, i = 0, \dots, M-1 \\ &\bar{\boldsymbol{x}}_{i+1} - \left[\mathcal{M}_{\bar{\boldsymbol{x}}_{i+1}}(\delta\bar{\boldsymbol{x}}_i)\right] = \boldsymbol{0} && \forall\, i = M, \dots, N-1 \end{aligned} \tag{2.22}$$

Together with the initial conditions, the equality constraints $\boldsymbol{g}(\boldsymbol{z}) = \boldsymbol{b}_{eq}$ take the following form:

$$\begin{bmatrix} \bar{\boldsymbol{x}}_0 \\ \bar{\boldsymbol{x}}_1 - [\mathcal{M}_{\bar{\boldsymbol{x}}_1}(\delta\bar{\boldsymbol{x}}_0) + \boldsymbol{B}_k\,\bar{\boldsymbol{u}}_0] \\ \vdots \\ \bar{\boldsymbol{x}}_M - [\mathcal{M}_{\bar{\boldsymbol{x}}_M}(\delta\bar{\boldsymbol{x}}_{M-1}) + \boldsymbol{B}_k\,\bar{\boldsymbol{u}}_{M-1}] \\ \bar{\boldsymbol{x}}_{M+1} - \left[\mathcal{M}_{\bar{\boldsymbol{x}}_{M+1}}(\delta\bar{\boldsymbol{x}}_M)\right] \\ \vdots \\ \bar{\boldsymbol{x}}_N - [\mathcal{M}_{\bar{\boldsymbol{x}}_N}(\delta\bar{\boldsymbol{x}}_{N-1})] \end{bmatrix} = \begin{bmatrix} \boldsymbol{x}(t_k) \\ \boldsymbol{0} \\ \vdots \\ \vdots \\ \vdots \\ \vdots \\ \boldsymbol{0} \end{bmatrix} \tag{2.23}$$

The gradients of the equality constraints represented by $g(\boldsymbol{z})$ assume a form similar to that computed for the LMPC, but instead of the matrix $\boldsymbol{A}_k$, the gradients of the polynomial maps $\boldsymbol{\nabla}\mathcal{M}_{\bar{\boldsymbol{x}}_{i+1}}(\delta\bar{\boldsymbol{x}}_i)$ are required. The analytical expression of the derivatives can be easily obtained within the DA framework, allowing their computation with limited computational effort. Nevertheless, a simplification must be introduced to reduce the time required to solve the OCP. Indeed, at each iteration the solver tests a different set of predicted states and controls, which requires multiple evaluations of both the constraints and their gradients; this is computationally demanding, as the number of functions representing the latter is six times the number of polynomials representing the constraints. Specifically, if $6N$ functions describe the high-order expansion of the flow along the prediction horizon, then $6 \times 6N$ functions are required to represent their gradients. Although the evaluation of a single polynomial is straightforward, when many are involved, as in the case of long prediction horizons characterized by large values of $N$, the procedure becomes prohibitively time-consuming. To mitigate this problem, the derivatives are treated as static quantities by extracting only the constant part of the polynomials, which corresponds to evaluating them at $\delta\bar{\boldsymbol{x}}_i = \boldsymbol{0}$; for compactness, these gradients are denoted as $\boldsymbol{\nabla}\mathcal{M}^0_{\bar{\boldsymbol{x}}_{i+1}} = \boldsymbol{\nabla}\mathcal{M}_{\bar{\boldsymbol{x}}_{i+1}}(\boldsymbol{0})$. This strategy eliminates the need for their evaluation within the OCP, thereby significantly reducing the required computational effort, while producing no significant changes in the final optimization results. The gradients so computed are organized as follows:

$$\boldsymbol{\nabla}\left[\boldsymbol{g}(\boldsymbol{z})\right] = \left[\begin{array}{cccccc|ccc}
\boldsymbol{I} & & & & & & \boldsymbol{0} & \cdots & \boldsymbol{0} \\
-\boldsymbol{\nabla}\mathcal{M}^0_{\bar{\boldsymbol{x}}_1} & \boldsymbol{I} & & & & & -\boldsymbol{B}_k & & \\
 & \ddots & \ddots & & & & & \ddots & \\
 & & \ddots & \ddots & & & & & -\boldsymbol{B}_k \\
 & & & \ddots & \ddots & & \boldsymbol{0} & \cdots & \boldsymbol{0} \\
 & & & & \ddots & \ddots & \vdots & & \vdots \\
 & & & & -\boldsymbol{\nabla}\mathcal{M}^0_{\bar{\boldsymbol{x}}_N} & \boldsymbol{I} & \boldsymbol{0} & \cdots & \boldsymbol{0}
\end{array}\right] \tag{2.24}$$

The same concept is applied for computing the Hessians: the derivatives are first obtained using DA, after which the constant part is extracted and passed to the OCP. In this case, the representation is more complex, as for each map $\mathcal{M}_{\bar{\boldsymbol{x}}_{i+1}}(\delta\bar{\boldsymbol{x}}_i)$ a $6 \times 6 \times 6$ Hessian can be defined, resulting in a three-dimensional array. Furthermore, the initial conditions and the linear contributions appearing in the constraints expressed in Equation 2.23 vanish, since a second-order derivative is involved. As a result, only the derivatives of the polynomial maps are required, with a negative sign to ensure consistency with the constraints, corresponding to a collection of $-\boldsymbol{\nabla}^2\mathcal{M}^0_{\bar{\boldsymbol{x}}_{i+1}}$.

## 2.2.2. Polynomial Maps Generation Strategy

So far, it has been assumed that the polynomial maps were already available. In this section, the procedure for generating them is described. In general, two factors affect the accuracy of a polynomial expansion: the order $n$ and the magnitude of the displacement with respect to the expansion point where the polynomial is evaluated. The choice of the order will be addressed in the next section, while the present discussion focuses on the selection of the expansion point and the propagation of the dynamics.

The basic idea is to rely on the initial guess to compute the polynomials; an OCP can be interpreted as a refinement procedure that, starting from a generic initial guess $\boldsymbol{z}_k^{guess}$, computes an optimal and constraint-compliant solution $\boldsymbol{z}_k^{opt}$. The choice of the initial guess is a critical factor that strongly influences the convergence speed of the method; a classical approach in the context of the MPC is the so called *warm start* strategy, where the optimal solution of the previous optimization, $\boldsymbol{z}_k^{opt}$, is used as the initial guess $\boldsymbol{z}_{k+1}^{guess}$ for the subsequent optimization, after being properly shifted forward by one time step. This approach ensures rapid convergence, as the previous solution is expected to be both optimal and constraint-compliant, making the new solution $\boldsymbol{z}_{k+1}^{opt}$ close to the provided guess. Consequently, each OCP requires only minor adjustments to the predicted trajectory, resulting in limited variations. The nominal points around which the polynomial expansions of the flow are computed can thus be extracted from the initial guess and propagated forward in time to obtain the maps used by the DAMPC, thereby limiting the displacement $\delta\bar{\boldsymbol{x}}_i$, as the predicted states are expected to remain close to the expansion point.
So, at a given time instant $t_k$ an initial guess $\boldsymbol{z}_k^{guess}$ is provided; let $\bar{\boldsymbol{x}}_i^{guess}$ and $\bar{\boldsymbol{u}}_i^{guess}$ denote the states and controls composing it:

$$\boldsymbol{z}_k^{guess} = \left[\bar{\boldsymbol{x}}_0^{guess}, \dots, \bar{\boldsymbol{x}}_N^{guess}\,,\ \bar{\boldsymbol{u}}_0^{guess}, \dots, \bar{\boldsymbol{u}}_{M-1}^{guess}\right] \tag{2.25}$$

The polynomial maps $\mathcal{M}_{\bar{\boldsymbol{x}}_{i+1}}(\delta\bar{\boldsymbol{x}}_i)$ can be computed by propagating the free dynamics forward in time within the DA framework, using $\bar{\boldsymbol{x}}_i^{guess}$ as initial condition. Given a generic system dynamics $f(\boldsymbol{x}, t)$, the initial value problem is formulated as follows:

$$\begin{cases} \dot{\boldsymbol{x}} = f(\boldsymbol{x}, t) \\ \boldsymbol{x}(t_{k+i}) = \bar{\boldsymbol{x}}_i^{guess} \end{cases} \Rightarrow\ \bar{\boldsymbol{x}}_{i+1} = \int_{t_{k+i}}^{t_{k+i+1}} f(\boldsymbol{x}, \tau)\, d\tau \qquad \forall\, i = 0, \dots, N-1 \tag{2.26}$$

Moreover, to increase computational efficiency, it is not necessary to generate all $N$ polynomial maps before every optimization, but instead, the previous list can be updated by removing the first map, now outdated, and generating the last one using $\bar{\boldsymbol{x}}_{N-1}^{guess}$ as expansion point. This approach greatly reduces the computational cost without introducing any significant difference compared to the case where all $N$ maps are recomputed at each optimization.

The only critical issue to be faced concerns the accuracy of the polynomial maps used in the very first iteration. At the beginning of the simulation, at time instant $t_0$, a manually provided initial guess is used to compute the polynomial maps and solve the OCP for the first time; in general, this initial guess tends to be far from the optimal solution, resulting in a significant displacement of the predicted states with respect to the expansion points, and thus the polynomial maps tend to be quite inaccurate. At the next time step $t_1$, if only the last map is updated, then $N-1$ inaccurate maps computed at $t_0$ are retained, leading to another inaccurate prediction; this situation persists until all the polynomials computed at $t_0$ have been replaced, which requires a total of $N$ optimizations. A better strategy is to recompute all the polynomial maps again at $t_1$, using the optimal solution obtained at $t_0$ and thus a more suitable set of expansion points; from $t_2$ onward, it is sufficient to update only the last map as previously described. This strategy is represented in Figure 2.1.

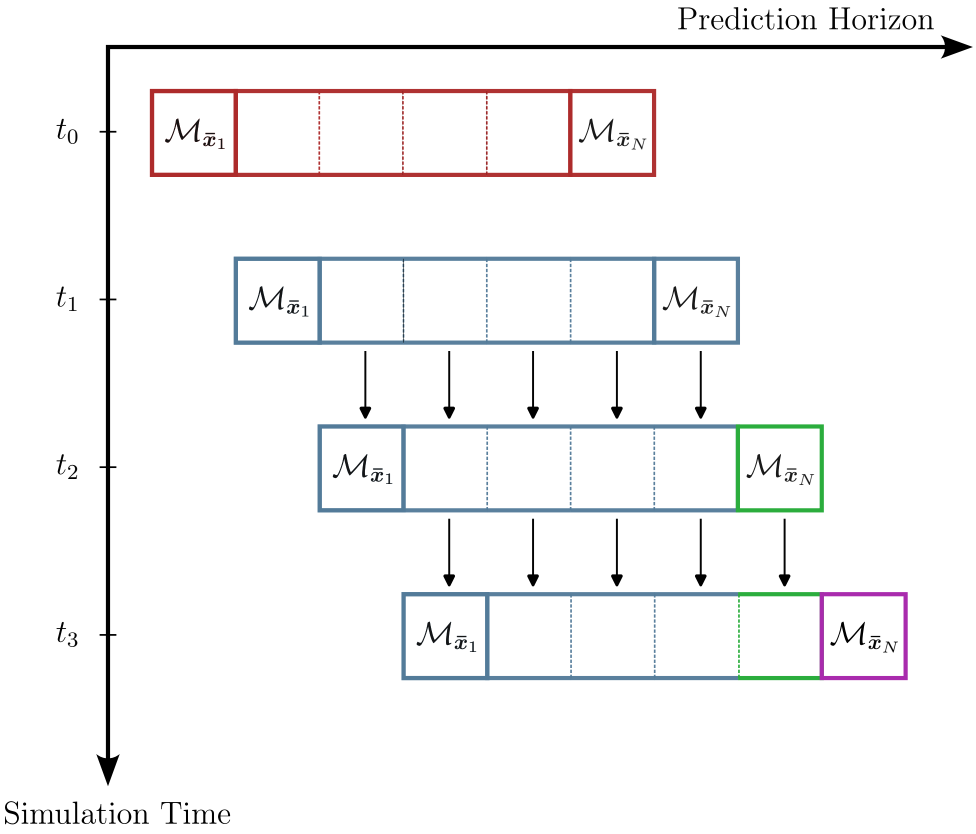


Figure 2.1: Polynomial maps generation strategy. Colors denote the time instant when the maps are generated, and arrows indicate how maps are passed to the next time instant.

Finally, a note on the propagation method is provided here. As already mentioned, any propagation method can be implemented within the DA framework by simply replacing real-number operations with their corresponding operators in DA. In this work, a Runge-Kutta (RK) scheme was implemented and tested in two variants: RK-23 and RK-78, with the first offering the best trade-off between accuracy and computational efficiency, since fewer function evaluations are required. The numerical results are reported in Chapter 3.2, where the simulations are presented.

### 2.2.3. Expansion Order Selection

The other critical factor affecting the accuracy of the polynomial maps is the choice of the expansion order $n$. In principle, higher-orders provide greater accuracy, however, they also require a significantly larger number of coefficients and, consequently, greater computational time for generating the maps; this issue is particularly critical in real time applications, where a trade-off becomes necessary. The procedure for selecting the expansion order is outlined here, while the numerical results for the specific case of the relative dynamics introduced in Chapter 1.3 are presented in the next chapter.

The procedure is straightforward. Given a nominal initial condition $\boldsymbol{x}_0^{nom}$, the generic nonlinear system dynamics $f(\boldsymbol{x}, t)$ is propagated within the DA framework, producing multiple maps $\mathcal{M}_{\boldsymbol{x}_k}(\delta\boldsymbol{x}_0)$ on a predefined time grid. Subsequently, a displaced initial condition $\boldsymbol{x}_0^{disp} = \boldsymbol{x}_0^{nom} + \delta\boldsymbol{x}_0^{disp}$ is considered, and the system dynamics is propagated again using a traditional approach to obtain the exact trajectory on the same time grid employed for generating the polynomial maps; the resulting points are denoted by $\boldsymbol{x}_k^{ex}$. The polynomial maps are then evaluated at the displacement $\delta\boldsymbol{x}_0^{disp}$, yielding a set of points $\boldsymbol{x}_k^{DA}$. The accuracy is assessed by computing, for each point, the error $\boldsymbol{\varepsilon}_k = \boldsymbol{x}_k^{DA} - \boldsymbol{x}_k^{ex}$, from which the position and velocity errors can be readily extracted:

$$\varepsilon_k^{pos} = \sqrt{\varepsilon_{k,x}^2 + \varepsilon_{k,y}^2 + \varepsilon_{k,z}^2}, \qquad \varepsilon_k^{vel} = \sqrt{\varepsilon_{k,v_x}^2 + \varepsilon_{k,v_y}^2 + \varepsilon_{k,v_z}^2} \tag{2.27}$$

This analysis can be performed for different orders $n$, and the results can be compared to provide a comprehensive perspective on which order offers the most suitable choice. It should be noted that the magnitude of the displacement $\delta\boldsymbol{x}_0^{disp}$ is a critical factor, as very large displacements can lead to significant prediction errors. Nevertheless, since all polynomial maps of different orders are evaluated using the same displacement, the resulting comparison between them remains valid, as the errors introduced affect all maps equally. In general, for $n$ equal to 1, the accuracy is expected to match that of the linearized model. Increasing the expansion order leads to higher accuracy, however, significant improvements can already be obtained with relatively low values, such as 3 or 4; for real-time applications, employing very high values, such as 10 or above, is typically unnecessary, as the gain in accuracy is expected to be limited while the computational cost increases substantially.

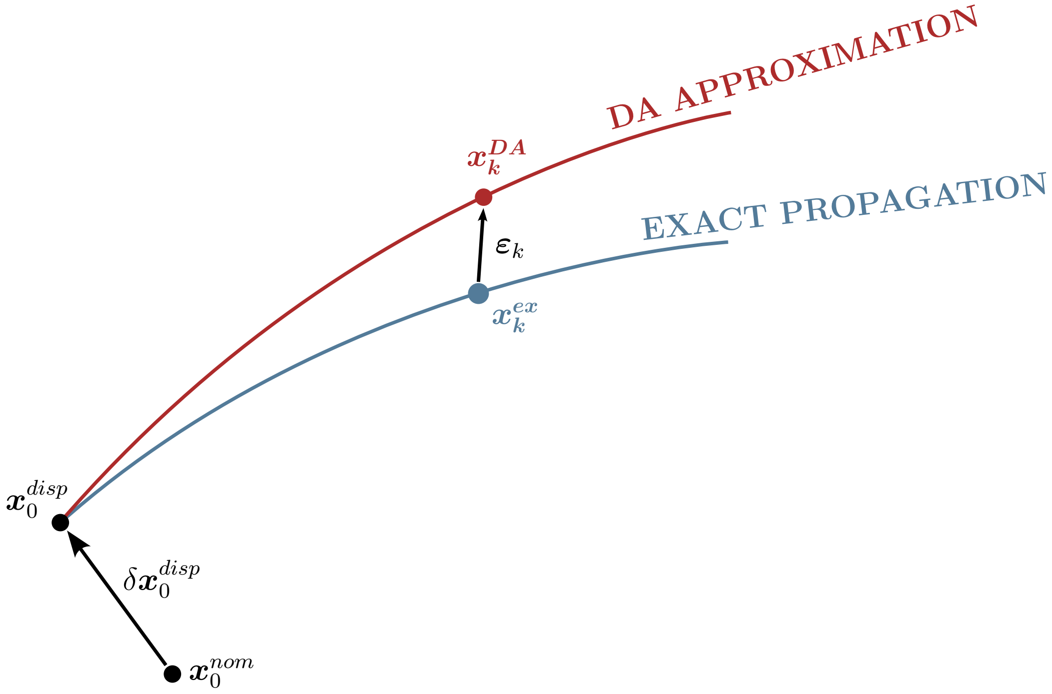
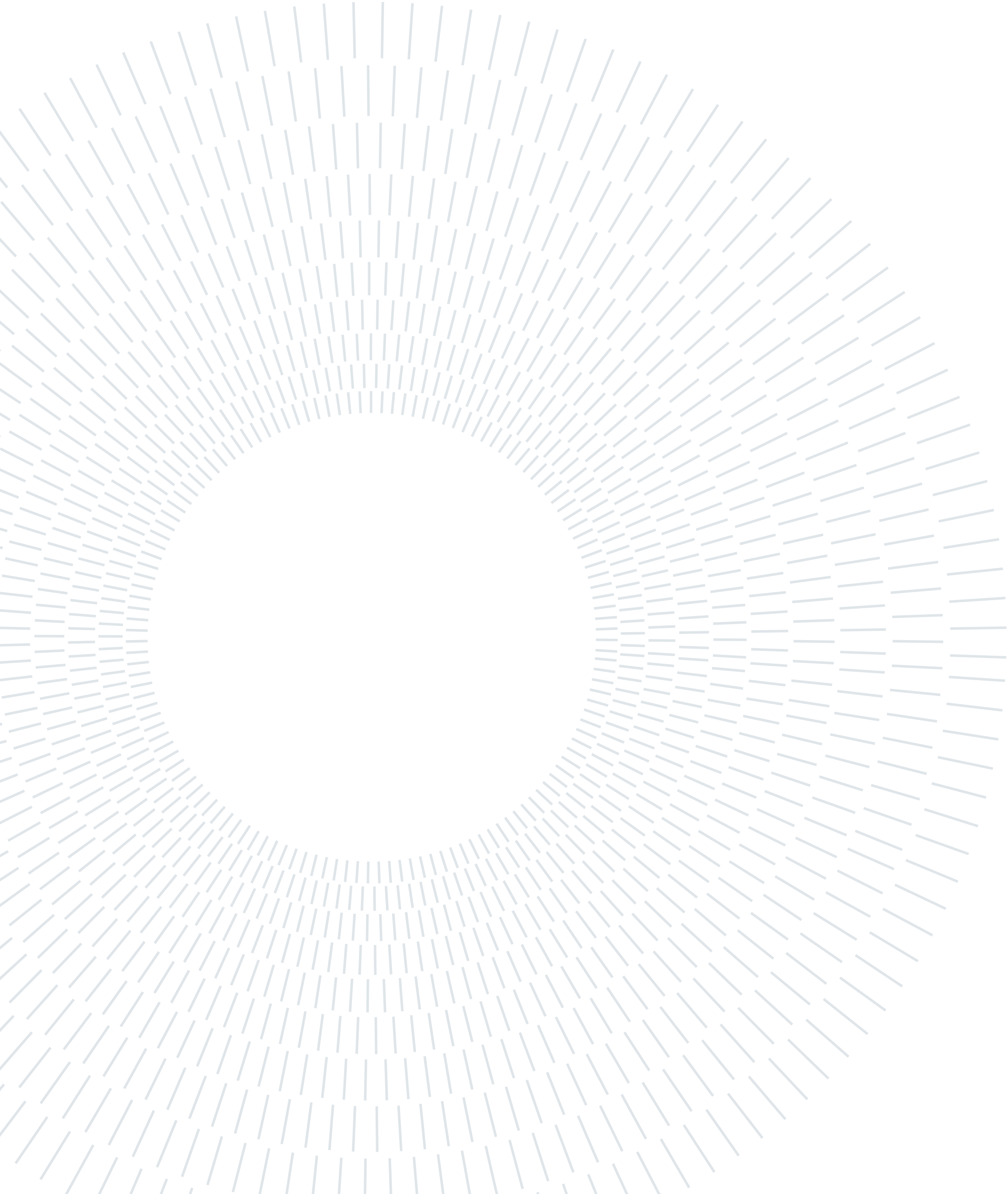


Figure 2.2: Schematic representation of the DA accuracy evaluation process.



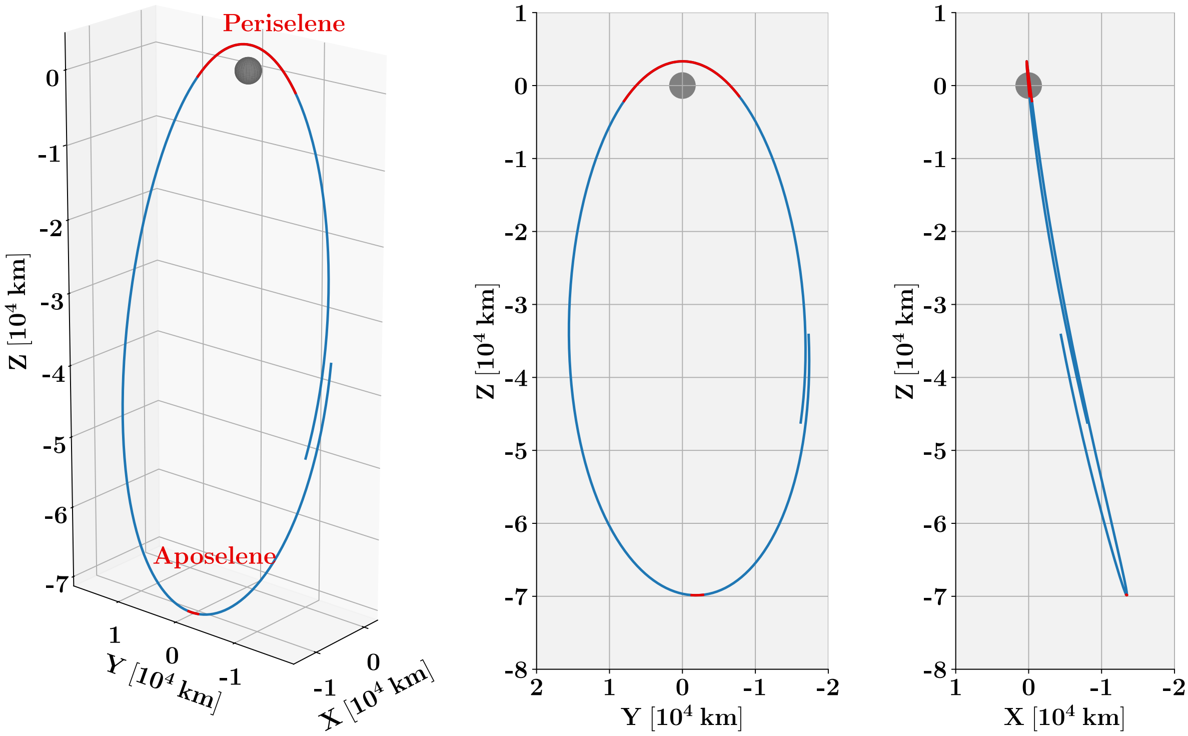
# 3 | Simulations and Results

In this chapter, the simulations are carried out and the results are presented. A preliminary section outlines the initial setup: the orbit is generated, the simulation environment is introduced, and a trade-off analysis is performed to select the expansion order. Subsequently, two sets of simulations are conducted depending on how the sampling time of the controller is managed.

