POLITECNICO
MILANO 1863

Politecnico di Milano
Department of Aerospace Science and Technology
Doctoral Programme in Aerospace Engineering

Multidisciplinary Guidance and Control Synthesis Methods for Small Satellites

Doctoral Dissertation of:
Jacopo Prinetto

Supervisor:
Prof. Michèle Lavagna

Tutor:
Prof. Giuseppe Sala

Coordinator:
Prof. Pierangelo Masarati

Year 2022 – Cycle XXXV

To my parents.

Abstract

The interest in small-sat and CubeSat missions is constantly increasing during the lasts years, as well as the complexity of the scenarios in which such class of satellites is employed and the performances that they shall guarantee in term of both attitude and orbital control. This growing interest is motivated on one side by the fact that small-sat can potentially reduce the costs and development time of a mission when compared with larger satellites, and on the other side by the fact that small-sat open the possibility to exploit innovative mission concepts with fractionated of repeated payload, increasing the scientific or economic return while minimizing the single-point-failures that can affect large monolithic satellites.

The research work presented here focuses on the development and testing of multidisciplinary guidance and control synthesis methods for small satellites. The multidisciplinary approach, that will be extensively recalled in this work, become almost essential when small-sat guidance and control problems are addressed, indeed the hardware performance of such satellites are limited by their size and the reduced heritage. In particular, the performance of these satellites are typically limited by some critical elements, namely the propulsive units, the available power and energy and the computational performances.

In this research work, two novel semi-analytical guidance algorithms for the center of mass and attitude motion are developed, presented and tested in mission scenarios of different complexity to prove their validity. In particular, the guidance algorithm for the center of mass motion is focused on the solution of the continuous control problem of a thrusting spacecraft, to be suitable for both

electrical thrusters and low thrust-to-weight ratio chemical propulsive units. The algorithm is also employed in a small-sat multi injection mission scenario, adopting a novel routing problem approach. The attitude guidance algorithm is focused on the optimization of highly constrained attitude maneuvers. The constrains implemented arise both from hardware limitations and from the imposition of forbidden/desired pointing directions thought the exploitation of keep-in and