## 3.1. Orbit Generation and Simulation Environment

As mentioned in the Introduction, NASA proposes to locate the Lunar Gateway on an NRHO of the $L_2$ southern family, with an orbital period selected for a 9:2 Lunar Synodic Resonance [6]. NASA also provides a kernel for the orbit, generated by performing a high fidelity ephemeris propagation considering four bodies: the Earth, the Moon, the Sun and Jupiter; no disturbance was taken into account. The target is assumed to lie on this orbit, shown in Figure 3.1 covering the time span from January 5, 2030, to seven days later; since the kernel provides the evolution of the state vector in the inertial frame $\mathcal{I}$, the state is converted to the synodic frame $\mathcal{M}$ by applying the corresponding direction cosine matrix to rotate the position vector, and by using the transport theorem to transform the velocity vector.

Several simulations will be carried out at both the aposelene and periselene to highlight the differences between these two critical locations. The aposelene is generally considered the most suitable for a RVD/B in this context, primarily due to the lower velocity that characterizes this region. In contrast, conducting such operations at the periselene is typically excluded because of the higher forces acting on the spacecraft; nevertheless, this location will be considered in the simulations as nonlinearities are more pronounced here, thereby accentuating the differences between the two controllers. Aposelene and periselene have been identified and are highlighted in Figure 3.1 by the red lines, showing the trajectory around those points within a time window of ±2 hours. The initial states obtained at the beginning of these regions are reported in Table 3.1 and 3.2, and they will serve as initial conditions for the target in the subsequent simulations.



(a) Three-dimensional view (b) Projection on $y$-$z$ plane (c) Projection on $x$-$z$ plane

Figure 3.1: NRHO representation in the synodic reference frame $\mathcal{M}$, generated on January 5, 2030. The aposelene and periselene are highlighted in red.

| Aposelene | |
|---|---|
| $x_0$ | $-13\,389.5$ km |
| $y_0$ | $-2\,814.8$ km |
| $z_0$ | $-69\,798.4$ km |
| $\dot{x}_0$ | $-0.007$ km/s |
| $\dot{y}_0$ | $0.107$ km/s |
| $\dot{z}_0$ | $-0.012$ km/s |

Table 3.1: Target initial conditions at the aposelene, in the synodic frame $\mathcal{M}$.

| Periselene | |
|---|---|
| $x_0$ | $-450.7$ km |
| $y_0$ | $8\,002.9$ km |
| $z_0$ | $-2\,116.0$ km |
| $\dot{x}_0$ | $0.109$ km/s |
| $\dot{y}_0$ | $-0.584$ km/s |
| $\dot{z}_0$ | $0.853$ km/s |

Table 3.2: Target initial conditions at the periselene, in the synodic frame $\mathcal{M}$.

Regarding the dynamical models, the CRTBP described in Equation 1.4 is assumed to represent the exact governing dynamics and is used to propagate the aforementioned initial conditions to compute the trajectory of the target in the synodic frame $\mathcal{M}$; in this way, the kernel is used only to extract the initial points, while the rest of the simulations are carried out independently of the kernel itself. The trajectory thus computed is then passed to the CNERM, introduced in Equation 1.11, which are used to propagate the relative trajectory of the chaser in the LVLH frame $\mathcal{L}$. It should be noted that the CNERM are based on the CRTBP, and no simplifying assumptions were introduced during their derivation; consequently, they represent the exact dynamics, just as the CRTBP does. The CNERM are also employed to generate the polynomial maps used in the DAMPC, whereas the LMPC is developed based on the linearized equations of motion, referred to as the CLERM and introduced in Equation 1.13.

Regarding the chaser, the only characteristic that needs to be defined is the maximum control input $u_{\text{max}}$, as required by the maximum control effort constraint imposed in Equation 2.15. Assuming a chaser mass of 1000 kg equipped with thrusters capable of generating a maximum force of 10 N, the maximum control inputs is set to approximately $5.7 \cdot 10^{-3}$ m/s$^2$.

## 3.2. Expansion Order Trade-Off

The procedure described in Chapter 2.2 has been applied to identify the expansion order that provides the best balance between accuracy and computational cost. The propagated system dynamics used to generate the polynomial maps correspond to the CNERM, as previously discussed, and the nominal and displaced initial conditions are reported in Table 3.3 and 3.4, both expressed in the LVLH frame $\mathcal{L}$. The analysis has been carried out over a total duration of 2 hours on a uniform time grid with a spacing of 4 seconds, resulting in 1 799 propagated points, and thus polynomial maps. Expansion orders from 1 to 4 have been tested at both the aposelene and periselene, and for comparison, the linearized CLERM used in the LMPC have also been included.

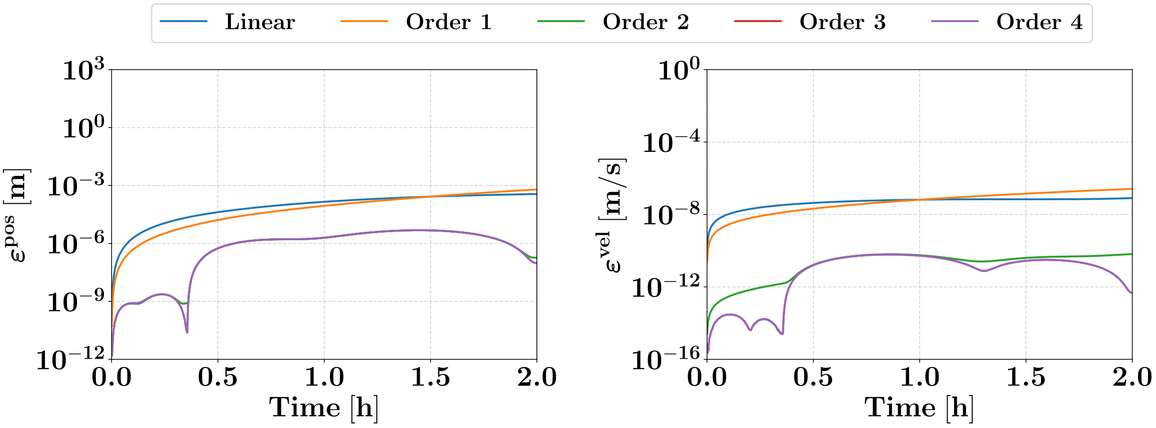
| **Nominal** | |
|---|---|
| $x_0$ | $-10\,000$ m |
| $y_0$ | 0 m |
| $z_0$ | 0 m |
| $\dot{x}_0$ | 0 m/s |
| $\dot{y}_0$ | 0 m/s |
| $\dot{z}_0$ | 0 m/s |

Table 3.3: Nominal initial conditions, in the LVLH frame $\mathcal{L}$.

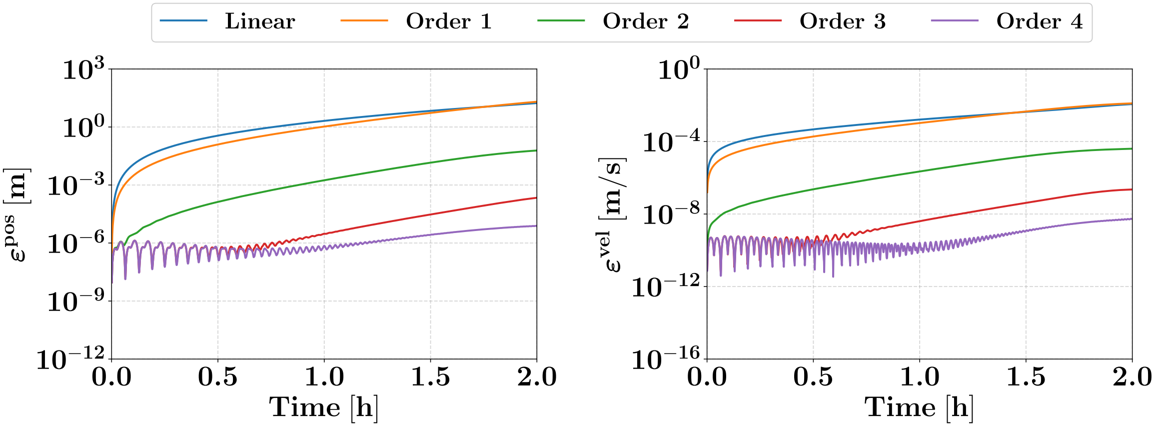
| **Displaced** | |
|---|---|
| $x_0$ | $-7\,000$ m |
| $y_0$ | $3\,000$ m |
| $z_0$ | $3\,000$ m |
| $\dot{x}_0$ | 1 m/s |
| $\dot{y}_0$ | 1 m/s |
| $\dot{z}_0$ | 1 m/s |

Table 3.4: Displaced initial conditions, in the LVLH frame $\mathcal{L}$.

The evolution of the position and velocity errors is shown in Figure 3.2 on a logarithmic scale. As expected, the accuracy of the linearized model matches that of the expansion of order 1; moreover, as the order increases, the errors decrease, indicating a more accurate representation of the system dynamics. The expansion of order 2 could be a viable solution at the aposelene, but at the periselene significant errors are introduced, and better results can be obtained with a higher-order expansion. The cases of order 3 and 4 yield identical results at the aposelene, with the two curves completely overlapping, while at the periselene they perform similarly only during the first 30 minutes, after which they begin to diverge. This behavior can be attributed to the stronger nonlinearities at the periselene, which require a higher expansion order to accurately capture the system dynamics over longer propagation times. Nevertheless, for sufficiently short periods, which is the case for an MPC prediction horizon, the expansion of order 3 is adequately accurate at both the aposelene and periselene, and the usage of a higher-order expansion does not provide significant benefits.



(a) Aposelene.



(b) Periselene.

Figure 3.2: Polynomial maps propagation errors for different expansion orders, propagated using RK-23. Left: position error, right: velocity error.

Notice that the curves exhibit several cusp points, particularly for the expansions of order 3 and 4 at the periselene. This behavior arises from the nature of the polynomial maps, which produce predicted trajectories that tend to oscillate around the exact one; when the two trajectories intersect, the errors temporarily drop to zero, and their norm produces a cusp point. The same effect will also be observed later when evaluating the prediction errors of the DAMPC.

Regarding the propagation method and computational cost, both RK-23 and RK-78 have been tested and evaluated; the results, obtained using an Intel Core i5-8300H, are reported in Table 3.5 and 3.6. The two methods showed no significant differences in the accuracy of the polynomial maps; for clarity, Figure 3.2 was generated using RK-23, while very similar results were obtained with RK-78, yielding an almost identical figure which is reported in Appendix A.2 for completeness. In contrast, the computational times required by the two propagation schemes differ significantly, with RK-78 being almost four times slower. Since RK-23 is faster while providing the same accuracy as RK-78, it was selected as the propagation scheme for generating the polynomial maps used in the DAMPC.

Finally, as expected, higher-orders require more time to generate the polynomial maps; nevertheless, the differences are relatively small, especially considering that 1 799 maps have been generated here, whereas in the DAMPC only a single propagation is required for each optimization. Taking this into account, together with the previously discussed observations on the errors, an expansion of order 3 provides the best trade-off between accuracy and computational efficiency. For this reason, it has been selected for generating the polynomial maps employed in the DAMPC.

| | **Aposelene** | **Periselene** |
|---|---|---|
| Order 1 | 55.3 s | 53.2 s |
| Order 2 | 56.4 s | 55.0 s |
| Order 3 | 58.9 s | 58.6 s |
| Order 4 | 64.0 s | 65.2 s |

Table 3.5: Computational cost with the RK-23 propagator.

| | **Aposelene** | **Periselene** |
|---|---|---|
| Order 1 | 215.4 s | 214.0 s |
| Order 2 | 221.3 s | 220.0 s |
| Order 3 | 240.2 s | 231.3 s |
| Order 4 | 252.4 s | 250.0 s |

Table 3.6: Computational cost with the RK-78 propagator.

## 3.3. Rendezvous Simulations - Fixed Sampling Time

### 3.3.1. Simulations Setup

With the simulation environment defined and the expansion order for the polynomial maps selected, the first set of simulations is carried out. A RVD/B scenario is simulated, with the chaser approaching the target from different initial conditions. The simulations are grouped into three categories based on the initial distance along V-bar: *short range* refers to cases where the chaser starts from 200 m, *medium range* from 2 000 m, and *long range* from 10 000 m. For each category, 21 simulations are conducted. In the first, the chaser is perfectly aligned with the V-bar and is referred to as the *nominal case*; this is used to generate specific figures, such as the control inputs history or the prediction accuracy. The remaining 20 include displacements along the other two directions, generated from a uniform grid of distances from which random pairs are sampled to define the coordinates along the R-bar and H-bar. In total, 63 simulations are conducted at both the aposelene and periselene, resulting in 126 simulations overall. The corresponding initial conditions are summarized in Table 3.7, 3.8 and 3.9.

| **Short Range** | |
|---|---|
| $x_0$ | $-200$ m |
| $y_0$ | $\in [-30, 30]$ m |
| $z_0$ | $\in [-30, 30]$ m |
| $\dot{x}_0$ | 0 m/s |
| $\dot{y}_0$ | 0 m/s |
| $\dot{z}_0$ | 0 m/s |

Table 3.7: Short range chaser initial conditions, in the LVLH frame $\mathcal{L}$.

| **Medium Range** | |
|---|---|
| $x_0$ | $-2\,000$ m |
| $y_0$ | $\in [-300, 300]$ m |
| $z_0$ | $\in [-300, 300]$ m |
| $\dot{x}_0$ | 0 m/s |
| $\dot{y}_0$ | 0 m/s |
| $\dot{z}_0$ | 0 m/s |

Table 3.8: Medium range chaser initial conditions, in the LVLH frame $\mathcal{L}$.

| **Long Range** | |
|---|---|
| $x_0$ | $-10\,000$ m |
| $y_0$ | $\in [-1\,500, 1\,500]$ m |
| $z_0$ | $\in [-1\,500, 1\,500]$ m |
| $\dot{x}_0$ | 0 m/s |
| $\dot{y}_0$ | 0 m/s |
| $\dot{z}_0$ | 0 m/s |

Table 3.9: Long range chaser initial conditions, in the LVLH frame $\mathcal{L}$.

Concerning the controller setup, both the DAMPC and LMPC are initialized with the same set of parameters. The sampling time is fixed at $T_s = 4$ s, and the prediction horizon length is set to $N = 30$ steps, corresponding to a total duration of 2 minutes; the control horizon covers $M = 15$ steps. The gains vary depending on the initial conditions and have been manually tuned through a trial and error approach to ensure successful docking. The parameters are summarized in Table 3.10, 3.11 and 3.12.

| Short Range | |
|---|---|
| $T_s$ | 4 s |
| $N$ | 30 |
| $M$ | 15 |
| $Q_{pos}$ | $10^{13}$ |
| $Q_{vel}$ | $10^{7}$ |
| $R$ | $10^{0}$ |

Table 3.10: Fixed $T_s$ simulations. Short range controller parameters.

| Medium Range | |
|---|---|
| $T_s$ | 4 s |
| $N$ | 30 |
| $M$ | 15 |
| $Q_{pos}$ | $10^{13}$ |
| $Q_{vel}$ | $10^{8}$ |
| $R$ | $10^{0}$ |

Table 3.11: Fixed $T_s$ simulations. Medium range controller parameters.

| Long Range | |
|---|---|
| $T_s$ | 4 s |
| $N$ | 30 |
| $M$ | 15 |
| $Q_{pos}$ | $10^{13}$ |
| $Q_{vel}$ | $10^{8}$ |
| $R$ | $10^{2}$ |

Table 3.12: Fixed $T_s$ simulations. Long range controller parameters.

The comparison is based on three evaluation metrics: prediction accuracy, solution optimality and computational cost. To evaluate the prediction accuracy, a procedure similar to the one described in Chapter 2.2 is employed, with a conceptual variation specifically adapted for these kinds of simulations. At each time instant $t_k$, an OCP is solved, generating a set of predicted states $\bar{\boldsymbol{x}}_i$; these are then compared to the exact trajectory propagated using the exact dynamics, and the position and velocity errors, $\varepsilon_i^{pos}$ and $\varepsilon_i^{vel}$, are computed as defined in Equation 2.27, yielding $N$ values for each optimization. To represent the prediction errors over the entire simulation in a compact form, all $N$ errors are summarized by their mean and standard deviation; these two quantities are then associated with the OCP solved at time $t_k$, which generated them.

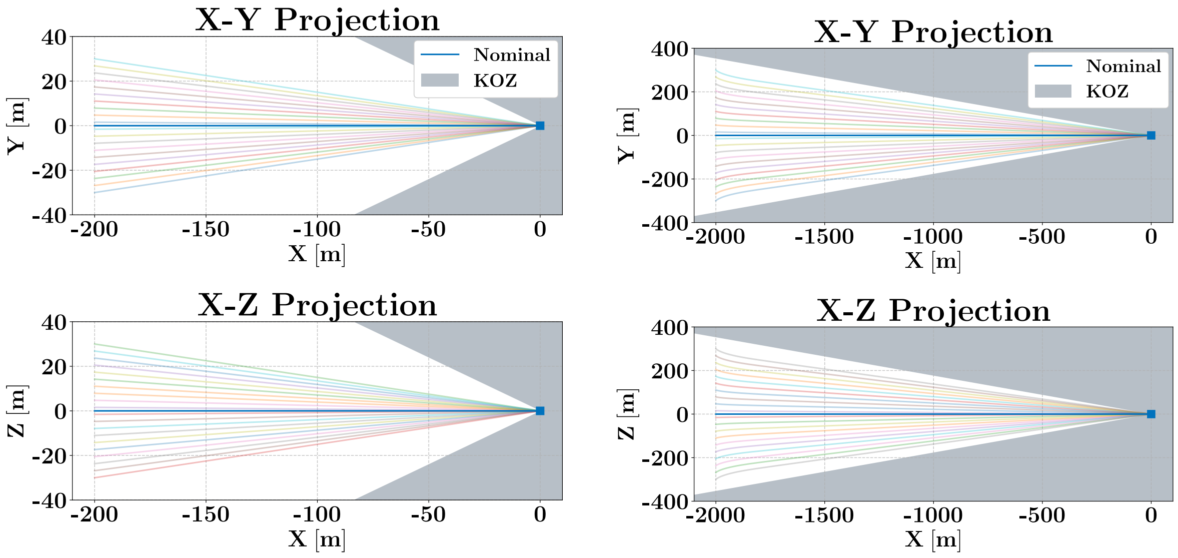
$$\begin{aligned} \bar{\varepsilon}_k^{pos} &= \frac{1}{N}\sum_{i=1}^{N}\varepsilon_i^{pos}, \qquad \sigma_{\varepsilon_k}^{pos} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\varepsilon_i^{pos}-\bar{\varepsilon}_k^{pos}\right)^2} \\ \bar{\varepsilon}_k^{vel} &= \frac{1}{N}\sum_{i=1}^{N}\varepsilon_i^{vel}, \qquad \sigma_{\varepsilon_k}^{vel} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\varepsilon_i^{vel}-\bar{\varepsilon}_k^{vel}\right)^2} \end{aligned} \tag{3.1}$$

The other two metrics are straightforward: the solution optimality is evaluated based on the total $\Delta v$ required for the maneuver, while the computational cost is assessed by measuring the CPU time required to set up the problem and solve the OCP.

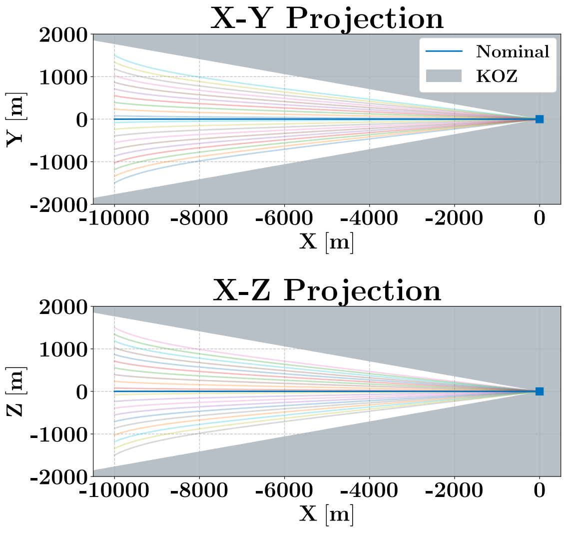
In the following sections, the results of the DAMPC and LMPC are first presented, followed by a direct comparison.

### 3.3.2. Simulation Results - DAMPC

All simulations have been successfully completed, and the trajectories obtained at the aposelene are shown in Figure 3.3. The simulations at the periselene produced almost identical results, and are presented in Appendix A.3.1 for brevity.



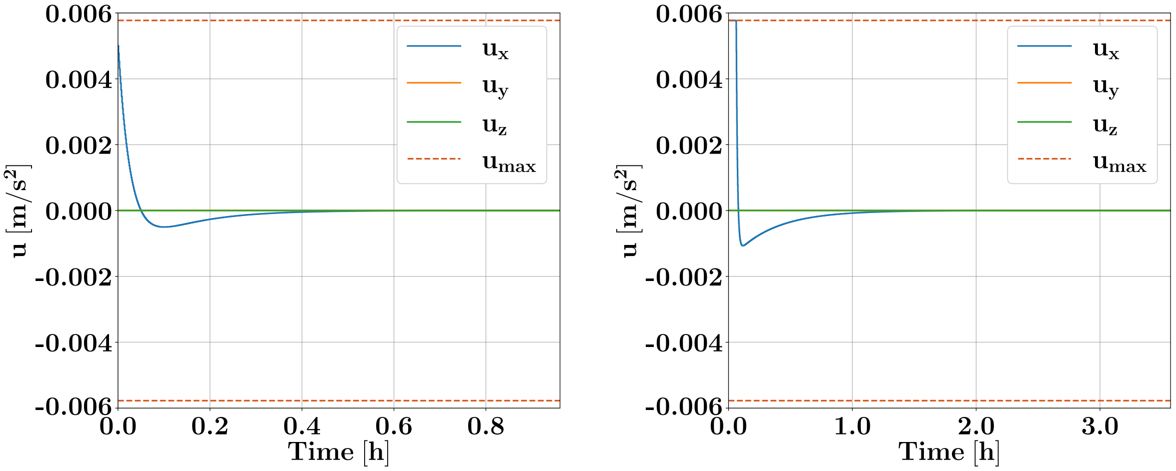
(a) Short range.    (b) Medium range.



(c) Long range.

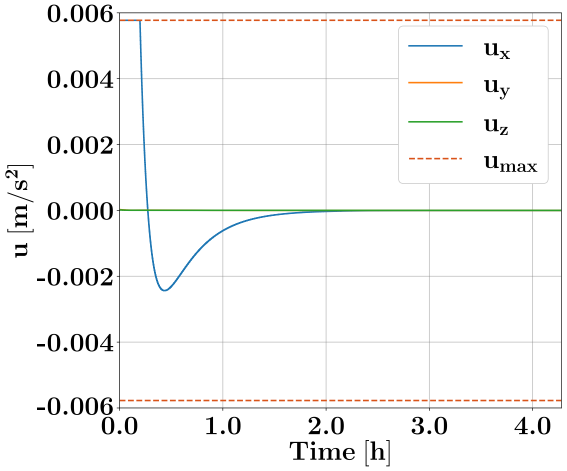
Figure 3.3: Fixed $T_s$ simulations. DAMPC trajectories at the aposelene in the LVLH reference frame $\mathcal{L}$. The blue line depicts the nominal case, while the fainter lines correspond to the trajectories obtained from displaced initial conditions; the target, located at the origin, is marked with a square.

Figure 3.4 shows the control inputs for the nominal case of the short, medium and long range simulations at the aposelene; as expected, due to the use of a quadratic cost function, the control actions are smooth and do not show any discontinuous behavior. At the beginning of the simulations, the controller applies a strong input along V-bar to gain velocity and move the chaser toward the target, with the control saturating at its maximum value in the medium and long range cases. Subsequently, the control input reverses direction to decelerate the chaser and gently guide it into the soft docking condition described in Chapter 2.1. Similar control patterns are observed in the simulations at the periselene and, as with the trajectories, and are presented in Appendix A.3.1 for brevity.



(a) Short range.

(b) Medium range.



(c) Long range.

Figure 3.4: Fixed $T_s$ simulations. DAMPC control inputs at the aposelene, in the LVLH reference frame $\mathcal{L}$.

Figure 3.5 shows the time of flight for all simulations at both the aposelene and periselene. The docking is accomplished within a few hours, with longer distances requiring more time to complete the rendezvous; moreover, the differences between the aposelene and periselene simulations are minimal, with only the long range case showing a noticeable variation. Table 3.13 summarizes the average time of flight across all simulations.

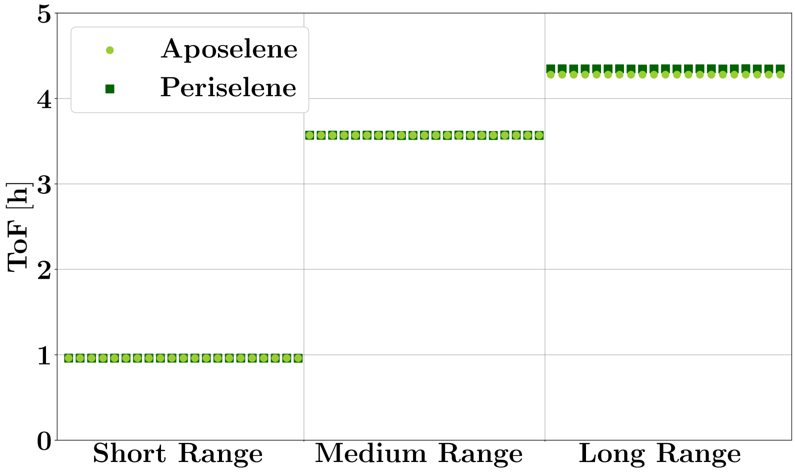
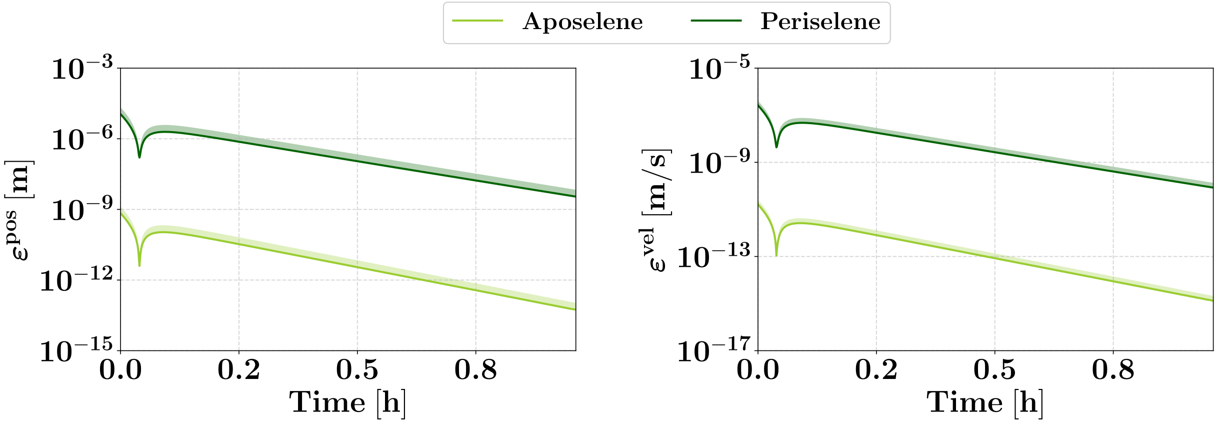


Figure 3.5: Fixed $T_s$ simulations. DAMPC time of flight.

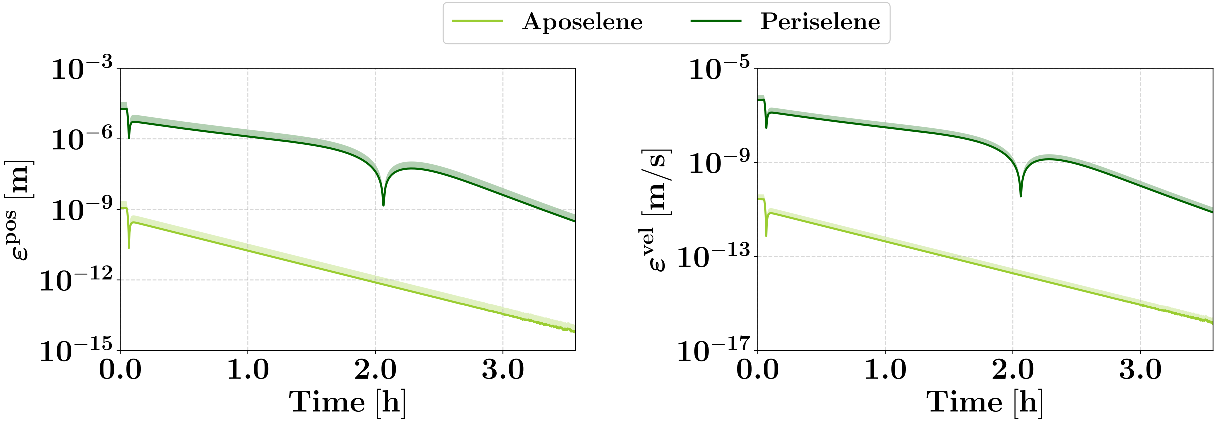
| | **Short Range** | **Medium Range** | **Long Range** |
|---|---|---|---|
| Aposelene | 0.9611 h | 3.5700 h | 4.2789 h |
| Periselene | 0.9610 h | 3.5696 h | 4.3466 h |

Table 3.13: Fixed $T_s$ simulations. DAMPC average time of flight.

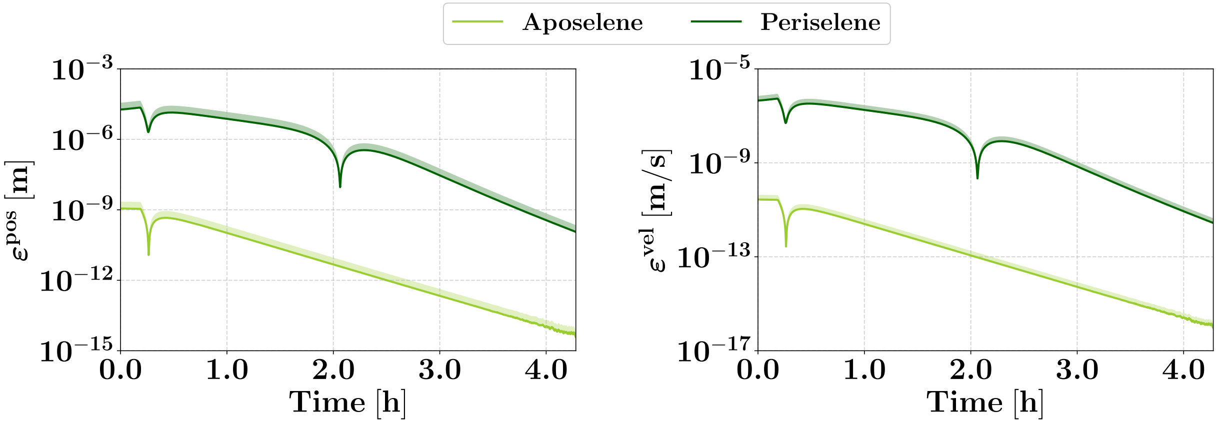
With regard to the required evaluation metrics, the accuracy is discussed first. In Figure 3.6, the absolute prediction errors for the nominal simulations, at both aposelene and periselene, are reported on a logarithmic scale. In general, all simulations show similar trends, with both position and velocity errors decreasing as time progresses. This behavior can be explained by looking at the relative errors in Figure 3.7, computed as the ratio of position and velocity errors to their corresponding predicted values, and expressed as percentages; these are quite stable throughout the simulations, indicating that as the chaser approaches the target, the absolute errors decrease proportionally with distance and velocity. Furthermore, the prediction errors at the periselene are four to five orders of magnitude higher than at the aposelene, confirming again that nonlinearities are more pronounced in this location.



(a) Short range.

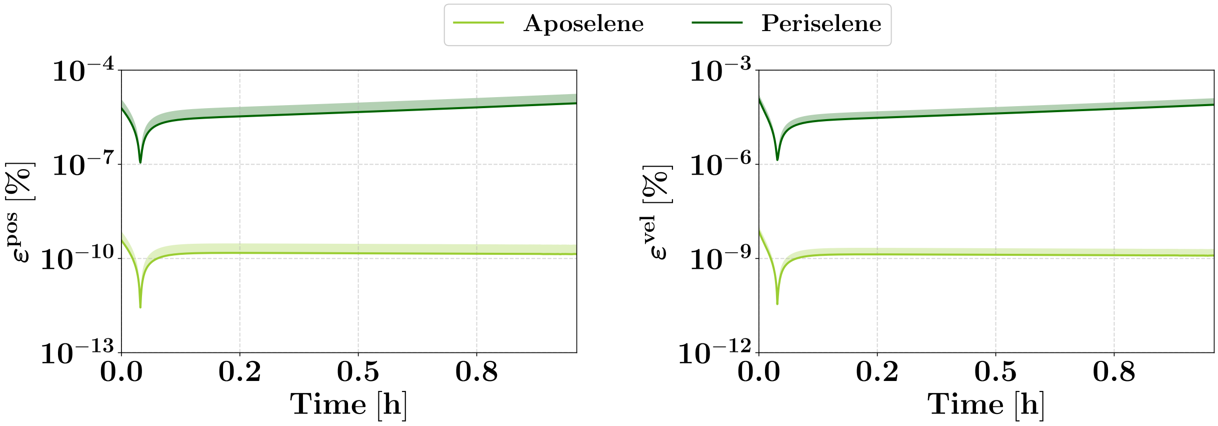


(b) Medium range.

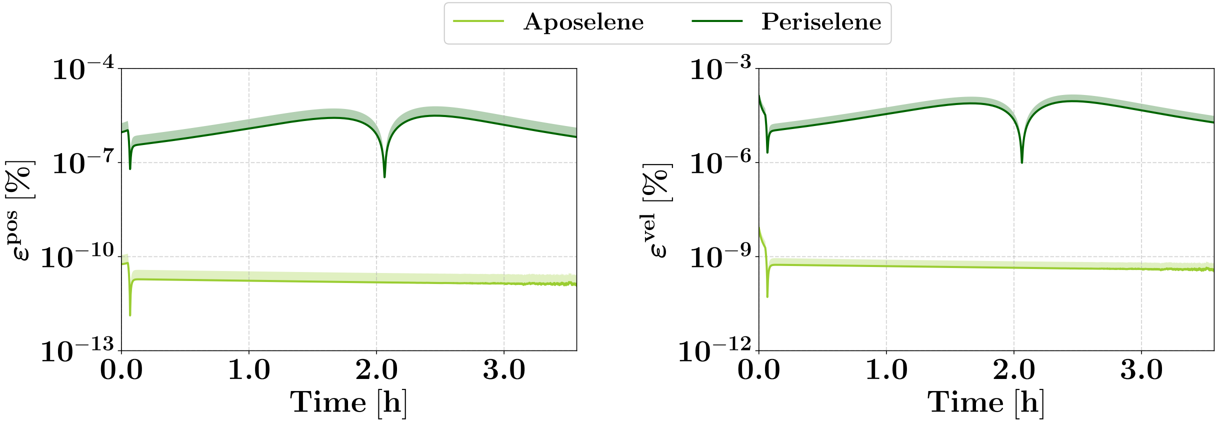


(c) Long range.

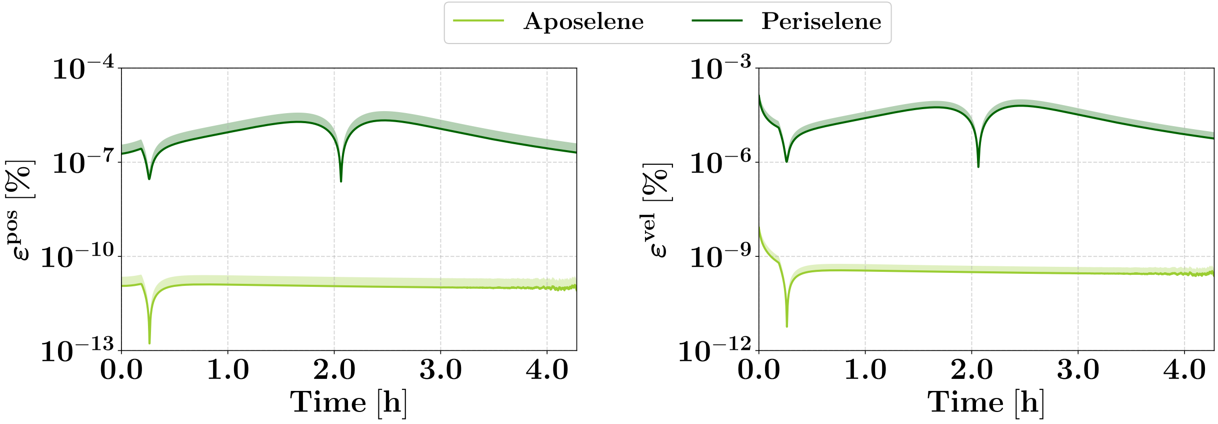
Figure 3.6: Fixed $T_s$ simulations. DAMPC absolute prediction errors for the nominal simulations, at both aposelene and periselene. The bold line shows the mean errors and the shaded area denotes the standard deviation. Left: position error, right: velocity error.



(a) Short range.



(b) Medium range.



(c) Long range.

Figure 3.7: Fixed $T_s$ simulations. DAMPC relative prediction errors for the nominal simulations, at both aposelene and periselene. The bold line shows the mean errors and the shaded area denotes the standard deviation. Left: position error, right: velocity error.

Regarding optimality, Figure 3.8 shows the $\Delta v$ required for each simulation at both the aposelene and periselene. In general, only a few m/s are needed, with the periselene simulations requiring slightly higher $\Delta v$ than their aposelene counterparts; this is due to the stronger gravitational forces and higher velocities in that region, which require the chaser to execute more intense maneuvers to reduce the distance and reach the target. Moreover, as the distance increases, the maneuver cost also rises, and the differences between the aposelene and periselene cases become more pronounced. The numerical averages are reported in Table 3.14.

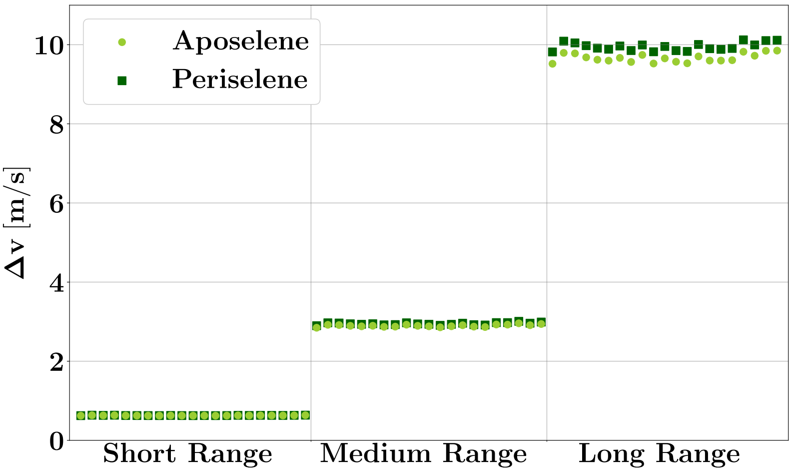


Figure 3.8: Fixed $T_s$ simulations. DAMPC maneuver cost.

| | **Short Range** | **Medium Range** | **Long Range** |
|---|---|---|---|
| Aposelene | 0.627644 m/s | 2.897577 m/s | 9.666391 m/s |
| Periselene | 0.629142 m/s | 2.947471 m/s | 9.953707 m/s |

Table 3.14: Fixed $T_s$ simulations. DAMPC average maneuver cost.

The average computational costs and standard deviations for the overall optimization are presented in Figure 3.9, while those required solely for the generation of the polynomial maps, along with the computation of their derivatives, are reported in Figure 3.10. On average, 57.6 ms are required to execute one optimization, of which 22.7 ms are spent on computing the polynomial maps.

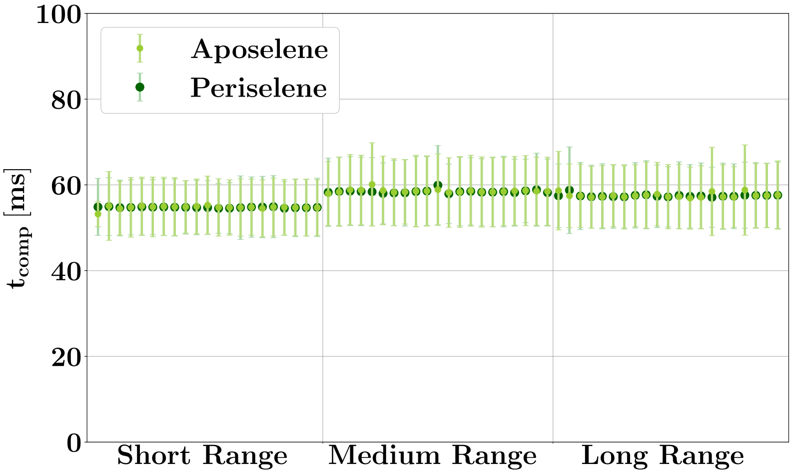


Figure 3.9: Fixed $T_s$ simulations. DAMPC computational cost for the overall optimization.

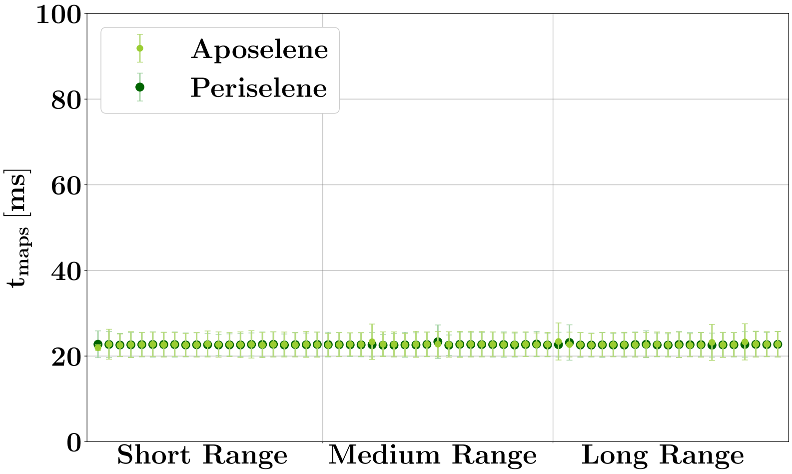
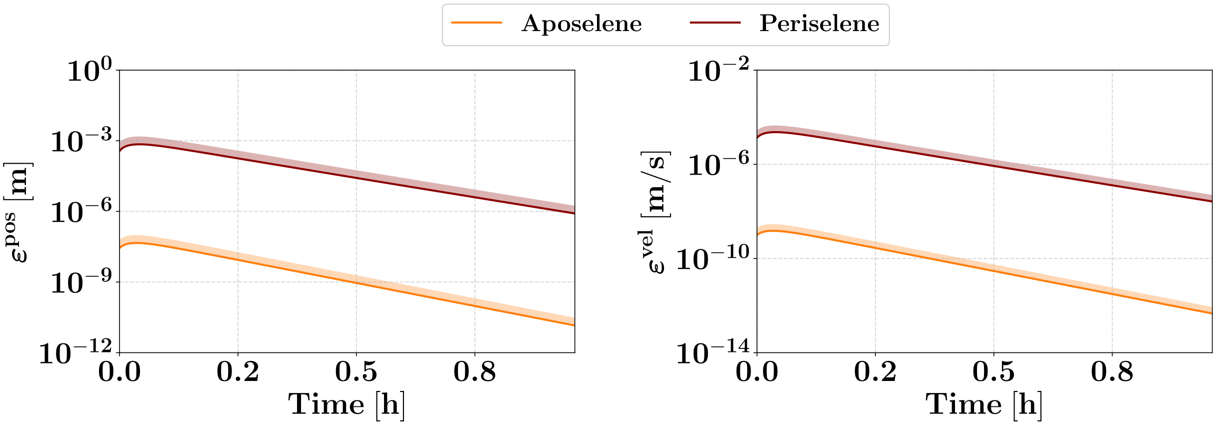


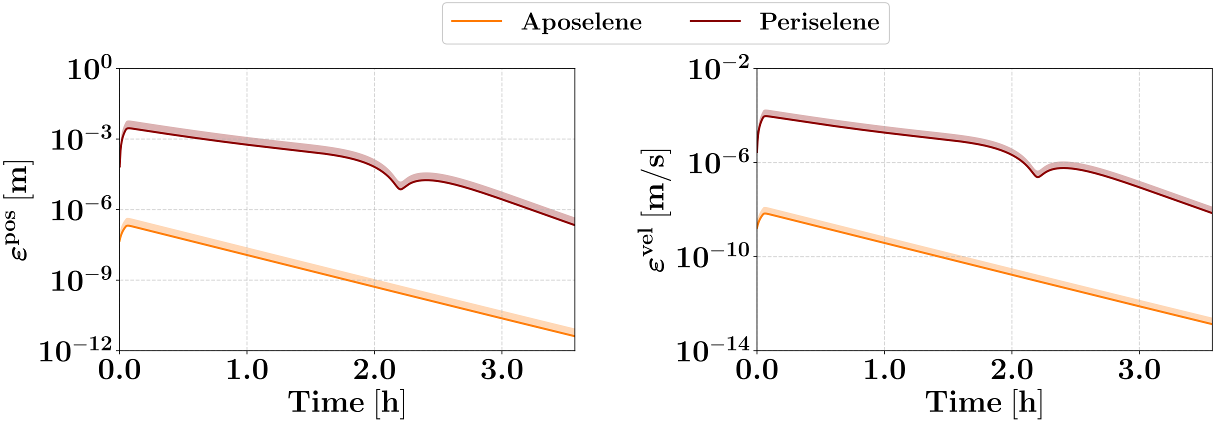
Figure 3.10: Fixed $T_s$ simulations. DAMPC computational cost for generating the polynomial maps.

### 3.3.3. Simulation Results - LMPC

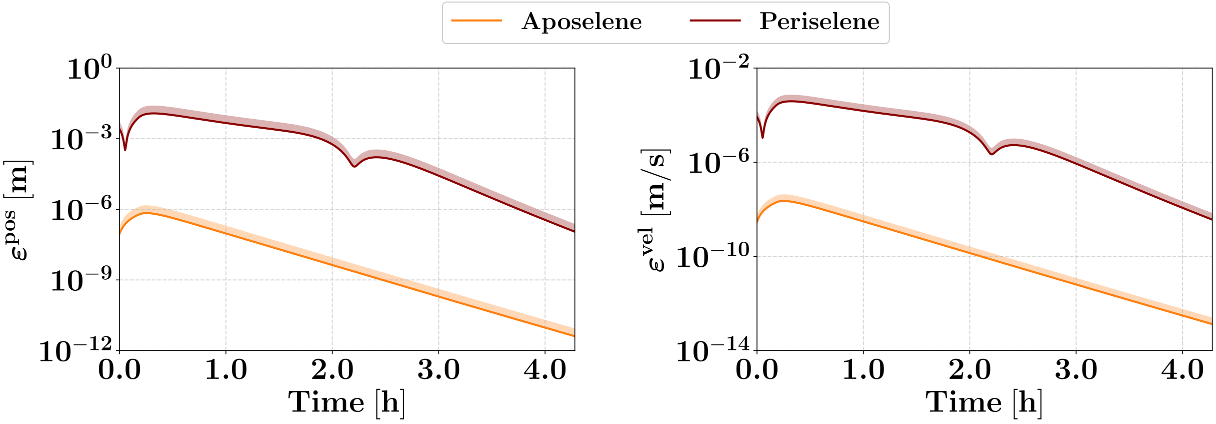
The LMPC produces solutions nearly identical to those obtained with the DAMPC. Regarding trajectories, control inputs and time of flight, the same results observed for the DAMPC are confirmed here, and are presented in Appendix A.3.2 for brevity. Here, only the results related to the three required evaluation metrics will be addressed.
The prediction accuracy exhibits a trend similar to that observed in the DAMPC: errors decrease as the simulation progresses, with relative errors remaining fairly stable; moreover, the errors at the periselene are again four to five orders of magnitude higher than those at the aposelene. The results are presented in Figure 3.11 for the absolute errors, and in Figure 3.12 for the relative errors.



(a) Short range.

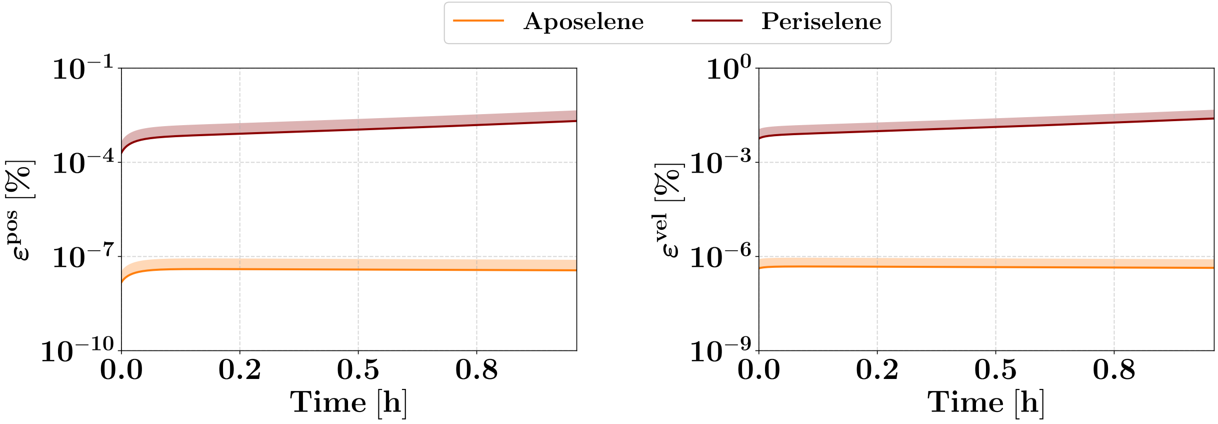


(b) Medium range.

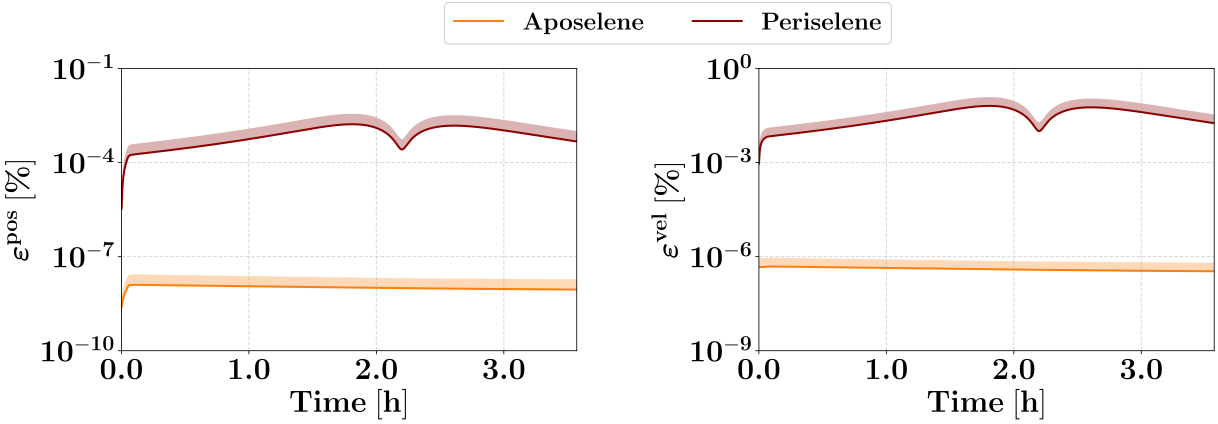


(c) Long range.

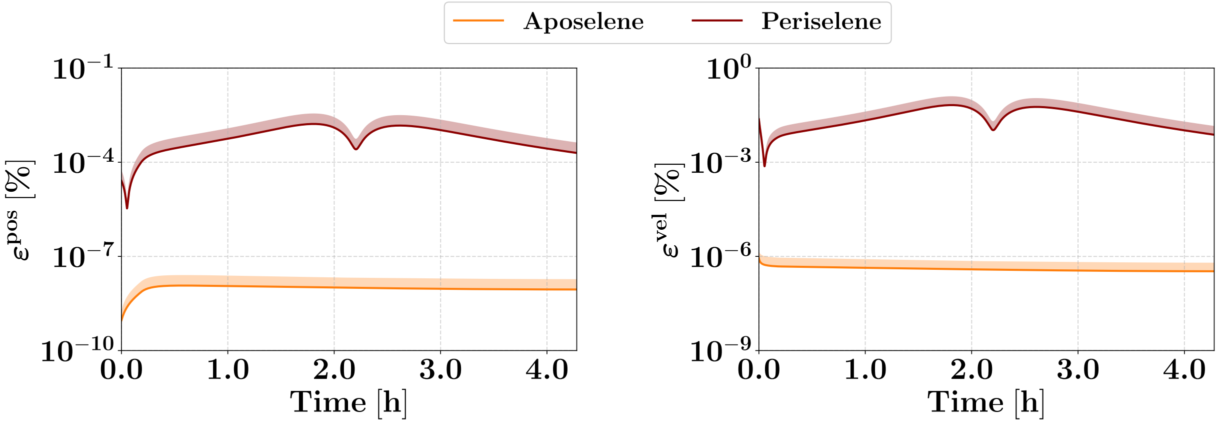
Figure 3.11: Fixed $T_s$ simulations. LMPC absolute prediction errors for the nominal simulations, at both aposelene and periselene. The bold line shows the mean errors and the shaded area denotes the standard deviation. Left: position error, right: velocity error.



(a) Short range.



(b) Medium range.



(c) Long range.

Figure 3.12: Fixed $T_s$ simulations. LMPC relative prediction errors for the nominal simulations, at both aposelene and periselene. The bold line shows the mean errors and the shaded area denotes the standard deviation. Left: position error, right: velocity error.

Since the control inputs and trajectories remain nearly unchanged, the $\Delta v$ values are almost equivalent to those required by the DAMPC, as shown in Figure 3.13; the averages are reported in Table 3.15.

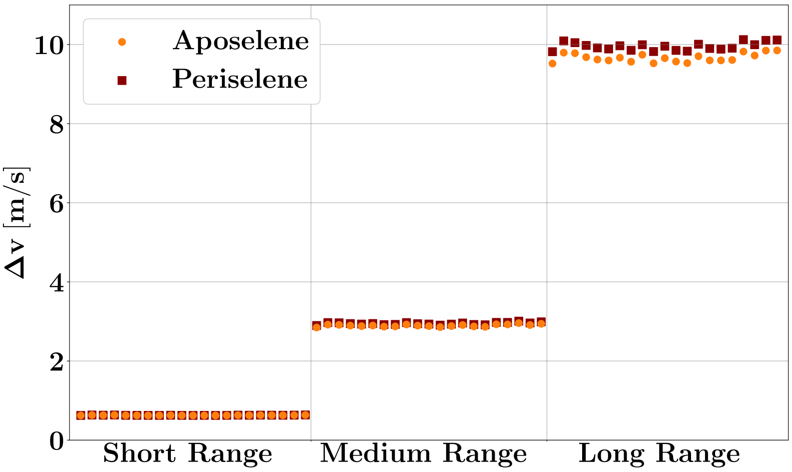


Figure 3.13: Fixed $T_s$ simulations. LMPC maneuver cost.

| | Short Range | Medium Range | Long Range |
|---|---|---|---|
| Aposelene | 0.627631 m/s | 2.897565 m/s | 9.666391 m/s |
| Periselene | 0.629141 m/s | 2.947486 m/s | 9.954426 m/s |

Table 3.15: Fixed $T_s$ simulations. LMPC average maneuver cost.

Finally, the computational costs, reported in Figure 3.14, tend to remain quite stable throughout the simulations, resulting in an average of 16.1 ms.

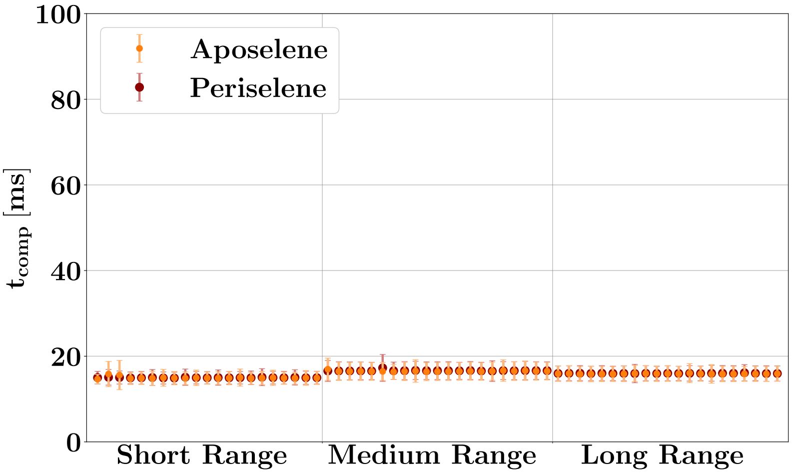
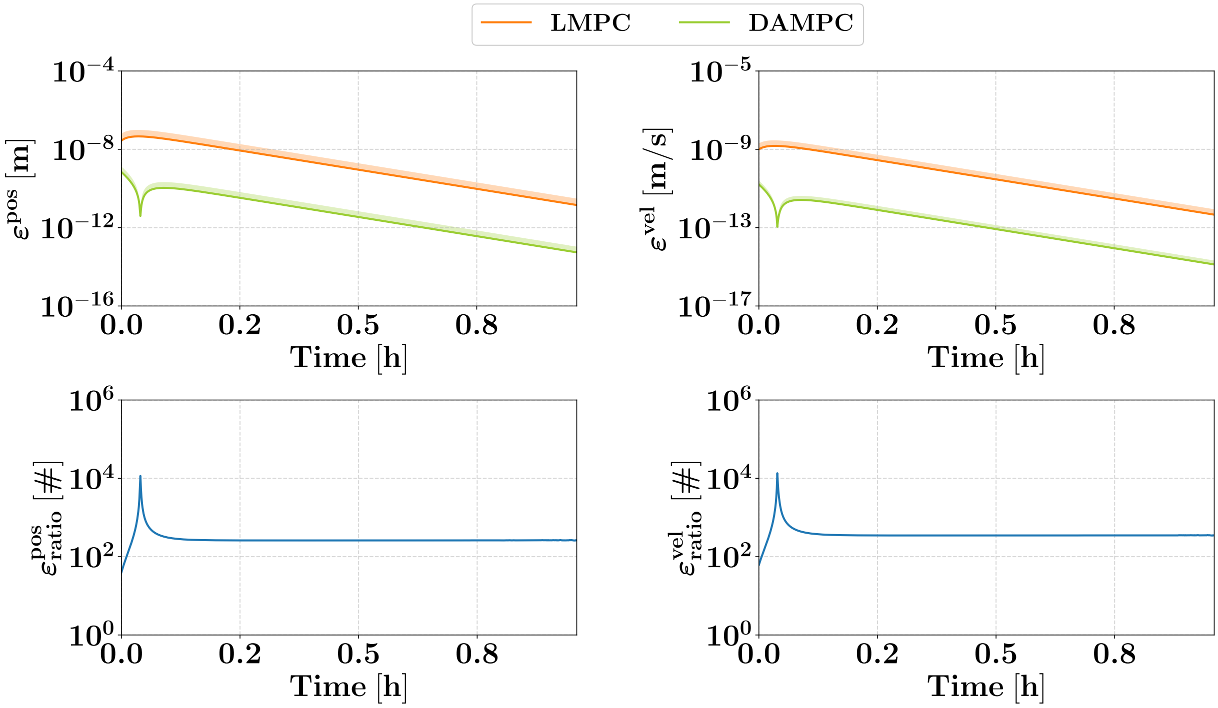


Figure 3.14: Fixed $T_s$ simulations. LMPC computational cost for the overall optimization.

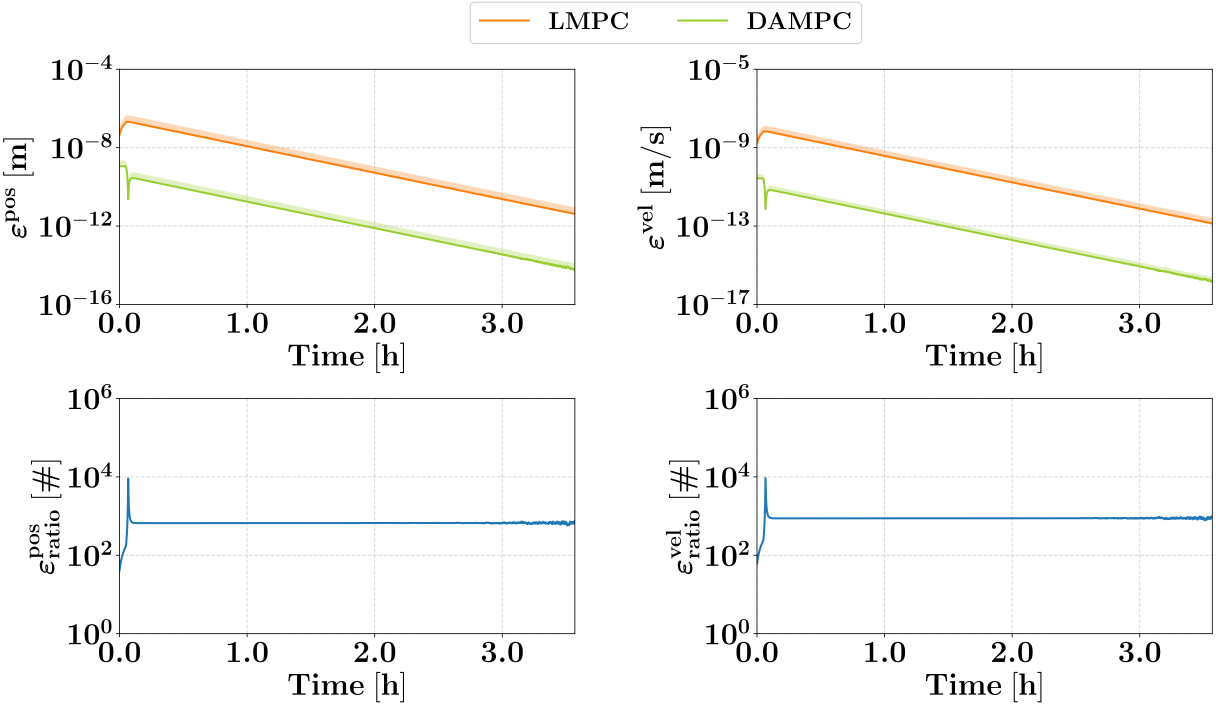
### 3.3.4. Simulation Results - DAMPC vs LMPC

As already mentioned, the two controllers tend to yield the same results in terms of trajectory and control inputs, resulting in a similar level of optimality; on average, the percentage difference in $\Delta v$ between the DAMPC and LMPC is on the order of 0.001%, demonstrating that there is no significant difference between them for this metric in the present simulation setup.

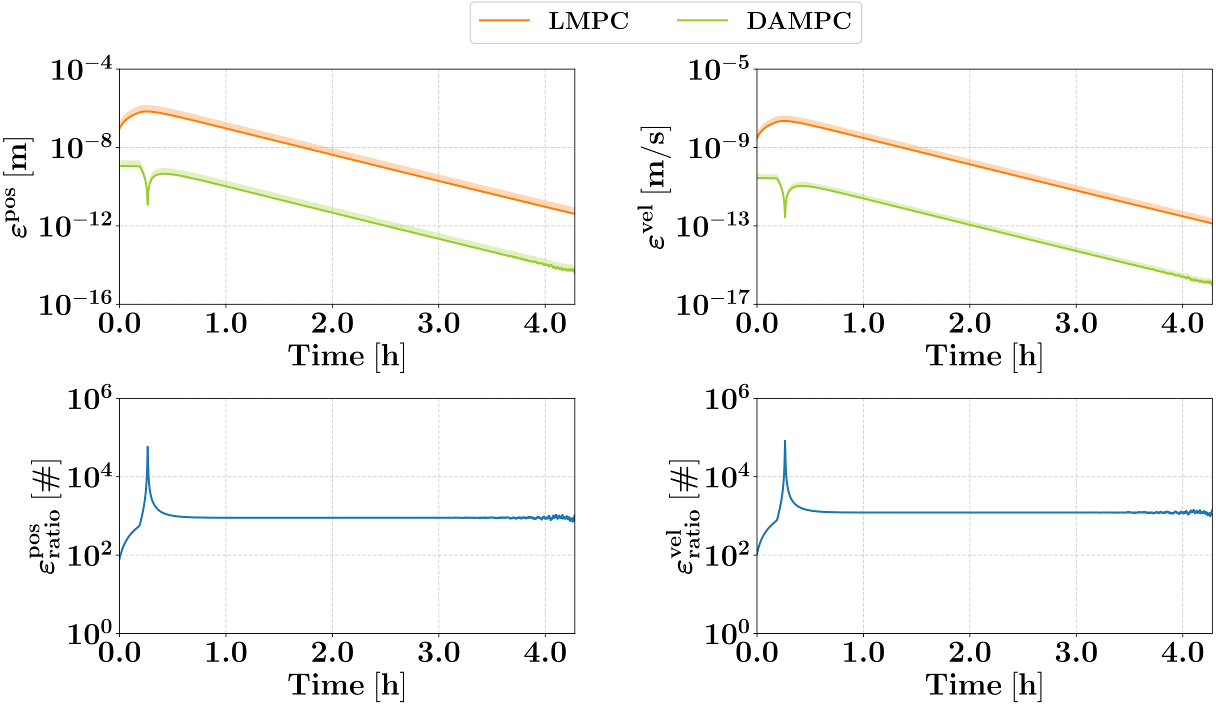
Conversely, the main differences lie in the computational cost and prediction accuracy. Regarding the former, as expected, the DAMPC is more demanding than the LMPC, being approximately 3.5 times slower. With regard to the prediction accuracy, the DAMPC demonstrates its main advantage by providing more accurate predictions, effectively reducing uncertainties in the modeling of the dynamics. Below, a direct comparison is presented by plotting the errors of both the DAMPC and LMPC for a clear visual comparison; furthermore, the ratio of the errors committed by the LMPC to those committed by the DAMPC is shown to quantify the difference between the two. This comparison is shown for all three nominal simulations at both the aposelene (see Figure 3.15) and periselene (see Figure 3.16).



(a) Short range.

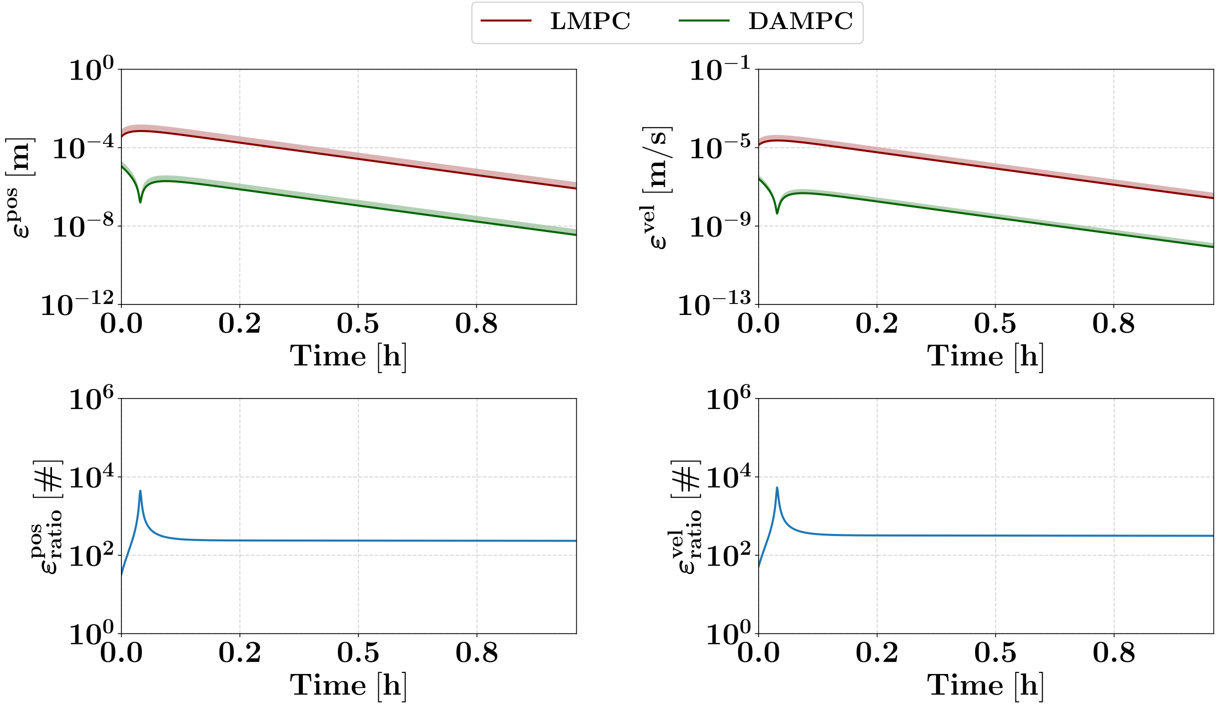


(b) Medium range.

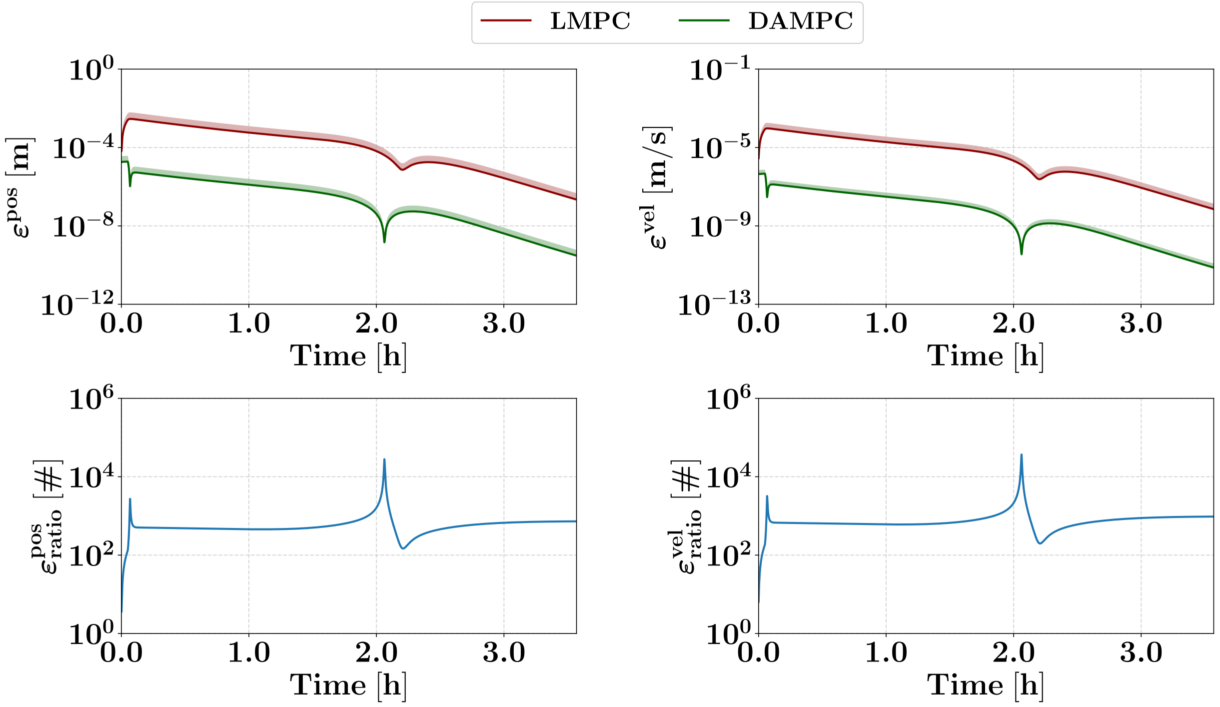


(c) Long range.

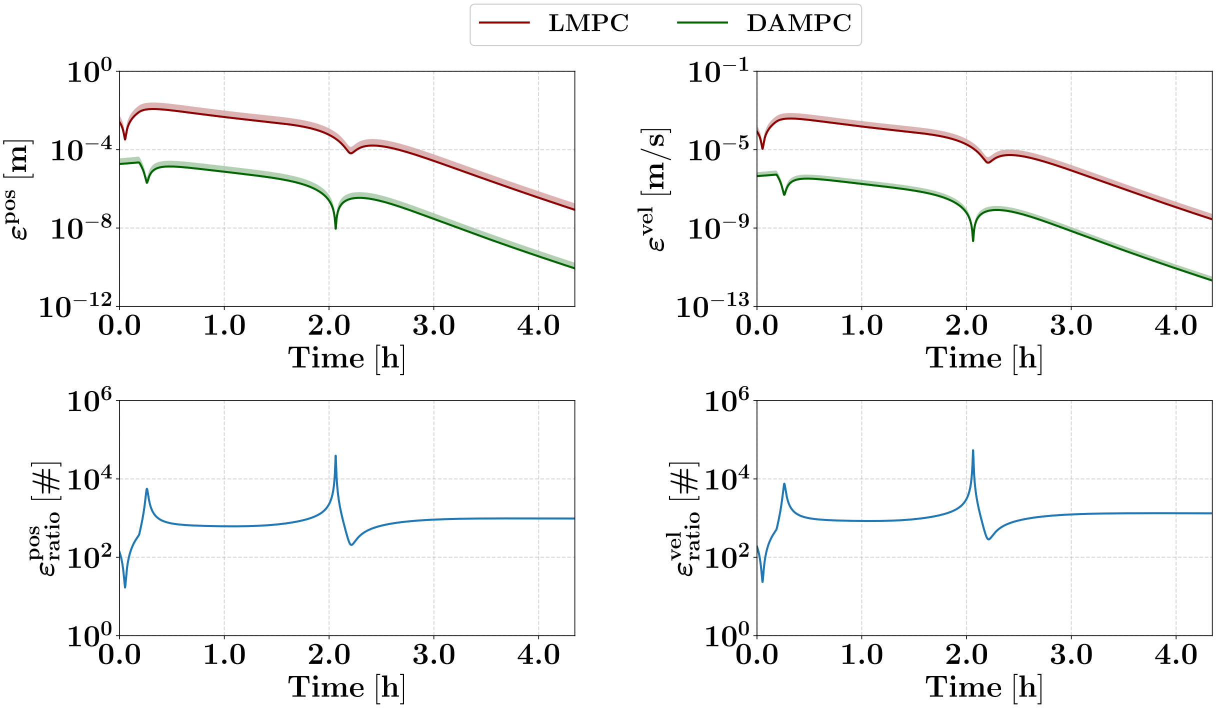
Figure 3.15: Fixed $T_s$ simulations. DAMPC vs LMPC accuracy comparison at aposelene. Top row: absolute errors in position (left) and velocity (right). Bottom row: ratio of LMPC errors to DAMPC errors for position (left) and velocity (right).



(a) Short range.



(b) Medium range.

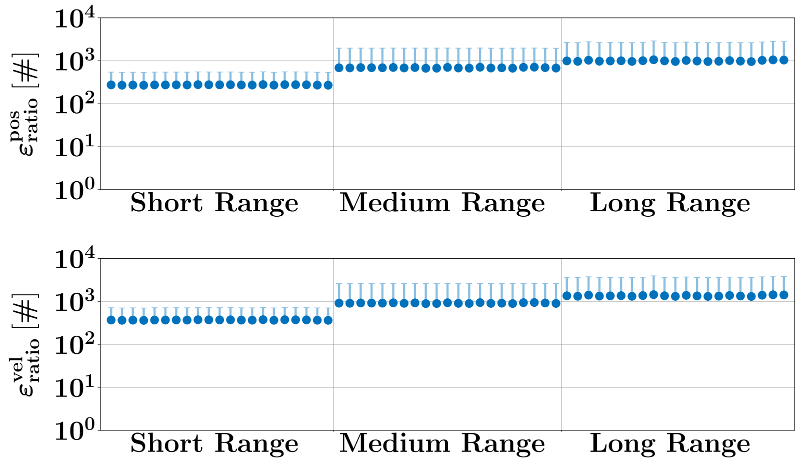


(c) Long range.

Figure 3.16: Fixed $T_s$ simulations. DAMPC vs LMPC accuracy comparison at periselene. Top row: absolute errors in position (left) and velocity (right). Bottom row: ratio of LMPC errors to DAMPC errors for position (left) and velocity (right).

In general, as already observed in Chapter 3.2, some cusp points appear in the prediction errors of the DAMPC; as previously explained, these occur in specific cases where the predicted trajectory intersects the exact one, causing the prediction errors to approach zero and thus indicating moments in which the DAMPC is much more accurate than the LMPC. Outside these cases, the error ratio typically ranges from $10^2$ to $10^3$, showing that the DAMPC is generally two to three orders of magnitude more accurate than the LMPC. To assess whether the observed results are systematic or merely a fortunate coincidence of the nominal simulations, the error ratios have been computed for all simulations, with Figure 3.17 presenting the mean and standard deviation of these curves at both the aposelene and periselene. Overall, the observations described above are confirmed, indicating that the higher accuracy of the DAMPC is systematic. Furthermore, the difference in accuracy between the two controllers increases with distance, with the DAMPC showing progressively better performance as the simulations start from farther initial conditions. Regarding the standard deviations, these are strongly influenced by the number and magnitude of the cusp points. For instance, in the aposelene simulations for the medium range case (see Figure 3.15), a single cusp point of moderate magnitude is present, resulting in a very small standard deviation. In contrast, in the corresponding periselene simulations (see Figure 3.16), the same cusp point is present together with a second one of higher magnitude, leading to a larger standard deviation. Similar patterns are also observed in the other cases.

(a) Aposelene.



(b) Periselene.

Figure 3.17: Fixed $T_s$ simulations. DAMPC vs LMPC accuracy comparison across all simulations. Top: mean position error with its standard deviation, bottom: mean velocity error with its standard deviation.

## 3.4. Rendezvous Simulations - Variable Sampling Time

### 3.4.1. Simulation Setup

In this set of simulations, a different perspective is adopted, aiming to exploit the higher accuracy of the DAMPC to reduce the overall $\Delta v$. The underlying idea is that a longer prediction horizon enables a more effective optimization, thereby lowering the maneuver cost. Two parameters can be adjusted to achieve this: the number of predicted steps $N$ and the sampling time $T_s$. A higher $N$ increases the computational cost, as more degrees of freedom are involved in the OCP, and is therefore avoided. In contrast, increasing $T_s$ extends the prediction horizon without affecting the number of variables; however, this also amplifies the prediction errors, particularly when using linearized dynamics. If the model is not sufficiently accurate, the predicted trajectories may diverge significantly from reality, and the control inputs optimized on this basis could produce suboptimal or unexpected behavior, creating the necessity for corrections in subsequent iterations. Conversely, when the model is highly accurate, as in the case of the DAMPC, the predicted trajectory remains closer to the real one, making the optimization more effective and reducing the necessity for such corrections. The present simulations apply this concept to assess whether the higher accuracy of the DAMPC results in better optimization compared to the LMPC.

For this set of simulations, the same initial conditions used previously are employed; however, only the *long range* scenario is considered (see Table 3.9). As before, 21 simulations are conducted: in the first, the chaser is perfectly aligned with the V-bar, referred to as the *nominal case*, while the remaining 20 include displacements along the other two directions; the analysis is conducted at both the aposelene and periselene, resulting in a total of 42 simulations. At the beginning the sampling time is set to $T_s = 400$ s, yielding a prediction horizon of over 3 hours. As the simulation progresses and the distance decreases, more frequent adjustments are required to maintain the level of accuracy necessary for completing the rendezvous, and therefore the sampling time is reduced. When the distance along the V-bar decreases to 2 000 m the sampling time is set to 40 s, and for the final 200 m it is further reduced to 4 s. The gains are also adapted according to the distance, resulting in the MPC settings reported in Table 3.16, 3.17 and 3.18.

With regard to the evaluation metrics, this analysis focuses on the prediction errors and the optimality of the solution, evaluated through the $\Delta v$ required by the rendezvous; regarding the computational cost, the same results as previously observed were obtained and are omitted here for compactness.

| $x > 2\,000$ m | |
|---|---|
| $T_s$ | 400 s |
| $N$ | 30 |
| $M$ | 15 |
| $Q_{pos}$ | $10^{13}$ |
| $Q_{vel}$ | $10^{8}$ |
| $R$ | $10^{2}$ |

Table 3.16: Variable $T_s$ simulations. Controller parameters for distances over 2 000 m.

| $2\,000 \text{ m} \geq x > 200$ m | |
|---|---|
| $T_s$ | 40 s |
| $N$ | 30 |
| $M$ | 15 |
| $Q_{pos}$ | $10^{13}$ |
| $Q_{vel}$ | $10^{8}$ |
| $R$ | $10^{1}$ |

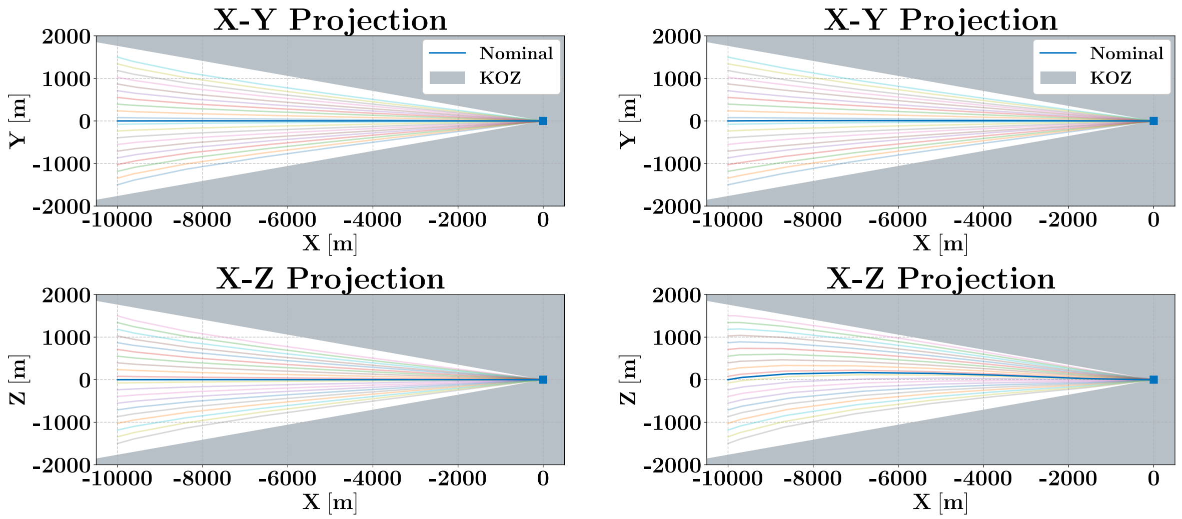
Table 3.17: Variable $T_s$ simulations. Controller parameters for distances between 2 000 and 200 m.

| $200 \text{ m} \geq x$ | |
|---|---|
| $T_s$ | 4 s |
| $N$ | 30 |
| $M$ | 15 |
| $Q_{pos}$ | $10^{13}$ |
| $Q_{vel}$ | $10^{8}$ |
| $R$ | $10^{0}$ |

Table 3.18: Variable $T_s$ simulations. Controller parameters for distances below 200 m.

## 3.4.2. Simulation Results - DAMPC vs LMPC

Many of the results previously obtained are confirmed in these simulations, therefore the analysis will focus directly on the comparison of the two controllers without presenting the findings separately. The resulting trajectories are shown in Figure 3.18, generated using the solutions obtained with the DAMPC; the trajectories obtained with the LMPC yielded nearly identical results and are presented in Appendix A.4 for brevity.

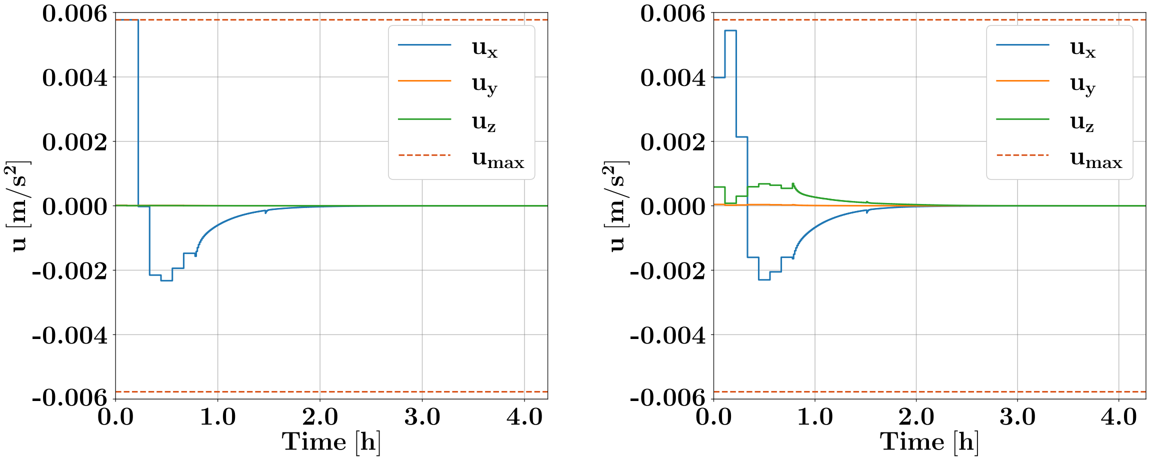


(a) Aposelene.

(b) Periselene.

Figure 3.18: Variable $T_s$ simulations. DAMPC trajectories in the LVLH reference frame $\mathcal{L}$, obtained from the simulations with variable sampling time. The blue line depicts the nominal case, while the fainter lines correspond to those obtained from displaced initial conditions; the target, located at the origin, is marked with a square.

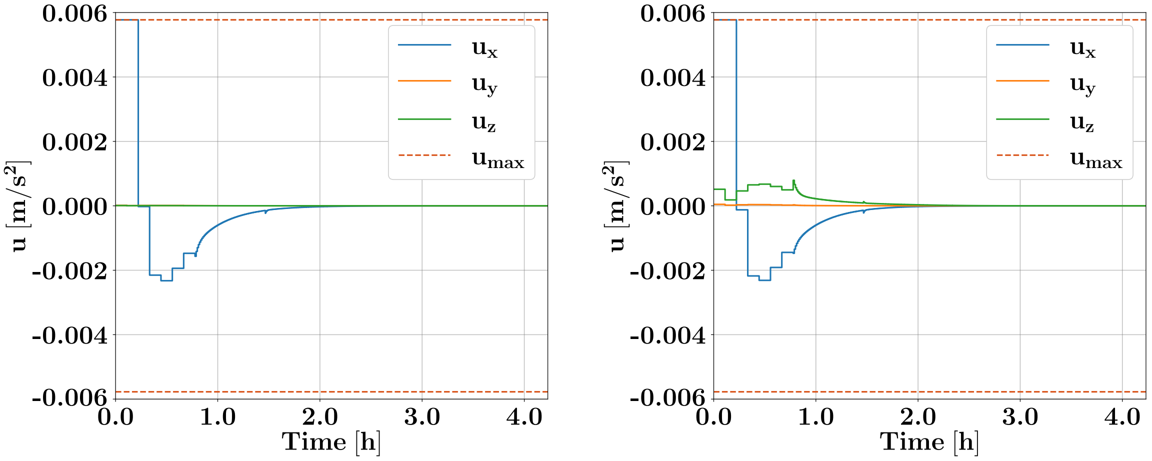
The first difference between the two controllers appears in the control inputs. Figure 3.19 shows them for the nominal simulation at both the aposelene and periselene, obtained with the DAMPC, while Figure 3.20 shows the same but obtained with the LMPC. Firstly, it can be observed that for almost the entire first hour the control exhibits clearly the piecewise constant behavior characteristic of the MPC, which is evident here due to the long sampling time; afterwards, it becomes smoother, indicating a reduction of the time interval. Comparing the control inputs at the aposelene, no significant discrepancies are observed between the two controllers; however, in the simulations at the periselene, the differences are evident, with the DAMPC generating much less intense inputs and avoiding saturation, in contrast to the behavior observed with the LMPC.



(a) Aposelene.

(b) Periselene.

Figure 3.19: Variable $T_s$ simulations. DAMPC control inputs, in the LVLH reference frame $\mathcal{L}$.

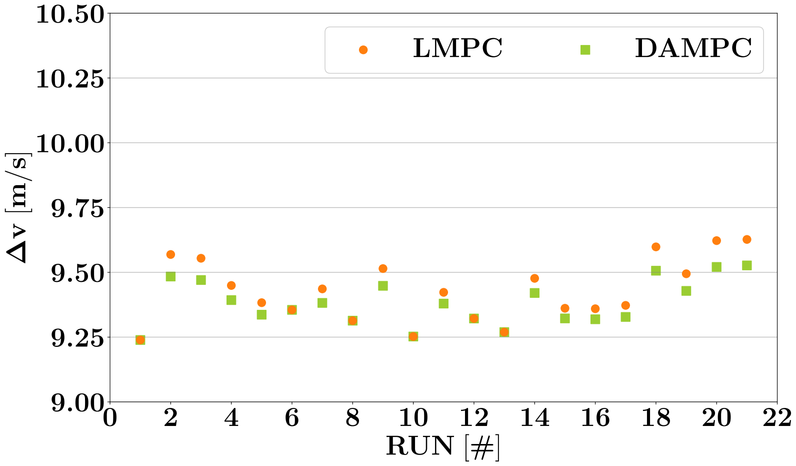


(a) Aposelene.

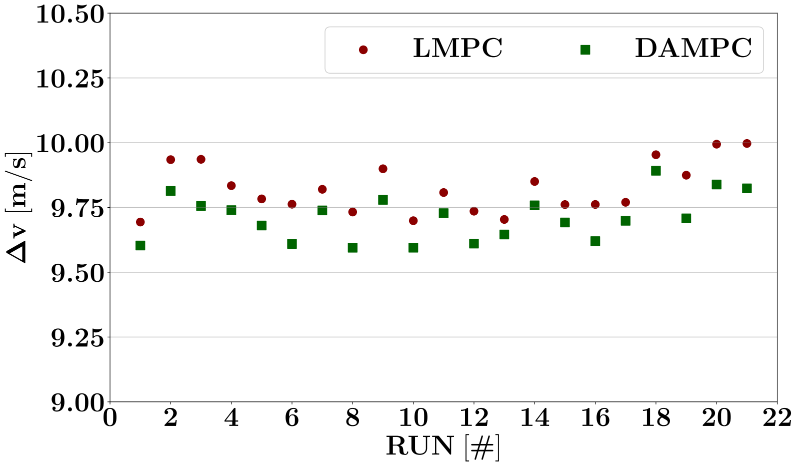
(b) Periselene.

Figure 3.20: Variable $T_s$ simulations. LMPC control inputs, in the LVLH reference frame $\mathcal{L}$.

Figure 3.21 shows the maneuver cost, expressed in terms of $\Delta v$, required by the two controllers at both the aposelene and periselene. In both cases, the cost is reduced compared to the previous simulations performed with a fixed sampling time of 4 s, confirming that a longer prediction horizon can lead to better optimizations. Furthermore, the DAMPC consistently requires slightly lower $\Delta v$, demonstrating that its higher accuracy enables more effective optimizations when very long sampling times are employed. At the aposelene, the maneuver cost obtained with the DAMPC is on average 0.49 % lower than that of the LMPC, while at the periselene this difference increases to 1.15 %. The numerical averages are reported in Table 3.19.



(a) Aposelene.



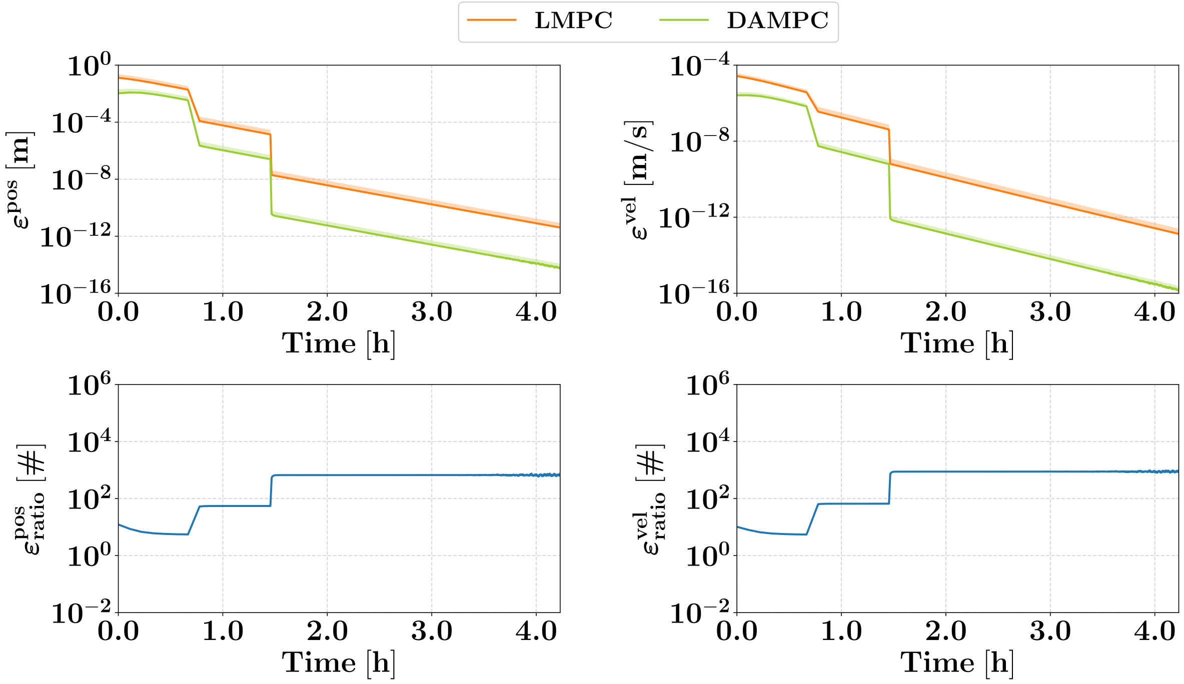
(b) Periselene.

Figure 3.21: Variable $T_s$ simulations. DAMPC vs LMPC maneuver cost comparison for all simulations with variable sampling time.

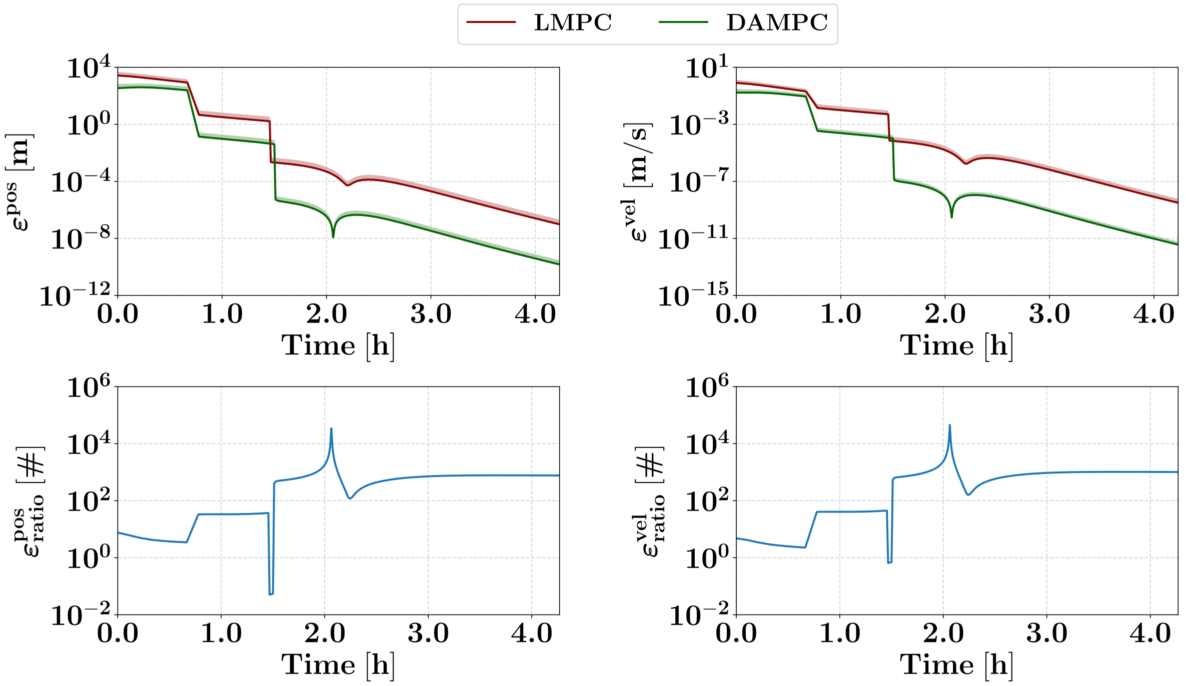
| | **DAMPC** | **LMPC** |
|---|---|---|
| Aposelene | 9.381688 m/s | 9.428194 m/s |
| Periselene | 9.710792 m/s | 9.824093 m/s |

Table 3.19: Variable $T_s$ simulations. DAMPC vs LMPC average maneuver cost comparison.

Similarly to the previous simulations, Figure 3.22 presents a direct assessment of the prediction accuracy by plotting the errors of both the DAMPC and LMPC for a clear visual comparison; additionally, the ratio of the errors produced by the LMPC to those of the DAMPC is shown to quantify the difference between the two. In general, both controllers are negatively affected by the longer sampling times. All curves exhibit two abrupt changes in the prediction errors for both position and velocity, corresponding to the time instants where the sampling time is reduced: first from 400 s to 40 s, and then from 40 s to 4 s. By examining the error ratio, it is evident that the DAMPC is consistently more accurate than the LMPC. However, in the error ratio plots at the periselene, a sharp drop appears at around one and a half hours into the simulation; this occurs because the change in sampling time is not perfectly synchronized between the two controllers: for a brief interval, the DAMPC still operates with $T_s = 40$ s while the LMPC has already switched to $T_s = 4$ s, temporarily making the DAMPC less accurate.



(a) Medium range.

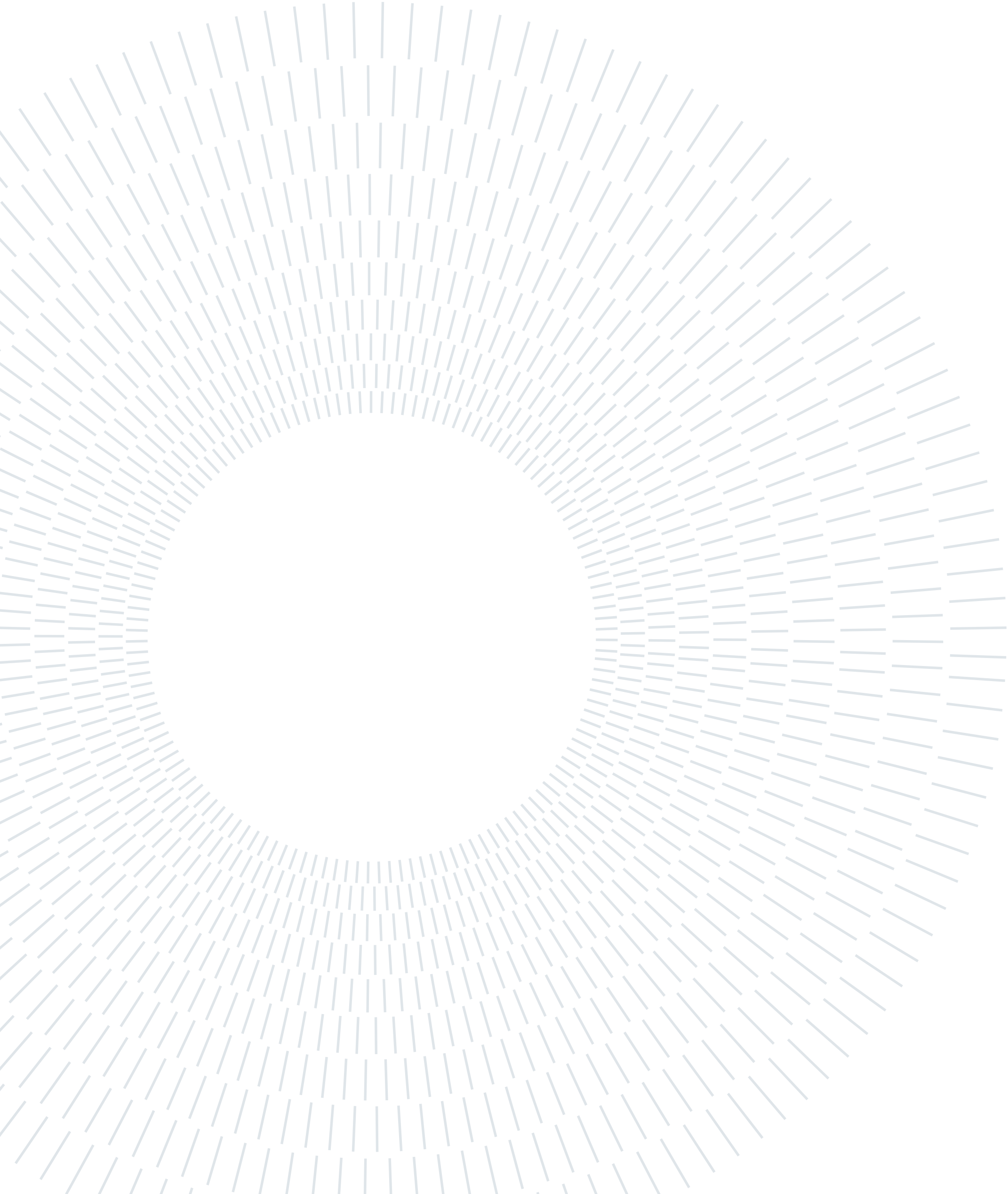


(b) Long range.

Figure 3.22: Variable $T_s$ simulations. DAMPC vs LMPC accuracy comparison for the simulations with variable sampling time. Top row: absolute errors in position (left) and velocity (right). Bottom row: ratio of LMPC errors to DAMPC errors for position (left) and velocity (right).

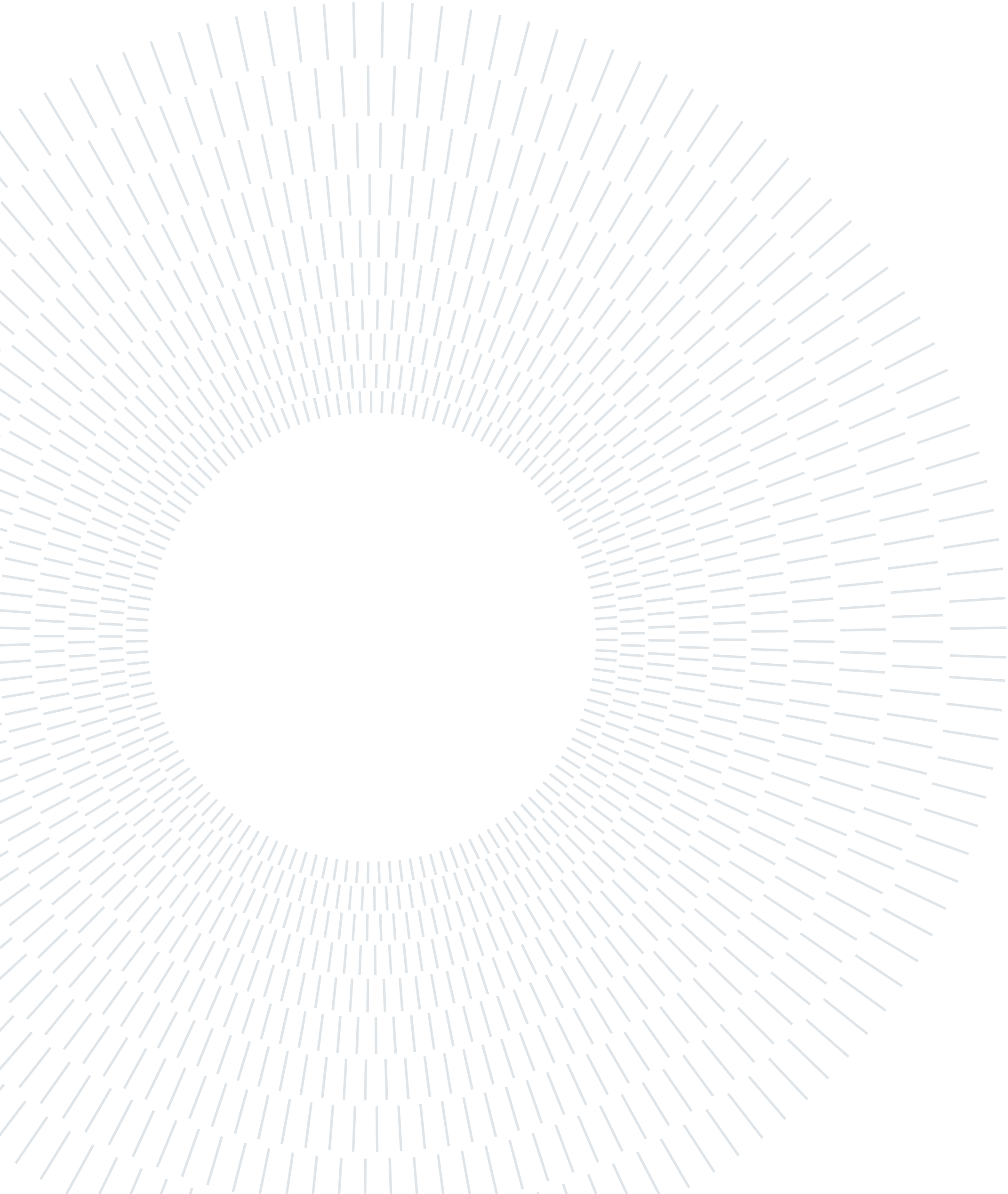
The error ratio further shows that although the DAMPC is consistently more accurate than the LMPC, it is also more sensitive to increments in the sampling time. When $T_s = 400$ s, the DAMPC is roughly one order of magnitude more accurate, with $T_s = 40$ s this difference grows to two orders of magnitude, and finally with $T_s = 4$ s it reaches three orders of magnitude, matching the accuracy levels observed in the fixed sampling time simulations discussed earlier. There are mainly two reasons that can justify this behavior. The first is related to how the control input is incorporated into the dynamics. As discussed in Chapter 2.2, differential algebra is applied only to expand the free dynamics, while the control term is introduced through the control matrix $\boldsymbol{B_k}$, the same used in the LMPC. This matrix is computed under the assumption that the system can be modeled as LTI for sufficiently small sampling times, allowing a single $\boldsymbol{B_k}$ to be used across the entire prediction horizon; however, when the sampling time is large, such as the 400 s used at the beginning of these simulations, this assumption no longer holds, and a more accurate approach would be to recompute $\boldsymbol{B_k}$ at each time instant within the prediction horizon. Because this is not done with the current implementation, the errors become dominated by the control contribution, reducing the advantages of the DAMPC. Consequently, for long sampling times the errors of both controllers tend to converge.

The second reason is most likely related to the strategy employed for generating the polynomials. As previously discussed, the accuracy of the maps is strongly influenced by the magnitude of the displacement between the predicted states and the expansion points. These last are taken from the initial guess, which comes from the output of the previous optimization. The method assumes that this displacement is small, as the new solution is expected to be close to the previously computed optimal one. While this assumption remains valid for longer sampling times, it must be noted that with very long prediction horizon the region of space in which the optimal solution may lie becomes much larger. As a result, the optimizer may still converge to a solution close to the initial guess but with larger displacements with respect to what would occur with a smaller sampling time. Since these displacements are used when evaluating the polynomial maps, the predictions become less accurate. To address this issue, more sophisticated strategies could be adopted for generating the maps, such as recomputing all polynomials at each iteration or mapping a larger region of space using multiple polynomial expansions, allowing the selection of the most appropriate one based on the predicted states. Both approaches would increase the computational cost, but depending on the sampling time, the requirement for very fast optimizations could be relaxed.



# 4 | Conclusions and future developments

To conclude, this work demonstrates how DA can be effectively embedded within an MPC framework to provide an accurate representation of the system dynamics in highly nonlinear environments. The proposed DAMPC was validated through a series of rendezvous simulations with a target placed on an NRHO, at both the aposelene and periselene, and the results were compared against those obtained with a classical LMPC. The simulations with a fixed sampling time showed that, although the DAMPC requires higher computational resources, it can significantly reduce prediction errors compared to the LMPC. This improvement does not directly translate into lower maneuver costs, since for small sampling times the linearized model remains sufficiently accurate. Nevertheless, it must be noted that the simulations were performed in a favorable environment where no additional sources of uncertainty were considered. A first possible advancement would be to explore the implementation of this approach within a robust MPC framework, like the TRMPC, while also accounting for different disturbances, such as the solar radiation pressure. In this context, having a more accurate representation of the system dynamics could prove particularly valuable. The simulations with variable sampling times demonstrated that the DAMPC can achieve better optimization when employing a longer prediction horizon, resulting in lower maneuver costs compared to the LMPC. Nevertheless, these simulations also raise some issues, mostly related to the treatment of the control input and the generation of the polynomial maps. To address these problems, more sophisticated strategies for performing the polynomial expansion of the system dynamics could be employed. One possible solution would be to perform the expansions not only with respect to the initial conditions but also with respect to the control, removing the necessity of using the control input matrix to incorporate this contribution; although this would certainly increase the computational cost of generating the maps, as more variables would be required, it would also lead to significantly higher accuracy. Finally, a more advanced strategy for selecting the expansion points is needed to ensure an accurate representation of the system dynamics, even when very long prediction horizons are considered.



# Bibliography

[1] James A. Chamberlin and James T. Rose. Gemini rendezvous program. *Journal of Spacecraft and Rockets*, 1(1):13–18, 1964. doi: 10.2514/3.27585. URL `https://doi.org/10.2514/3.27585`.

[2] J. R. Burton and W. E. Hayes. Gemini rendezvous. *Journal of Spacecraft and Rockets*, 3(1):145–147, 1966. doi: 10.2514/3.59526. URL `https://doi.org/10.2514/3.59526`.

[3] John L. Goodman. History of space shuttle rendezvous and proximity operations. *Journal of Spacecraft and Rockets*, 43(5):944–959, 2006. doi: 10.2514/1.19653. URL `https://doi.org/10.2514/1.19653`.

[4] NASA. Artemis plan: Nasa's lunar exploration program overview. `https://www.nasa.gov/wp-content/uploads/2020/12/artemis_plan-20200921.pdf`, September 2020.

[5] Ryan Whitley and Roland Martinez. Options for staging orbits in cislunar space. In *2016 IEEE Aerospace Conference*, pages 1–9, 2016. doi: 10.1109/AERO.2016.7500635.

[6] David E. Lee. Gateway destination orbit model: A continuous 15 year nrho reference trajectory. Technical report, NASA, August 2019.

[7] Giordana Bucchioni and Mario Innocenti. Rendezvous in cis-lunar space near rectilinear halo orbit: Dynamics and control issues. *Aerospace*, 8(3), 2021. ISSN 2226-4310. URL `https://www.mdpi.com/2226-4310/8/3/68`.

[8] Giovanni Franzini and Mario Innocenti. Relative motion dynamics in the restricted three-body problem. *Journal of Spacecraft and Rockets*, 56(5):1322–1337, 2019. doi: 10.2514/1.A34390. URL `https://doi.org/10.2514/1.A34390`.

[9] Mongrard, O., Cavrois, B., Ankersen, F., Dubois-Matra, O., Zink, M., Vergnol, A., Piquemal, E., Pionnier, G., and Southivong, U. Atv gnc flight performance and lessons learned. In Array, editor, *EUCASS Book Series – Advances in AeroSpace*

*Sciences*, volume 10, pages 217–238, 2018. doi: 10.1051/eucass/201810217. URL `https://doi.org/10.1051/eucass/201810217`.

[10] Timothy E. Rumford. Demonstration of autonomous rendezvous technology (DART) project summary. In Peter Tchoryk Jr. and James Shoemaker, editors, *Space Systems Technology and Operations*, volume 5088, pages 10 – 19. International Society for Optics and Photonics, SPIE, 2003. doi: 10.1117/12.498811. URL `https://doi.org/10.1117/12.498811`.

[11] Martina Mammarella, Elisa Capello, and Giorgio Guglieri. A comprehensive analysis of guidance and control algorithms for orbital rendezvous maneuvers. In *AIAA/AAS Astrodynamics Specialist Conference*. AIAA, 2016. doi: 10.2514/6.2016-5214. URL `https://arc.aiaa.org/doi/abs/10.2514/6.2016-5214`.

[12] Elisa Capello, Fabrizio Dabbene, Giorgio Guglieri, and Elisabetta Punta. “flyable” guidance and control algorithms for orbital rendezvous maneuver. *SICE Journal of Control, Measurement, and System Integration*, 11(1):14–24, 2018. doi: 10.9746/jcmsi.11.14. URL `https://doi.org/10.9746/jcmsi.11.14`.

[13] Mauro Mancini, Nicoletta Bloise, Elisa Capello, and Elisabetta Punta. Sliding mode control techniques and artificial potential field for dynamic collision avoidance in rendezvous maneuvers. *IEEE Control Systems Letters*, 4(2):313–318, 2020. doi: 10.1109/LCSYS.2019.2926053.

[14] Michele Galullo, Giordana Bucchioni, Giovanni Franzini, and Mario Innocenti. Closed loop guidance during close range rendezvous in a three body problem. *The Journal of the Astronautical Sciences*, 69(1):28–50, jan 2022. doi: 10.1007/s40295-021-00289-6. URL `http://dx.doi.org/10.1007/s40295-021-00289-6`.

[15] Lu Cao, Dong Qiao, and Jingwen Xu. Suboptimal artificial potential function sliding mode control for spacecraft rendezvous with obstacle avoidance. *Acta Astronautica*, 143:133–146, 2018. ISSN 0094-5765. doi: https://doi.org/10.1016/j.actaastro.2017.11.022. URL `https://www.sciencedirect.com/science/article/pii/S0094576517307701`.

[16] Edward N. Hartley. A tutorial on model predictive control for spacecraft rendezvous. In *2015 European Control Conference (ECC)*, pages 1355–1361, 2015. doi: 10.1109/ECC.2015.7330727.

[17] S. {Di Cairano}, H. Park, and I. Kolmanovsky. Model predictive control approach for guidance of spacecraft rendezvous and proximity maneuvering. *International Journal*

*of Robust and Nonlinear Control*, 22(12):1398–1427, August 2012. ISSN 1049-8923. doi: 10.1002/rnc.2827.

[18] M. Mammarella, E. Capello, H. Park, G. Guglieri, and M. Romano. Tube-based robust model predictive control for spacecraft proximity operations in the presence of persistent disturbance. *Aerospace Science and Technology*, 77:585–594, 2018. ISSN 1270-9638. doi: https://doi.org/10.1016/j.ast.2018.04.009. URL `https://www.sciencedirect.com/science/article/pii/S1270963817321223`.

[19] Martina Mammarella, Elisa Capello, and Giorgio Guglieri. Robust model predictive control for automated rendezvous maneuvers in near-earth and moon proximity. In *2018 AIAA SPACE and Astronautics Forum and Exposition*. AIAA, 2018. doi: 10.2514/6.2018-5343. URL `https://arc.aiaa.org/doi/abs/10.2514/6.2018-5343`.

[20] Julio C. Sanchez, Francisco Gavilan, and Rafael Vazquez. Chance-constrained model predictive control for near rectilinear halo orbit spacecraft rendezvous. *Aerospace Science and Technology*, 100:105827, 2020. ISSN 1270-9638. doi: https://doi.org/10.1016/j.ast.2020.105827. URL `https://www.sciencedirect.com/science/article/pii/S127096381932348X`.

[21] Michele Pagone, Giordana Bucchioni, Francesco Alfino, and Carlo Novara. A minimum-propellant pontryagin-based nonlinear mpc for spacecraft rendezvous in lunar orbit: the extended version, 2023. URL `https://arxiv.org/abs/2309.05453`.

[22] Michele Pagone, Mattia Boggio, Carlo Novara, Anton Proskurnikov, and Giuseppe C. Calafiore. A penalty function approach to constrained pontryagin-based nonlinear model predictive control. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 3705–3710, 2022. doi: 10.1109/CDC51059.2022.9992438.

[23] Joseph Fels Ritt. *Integration in Finite Terms Liouville's Theory of Elementary Methods*, pages 31–134. Springer International Publishing, Cham, 2022. ISBN 978-3-030-98767-1. doi: 10.1007/978-3-030-98767-1_3. URL `https://doi.org/10.1007/978-3-030-98767-1_3`.

[24] Joseph Fels Ritt. *Differential equations from the algebraic standpoint.* Colloquium Publications. American mathematical society, New York, online-ausg edition, 1932. ISBN 9780821846056.

[25] Joseph Fels Ritt. *Differential algebra.* Colloquium Publications. American Mathematical Society, Providence, Rhode Island, 1950. ISBN 9780821832059.

[26] M. Berz. Differential algebraic treatment of beam dynamics to very high orders

including applications to spacecharge. *AIP Conference Proceedings*, 177(1):275–300, 12 1988. ISSN 0094-243X. doi: 10.1063/1.37800. URL `https://doi.org/10.1063/1.37800`.

[27] P. Di Lizia, R. Armellin, and M. Lavagna. Application of high order expansions of two-point boundary value problems to astrodynamics. *Celestial Mechanics and Dynamical Astronomy*, 102(4):355–375, December 2008. ISSN 0923-2958, 1572-9478. doi: 10.1007/s10569-008-9170-5. URL `http://link.springer.com/10.1007/s10569-008-9170-5`.

[28] P. Di Lizia, R. Armellin, A. Morselli, and F. Bernelli-Zazzera. High order optimal feedback control of space trajectories with bounded control. *Acta Astronautica*, 94(1):383–394, 2014. ISSN 0094-5765. doi: https://doi.org/10.1016/j.actaastro.2013.02.011. URL `https://www.sciencedirect.com/science/article/pii/S0094576513000593`.

[29] P. Di Lizia, R. Armellin, F. Bernelli-Zazzera, and M. Berz. High order optimal control of space trajectories with uncertain boundary conditions. *Acta Astronautica*, 93:217–229, 2014. ISSN 0094-5765. doi: https://doi.org/10.1016/j.actaastro.2013.07.007. URL `https://www.sciencedirect.com/science/article/pii/S0094576513002397`.

[30] M. Valli, R. Armellin, P. Di Lizia, and M. R. Lavagna. Nonlinear mapping of uncertainties in celestial mechanics. *Journal of Guidance, Control, and Dynamics*, 36(1):48–63, 2013. doi: 10.2514/1.58068. URL `https://doi.org/10.2514/1.58068`.

[31] Alexander Wittig, Camilla Colombo, and Roberto Armellin. Long-term density evolution through semi-analytical and differential algebra techniques. *Celestial Mechanics and Dynamical Astronomy*, 128(4):435–452, August 2017. ISSN 0923-2958, 1572-9478. doi: 10.1007/s10569-017-9756-x. URL `http://link.springer.com/10.1007/s10569-017-9756-x`.

[32] Francesco Cavenago, Pierluigi Di Lizia, Mauro Massari, and Alexander Wittig. On-board spacecraft relative pose estimation with high-order extended kalman filter. *Acta Astronautica*, 158:55–67, 2019. ISSN 0094-5765. doi: https://doi.org/10.1016/j.actaastro.2018.11.020. URL `https://www.sciencedirect.com/science/article/pii/S0094576518301516`.

[33] Wigbert Fehse. *Automated Rendezvous and Docking of Spacecraft*. Cambridge Aerospace Series. Cambridge University Press, 2003.

[34] William H. Gerstenmaier, David Parker, Gilles Leclerc, and Ryuichiro Shirama. In-

ternational rendezvous system interoperability standards (irsis). Technical Report Baseline, NASA, March 2019.

[35] William H. Gerstenmaier, Sergey Krikalev, David Parker, Gilles Leclerc, and Ryuichiro Shirama. International docking system standard (idss): Interface definition document (idd). Technical Report Revision E, NASA, October 2016.

[36] Charles F. Yoder. *Astrometric and Geodetic Properties of Earth and the Solar System*, pages 1–31. American Geophysical Union (AGU), 1995. ISBN 9781118668078. doi: https://doi.org/10.1029/RF001p0001. URL `https://agupubs.onlinelibrary.wiley.com/doi/abs/10.1029/RF001p0001`.

[37] Afonso Botelho, Baltazar Parreira, Paulo Rosa, and João Lemos. Predictive control for spacecraft rendezvous. *SpringerBriefs in Applied Sciences and Technology*, 01 2021. doi: 10.1007/978-3-030-75696-3.

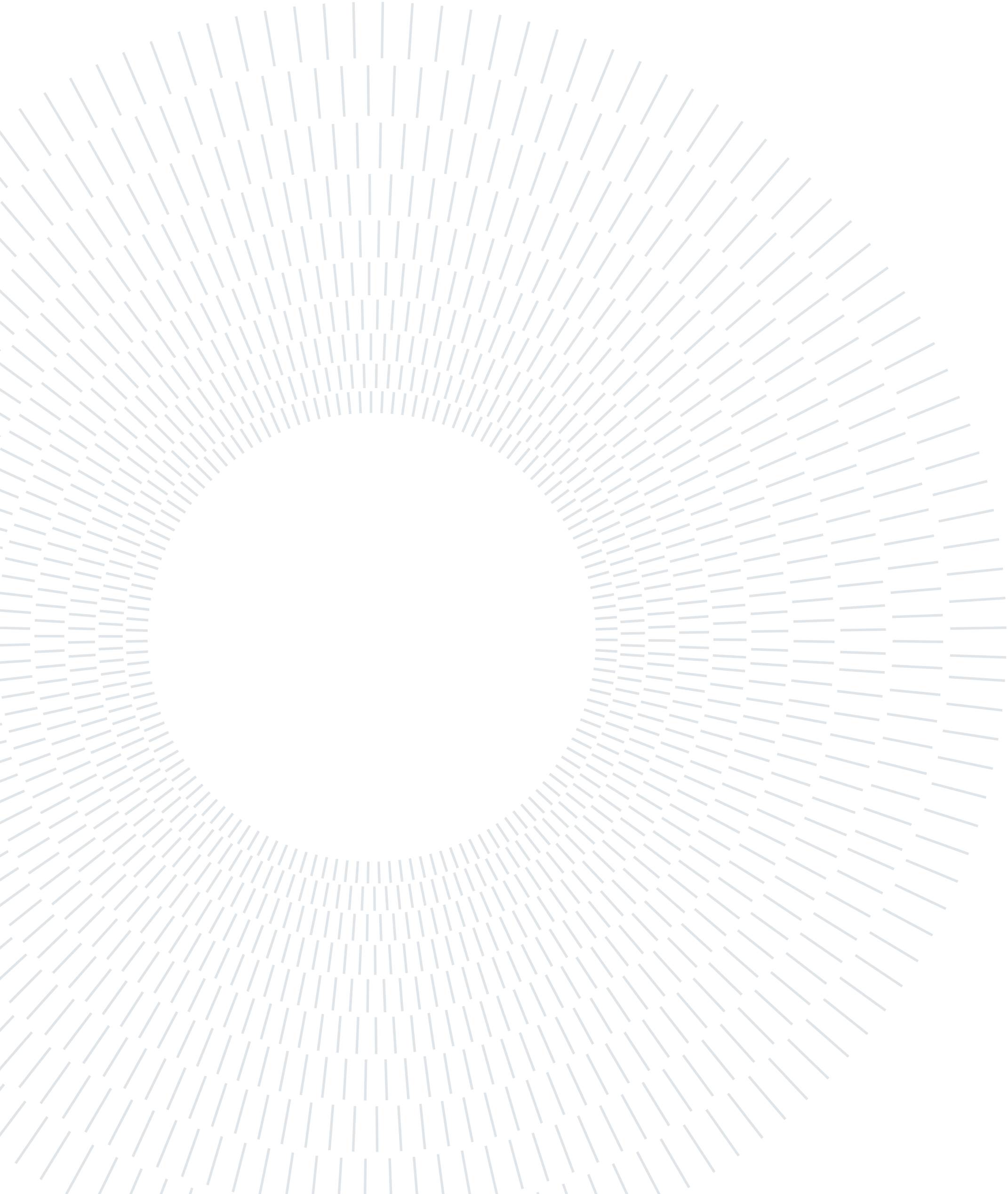
[38] D.Q. Mayne, J.B. Rawlings, C.V. Rao, and P.O.M. Scokaert. Constrained model predictive control: Stability and optimality. *Automatica*, 36(6):789–814, 2000. ISSN 0005-1098. doi: https://doi.org/10.1016/S0005-1098(99)00214-9. URL `https://www.sciencedirect.com/science/article/pii/S0005109899002149`.

[39] Martin Berz. *Modern map methods in particle beam physics*, volume 108 of *Advances in imaging and electron physics*. Academic Press, San Diego, 1999. ISBN 0120147505. URL `https://books.google.it/books?id=W1HKMgEACAAJ`.

[40] Martin Berz and K Makino. *COSY INFINITY 10.2 - Programmer's Manual*. Michigan State University, 04 2023. URL `https://www.bmtdynamics.org/cosy/manual/COSYProgMan102.pdf`.

[41] Mauro Massari, Pierluigi Di Lizia, Francesco Cavenago, and Alexander Wittig. Differential algebra software library with automatic code generation for space embedded applications. *AIAA Journal*, 2018. doi: 10.2514/6.2018-0398. URL `https://arc.aiaa.org/doi/abs/10.2514/6.2018-0398`.

[42] Jason K. Moore. Cyipopt: Python wrapper for the ipopt nonlinear solver. `https://pypi.org/project/cyipopt/`, 2021.



# A | Appendix A

## A.1. LVLH Angular Velocity and Acceleration

In the context of the CNERM the angular velocity and acceleration involved in the computation of the skew symmetric matrices reads as:

$$\boldsymbol{\omega}_{\boldsymbol{l/i}} = \boldsymbol{\omega}_{\boldsymbol{l/m}} + \hat{\boldsymbol{k}}_{\boldsymbol{m}}, \qquad [\dot{\boldsymbol{\omega}}_{\boldsymbol{l/i}}]_{\mathcal{L}} = [\dot{\boldsymbol{\omega}}_{\boldsymbol{l/m}}]_{\mathcal{L}} - \boldsymbol{\omega}_{\boldsymbol{l/m}} \times \hat{\boldsymbol{k}}_{\boldsymbol{m}} \tag{A.1}$$

Here it is reported the set of equation for computing the angular velocity $\boldsymbol{\omega}_{\boldsymbol{l/m}}$ and acceleration $[\dot{\boldsymbol{\omega}}_{\boldsymbol{l/m}}]_{\mathcal{L}}$ of $\mathcal{L}$ with respect to $\mathcal{M}$. The angular velocity is given by:

$$\begin{cases} \omega^x_{l/m} = 0 \\ \omega^y_{l/m} = -\dfrac{h}{r^2} \\ \omega^z_{l/m} = -\dfrac{r}{h^2}\,\boldsymbol{h} \cdot [\ddot{\boldsymbol{r}}]_{\mathcal{M}} \end{cases} \tag{A.2}$$

while the angular acceleration is:

$$\begin{cases} \dot{\omega}^x_{l/m} = 0 \\ \dot{\omega}^y_{l/m} = -\dfrac{1}{r}\left(\dfrac{\dot{h}}{r} + 2\,\dot{r}\,\omega^y_{l/m}\right) \\ \dot{\omega}^z_{l/m} = \left(\dfrac{\dot{r}}{r} - 2\dfrac{\dot{h}}{h}\right)\omega^z_{l/m} - \dfrac{r}{h^2}\boldsymbol{h} \cdot [\dddot{\boldsymbol{r}}]_{\mathcal{M}} \end{cases} \tag{A.3}$$

The target specific angular momentum with respect to the Moon has already been introduced and is given by $\boldsymbol{h} = \boldsymbol{r} \times [\dot{\boldsymbol{r}}]_{\mathcal{M}}$; the norm of its derivative is obtained as:

$$\dot{h} = -[\dot{\boldsymbol{h}}]_{\mathcal{M}} \cdot \hat{\boldsymbol{j}} = -(\boldsymbol{r} \times [\ddot{\boldsymbol{r}}]_{\mathcal{M}}) \cdot \hat{\boldsymbol{j}} \tag{A.4}$$

The derivative of the norm of $\boldsymbol{r}$ is computed as:

$$\dot{r} = \frac{1}{r}\boldsymbol{r} \cdot [\dot{\boldsymbol{r}}]_{\mathcal{M}} \tag{A.5}$$

The target acceleration $[\ddot{\boldsymbol{r}}]_{\mathcal{M}}$ and jerk $[\dddot{\boldsymbol{r}}]_{\mathcal{M}}$, under the assumption of the CRTBP are simplified with respect the most general case, and reads:

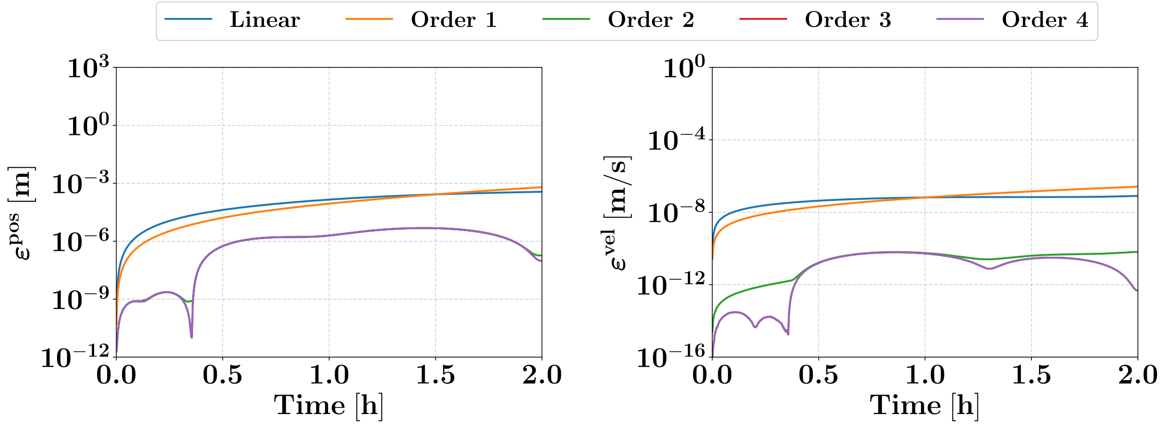
$$\begin{aligned}[\ddot{\boldsymbol{r}}]_{\mathcal{M}} =& -2\boldsymbol{\omega_{m/i}} \times [\dot{\boldsymbol{r}}]_{\mathcal{M}} - \boldsymbol{\omega_{m/i}} \times (\boldsymbol{\omega_{m/i}} \times \boldsymbol{r}) \\ & -\mu\frac{\boldsymbol{r}}{r^3} - (1-\mu)\left(\frac{\boldsymbol{r}+\boldsymbol{r_{em}}}{\|\boldsymbol{r}+\boldsymbol{r_{em}}\|^3} + \frac{\boldsymbol{r_{em}}}{r_{em}^3}\right)\end{aligned} \tag{A.6}$$

$$\begin{aligned}[\dddot{\boldsymbol{r}}]_{\mathcal{M}} =& -2\boldsymbol{\omega_{m/i}} \times [\ddot{\boldsymbol{r}}]_{\mathcal{M}} - \boldsymbol{\omega_{m/i}} \times (\boldsymbol{\omega_{m/i}} \times [\dot{\boldsymbol{r}}]_{\mathcal{M}}) \\ & -\mu\frac{\partial}{\partial\boldsymbol{r}}\left[\frac{\boldsymbol{r}}{r^3}\right][\dot{\boldsymbol{r}}]_{\mathcal{M}} - (1-\mu)\frac{\partial}{\partial\boldsymbol{r}}\left[\frac{\boldsymbol{r}+\boldsymbol{r_{rm}}}{\|\boldsymbol{r}+\boldsymbol{r_{rm}}\|^3}\right][\dot{\boldsymbol{r}}]_{\mathcal{M}}\end{aligned} \tag{A.7}$$

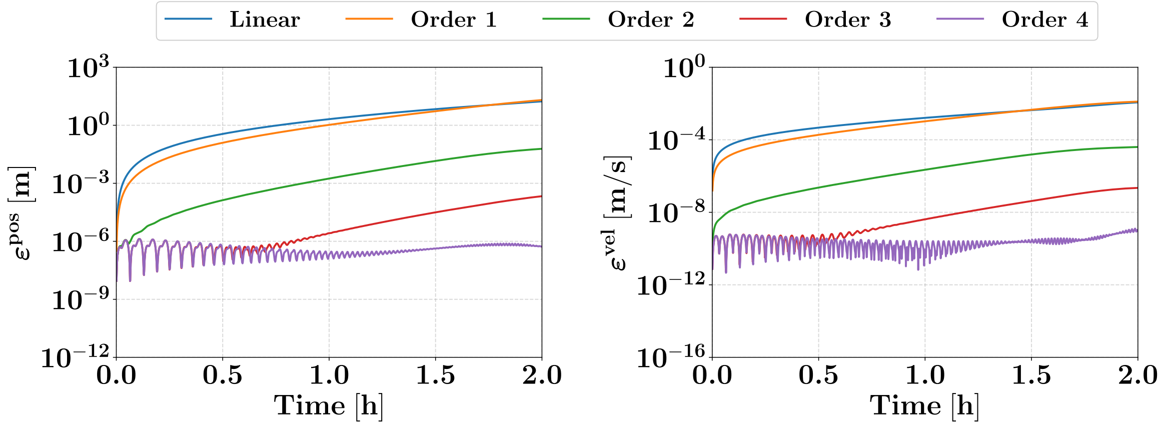
where the partial derivative appearing in the equation assumes the following generic form:

$$\frac{\partial}{\partial\boldsymbol{q}}\left[\frac{\boldsymbol{q}}{q^3}\right] = \frac{1}{q^3}\left(\boldsymbol{I} - 3\frac{\boldsymbol{q}\boldsymbol{q}^T}{q^2}\right) \tag{A.8}$$

## A.2. Accuracy of Polynomial Maps Using the RK-78 Propagator



(a) Aposelene.

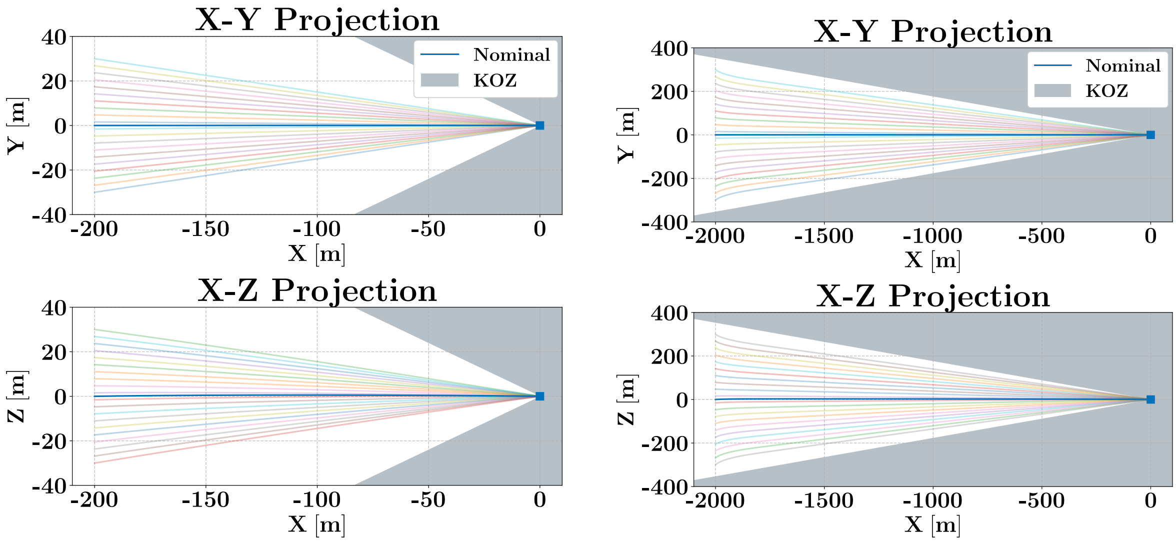


(b) Periselene.

Figure A.1: Polynomial maps propagation errors for different expansion orders, propagated using RK-78. Left: position error, right: velocity error.

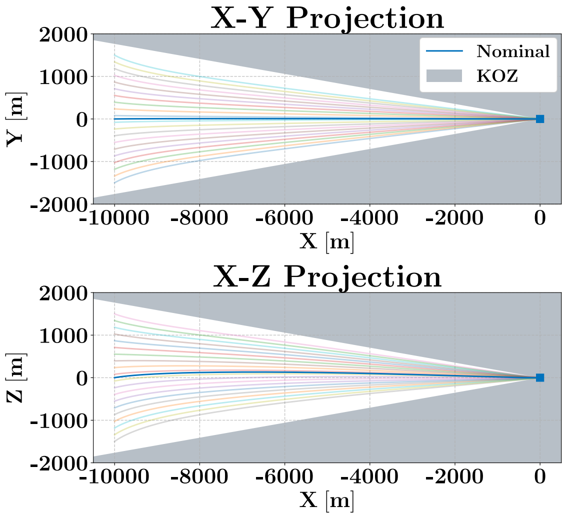
# A.3. Fixed Sampling Time Simulations - Additional Plots

## A.3.1. DAMPC



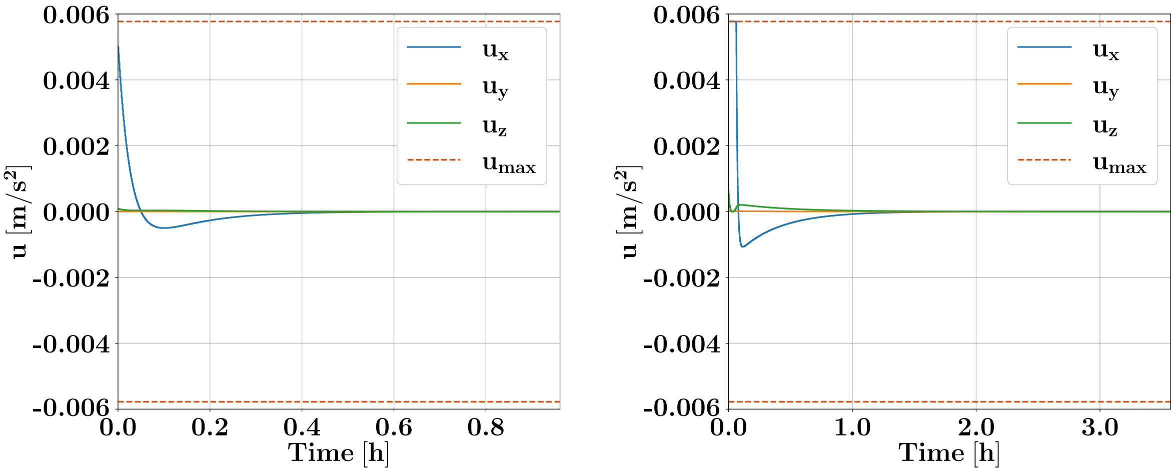
(a) Short range.

(b) Medium range.



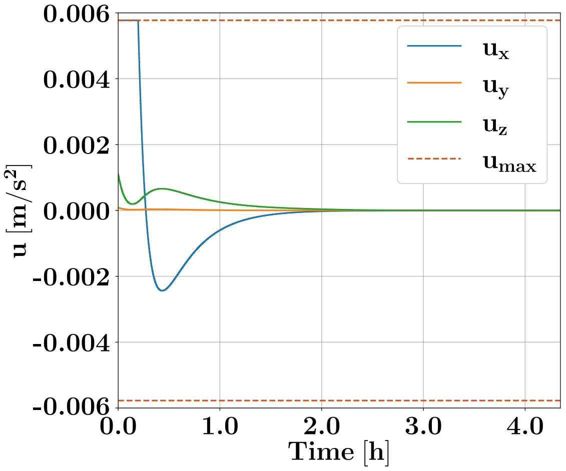
(c) Long range.

Figure A.2: Fixed $T_s$ simulations. DAMPC trajectories at the periselene in the LVLH reference frame $\mathcal{L}$. The blue line depicts the nominal case, while the fainter lines correspond to the trajectories obtained from displaced initial conditions; the target, located at the origin, is marked with a square.



(a) Short range.

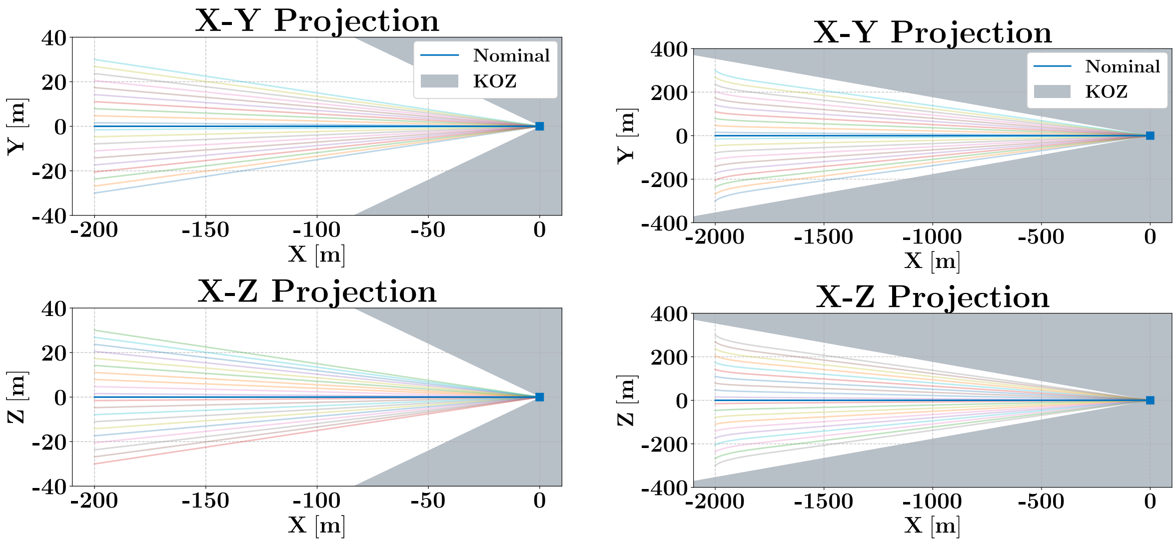
(b) Medium range.



(c) Long range.

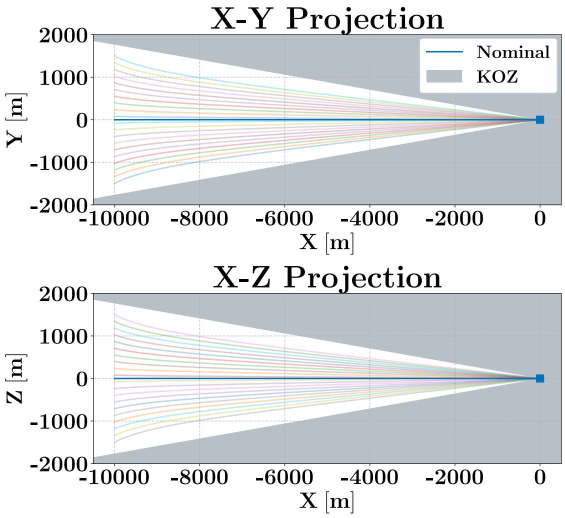
Figure A.3: Fixed $T_s$ simulations. DAMPC control inputs at the periselene, in the LVLH reference frame $\mathcal{L}$.

## A.3.2. LMPC



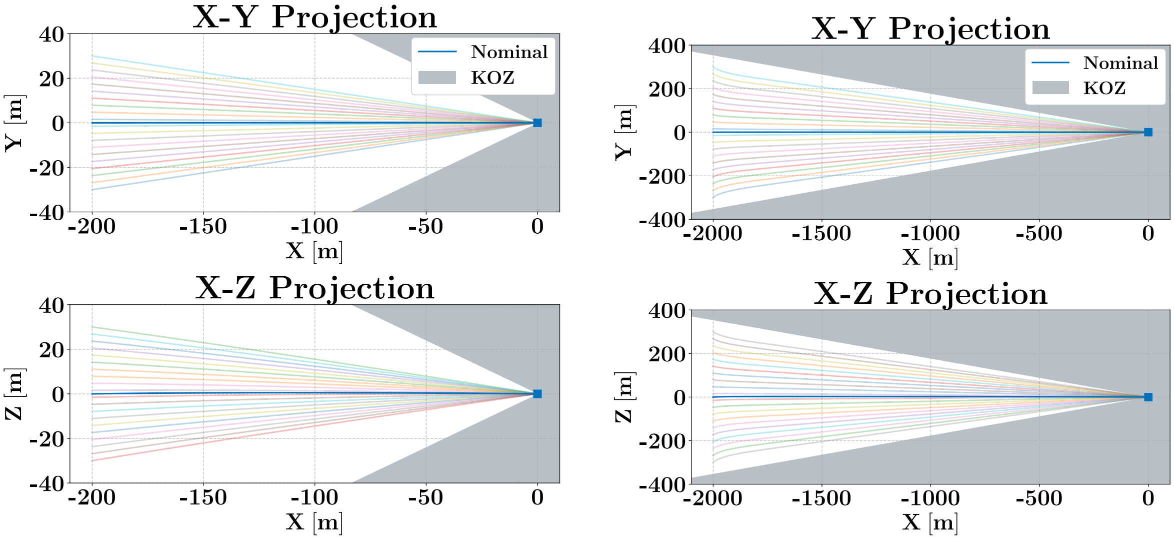
(a) Short range.

(b) Medium range.



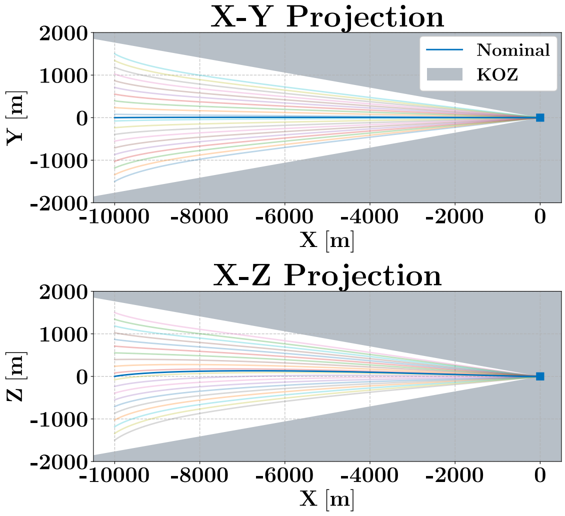
(c) Long range.

Figure A.4: Fixed $T_s$ simulations. LMPC trajectories at the aposelene in the LVLH reference frame $\mathcal{L}$. The blue line depicts the nominal case, while the fainter lines correspond to the trajectories obtained from displaced initial conditions; the target, located at the origin, is marked with a square.



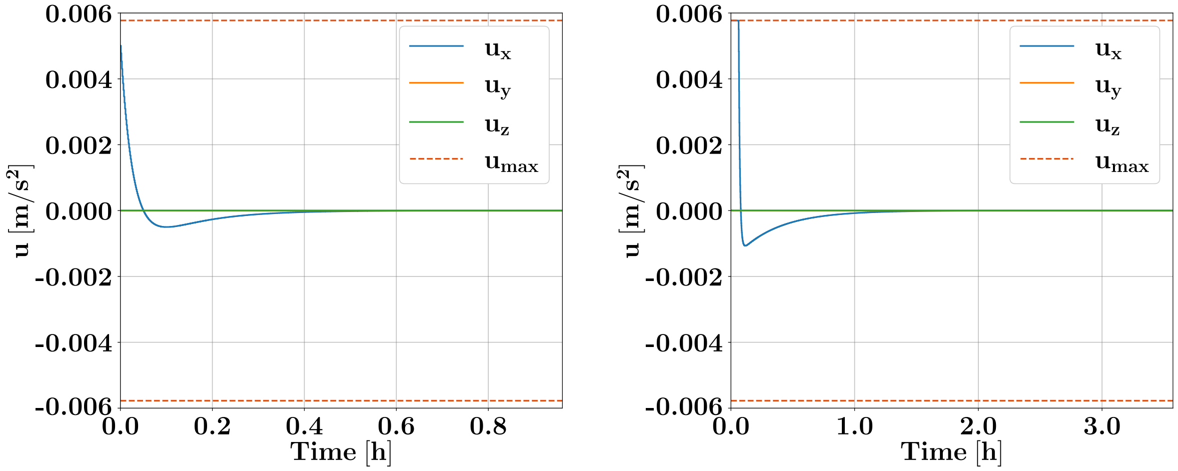
(a) Short range.

(b) Medium range.

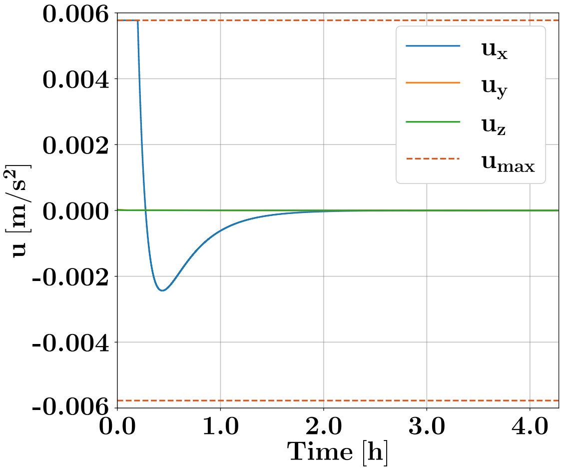


(c) Long range.

Figure A.5: Fixed $T_s$ simulations. LMPC trajectories at the periselene in the LVLH reference frame $\mathcal{L}$. The blue line depicts the nominal case, while the fainter lines correspond to the trajectories obtained from displaced initial conditions; the target, located at the origin, is marked with a square.

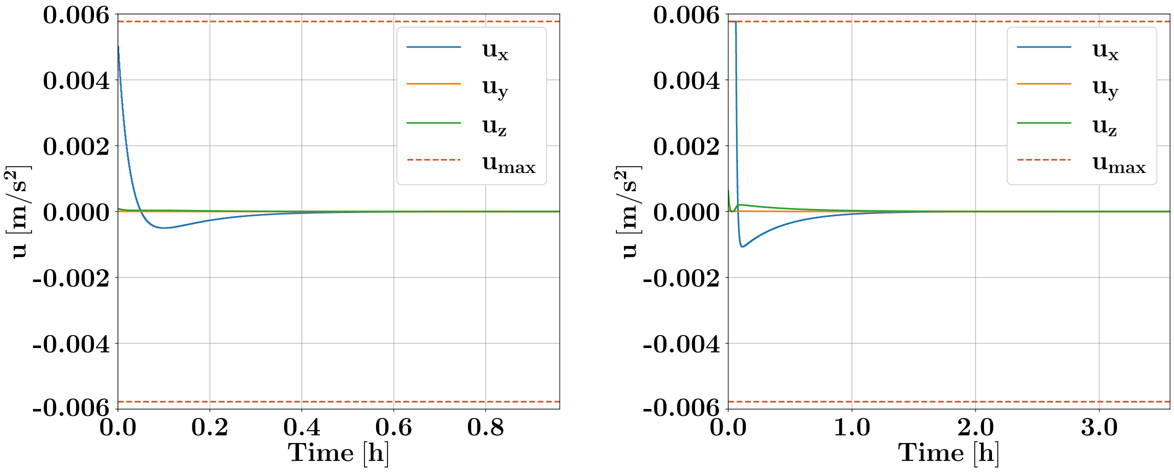


(a) Short range. (b) Medium range.

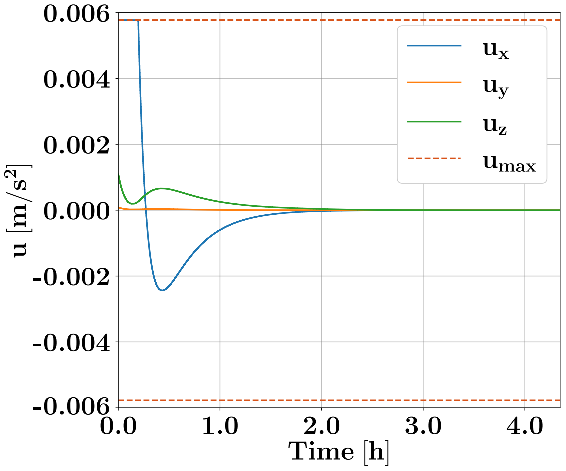


(c) Long range.

Figure A.6: Fixed $T_s$ simulations. LMPC control inputs at the aposelene, in the LVLH reference frame $\mathcal{L}$.



(a) Short range. (b) Medium range.



(c) Long range.

Figure A.7: Fixed $T_s$ simulations. LMPC control inputs at the periselene, in the LVLH reference frame $\mathcal{L}$.

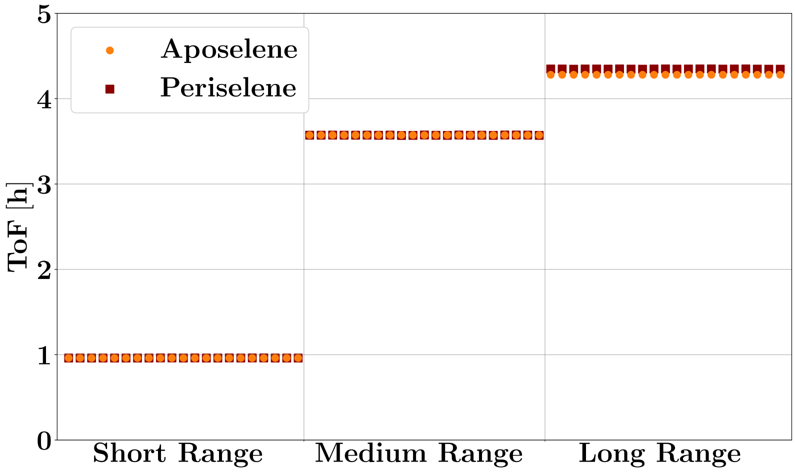
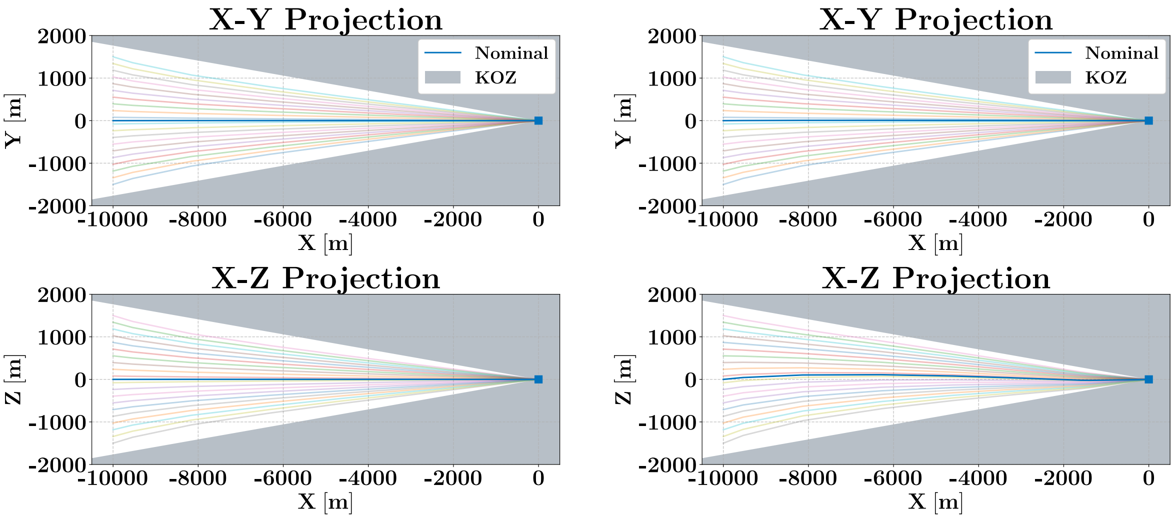


Figure A.8: Fixed $T_s$ simulations. LMPC time of flight.

| | **Short Range** | **Medium Range** | **Long Range** |
|---|---|---|---|
| Aposelene | 0.9611 h | 3.5733 h | 4.2800 h |
| Periselene | 0.9614 h | 3.5721 h | 4.3474 h |

Table A.1: Fixed $T_s$ simulations. LMPC average time of flight.

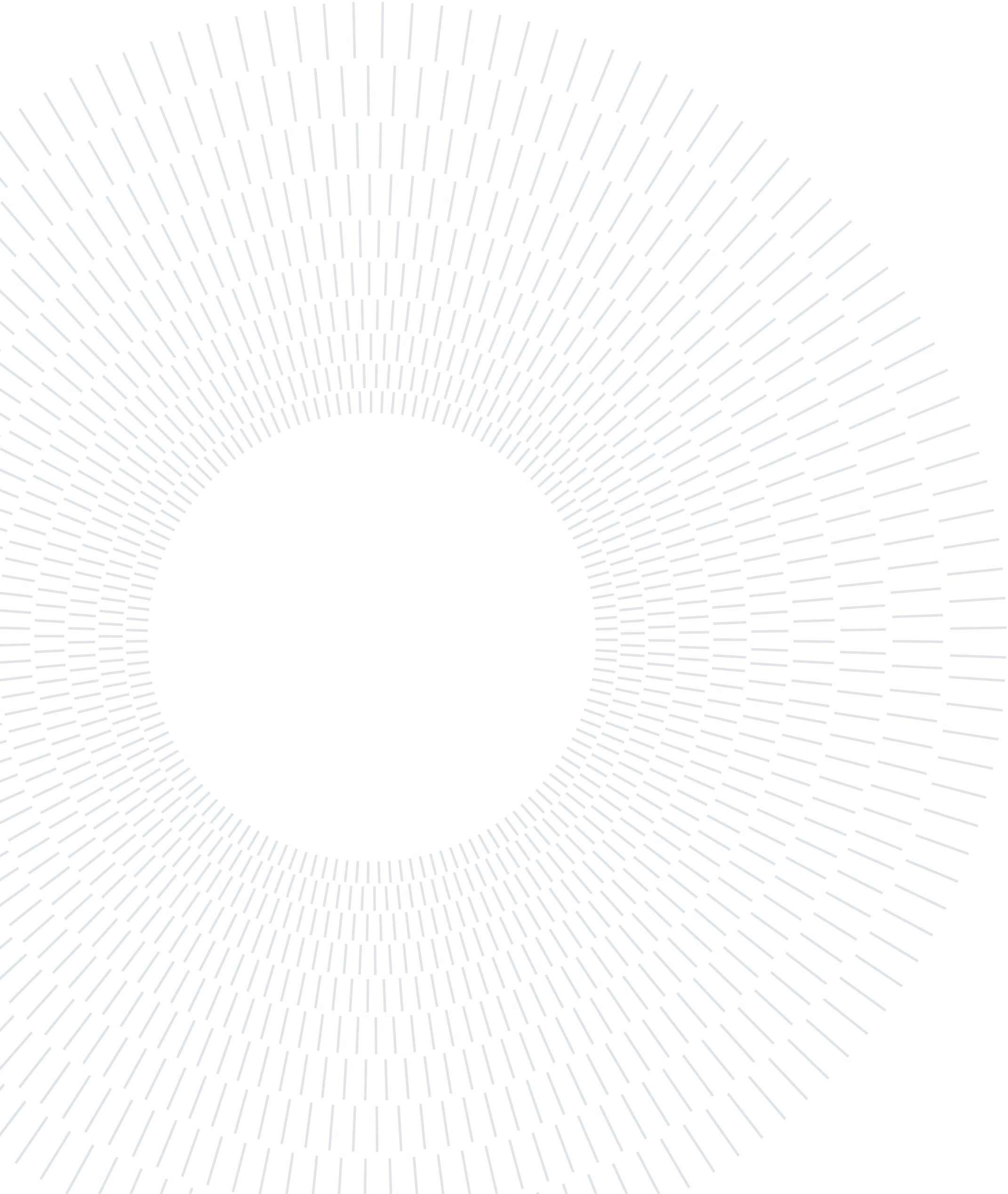
## A.4. Variable Sampling Time Simulations - Additional Plots



(a) Aposelene.

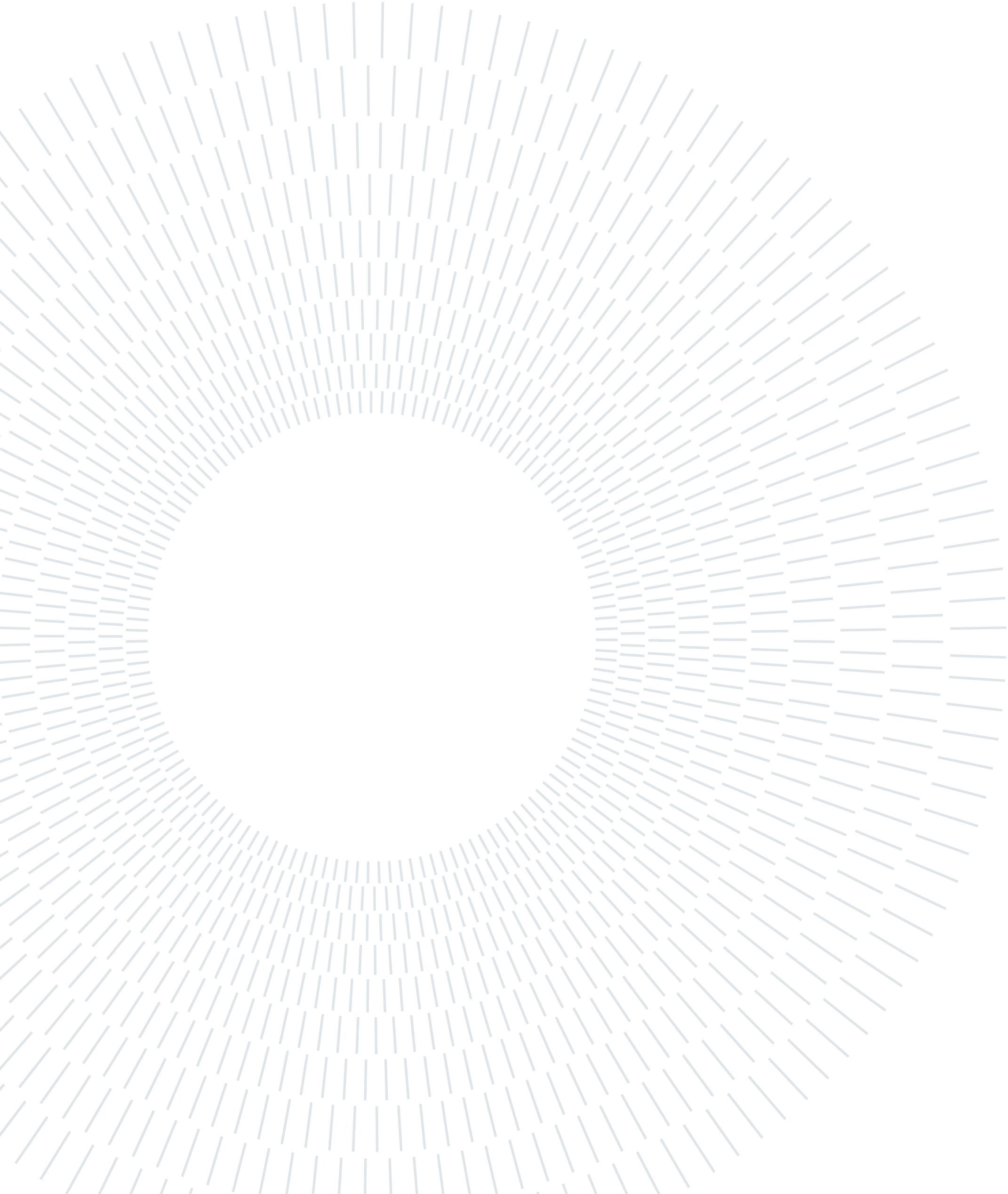
(b) Periselene.

Figure A.9: Variable $T_s$ simulations. LMPC trajectories in the LVLH reference frame $\mathcal{L}$, obtained from the simulations with variable sampling time. The blue line depicts the nominal case, while the fainter lines correspond to those obtained from displaced initial conditions; the target, located at the origin, is marked with a square.



# List of Figures

# List of Tables



# List of Symbols

| Variable | Description | SI unit |
|---|---|---|
| $\mathcal{I} : \{\boldsymbol{O}; \hat{\mathbf{I}}, \hat{\mathbf{J}}, \hat{\mathbf{K}}\}$ | Inertial reference frame | - |
| $\mathcal{M} : \{\boldsymbol{R}_m; \hat{\boldsymbol{\imath}}_m, \hat{\boldsymbol{\jmath}}_m, \hat{\boldsymbol{k}}_m\}$ | Synodic reference frame | - |
| $\mathcal{L} : \{\boldsymbol{r}; \hat{\boldsymbol{\imath}}, \hat{\boldsymbol{\jmath}}, \hat{\boldsymbol{k}}\}$ | LVLH reference frame | - |
| $\boldsymbol{R}_i$ | Inertial position vector of i | km |
| $\boldsymbol{r}_{ij}$ | Position vector of i with respect to j | km |
| $\boldsymbol{\rho}$ | Chaser position vector with respect to the target | km |
| $\boldsymbol{h}_{i/j}$ | Angular momentum of i with respect to j | km$^2$/s |
| $M_i$ | Primary body i mass | kg |
| $m_i$ | Spacecraft i mass | kg |
| $\mu_i$ | Gravitational parameter of the primary body i | km$^3$/s$^2$ |
| $\boldsymbol{\omega}_{i/j}$ | Angular velocity of i with respect to j | rad/s |
| $\boldsymbol{\Omega}_{i/j}$ | Skew symmetric matrix containing $\boldsymbol{\omega}_{i/j}$ | rad/s |
| $\boldsymbol{x}$ | State vector | km, km/s |
| $\boldsymbol{u}$ | Control input vector | km/s$^2$ |
| $\boldsymbol{x}_k$ | Discrete state vector | km, km/s |
| $\boldsymbol{u}_k$ | Discrete control input vector | km/s$^2$ |
| $\bar{\boldsymbol{x}}_i$ | Predicted state vector at time instant i within the prediction horizon | km, km/s |
| $\bar{\boldsymbol{u}}_i$ | Predicted control input vector at time instant i within the prediction horizon | km/s$^2$ |
| $\tilde{\boldsymbol{A}}$ | Continuous state transition matrix | - |
| $\tilde{\boldsymbol{B}}$ | Continuous control input matrix | - |
| $\boldsymbol{A}_k$ | Discrete state transition matrix | - |

*Continued on next page*

| **Variable** | **Description** | **SI unit** |
|---|---|---|
| $\boldsymbol{B}_k$ | Discrete control input matrix | - |
| $N$ | MPC prediction horizon length | - |
| $M$ | MPC control horizon length | - |
| $T_s$ | Sampling time | s |
| $\boldsymbol{Q}$ | MPC state weight | - |
| $Q_{pos}$ | MPC state weight applied to the position | - |
| $Q_{vel}$ | MPC state weight applied to the velocity | - |
| $\boldsymbol{R}$ | MPC control weight | - |
| $\boldsymbol{P}$ | Solution of the algebraic Riccati equation | - |
| $n$ | Polynomial expansion order | - |
| $v$ | Number of variable in DA environment | - |
| $\mathcal{M}_{\boldsymbol{x}_{k+1}}(\delta\boldsymbol{x}_k)$ | Polynomial maps for propagating the state $\boldsymbol{x}_{k+1}$ evaluated at $\delta\boldsymbol{x}_k$ | - |
| $\mathcal{M}_{\bar{\boldsymbol{x}}_{i+1}}(\delta\bar{\boldsymbol{x}}_i)$ | Polynomial maps applied within the prediction horizon | - |
| $\Delta v$ | Maneuver cost | m/s |
| $\varepsilon^{pos}$ | Position error | m |
| $\varepsilon^{vel}$ | Velocity error | m/s |
| $\gamma$ | Docking port aperture angle | rad |
| $c_x$, $c_y$ | Docking port aperture | m |
| $T_{max}$ | Maximum trust deliverable | N |
| $u_{max}$ | Maximum control action deliverable | m/s$^2$ |

# List of Acronyms

| | |
|---|---|
| **APF** | Artificial Potential Field. |
| **AS** | Approach Sphere. |
| **ATV** | Automated Transfer Vehicle. |
| **CAM** | Collision Avoidance Maneuver. |
| **CCMPC** | Chance Constrained MPC. |
| **CLERM** | Circular Linear Equations of Relative Motion. |
| **CNERM** | Circular Nonlinear Equations of Relative Motion. |
| **CRTBP** | Circular Restricted Three Body Problem. |
| **DA** | Differential Algebra. |
| **DACE** | Differential Algebra Core Engine. |
| **DAMPC** | Differential Algebra MPC. |
| **DART** | Demonstration for Autonomous Rendezvous Technology. |
| **ERTBP** | Elliptical Restricted Three Body Problem. |
| **FP** | Floating-Point. |
| **G&C** | Guidance and Control. |
| **HCS** | Hard Capture System. |
| **IDSS** | International Docking System Standard. |
| **IPOPT** | Interior Point Optimizer. |
| **IRSIS** | International Rendezvous System Interoperability Standards. |
| **ISS** | International Space Station. |
| **KOS** | Keep-out Sphere. |
| **LMPC** | Linear MPC. |
| **LQR** | Linear Quadratic Regulator. |
| **LTI** | Linear Time Invariant. |
| **LTV** | Linear Time Variant. |
| **LVLH** | Local Vertical Local Horizontal. |
| **MPC** | Model Predictive Control. |
| **NASA** | National Aeronautics and Space Administration. |

| | |
|---|---|
| **NMPC** | Nonlinear MPC. |
| **NRHO** | Near Rectilinear Halo Orbit. |
| **OCP** | Optimal Control Problem. |
| **ODE** | Ordinary Differential Equation. |
| **PN** | Proportional Navigation. |
| **QP** | Quadratic Programming. |
| **RK** | Runge-Kutta. |
| **RPO** | Rendezvous and Proximity Operation. |
| **RS** | Rendezvous Sphere. |
| **RTBP** | Restricted Three Body Problem. |
| **RVD/B** | Rendezvous and Docking/Berthing. |
| **SCS** | Soft Capture System. |
| **SDRE** | State Dependent Riccati Equation. |
| **SLS** | Space Launch System. |
| **SMC** | Sliding Mode Control. |
| **TPBVP** | Two Point Boundary Value Problem. |
| **TPSA** | Truncated Power Series Algebra. |
| **TRMPC** | Tube-based Robust MPC. |
| **ZEM/ZEV** | Zero-Effort-Miss/Zero-Effort-Velocity. |
| **ZOH** | Zero Order Hold. |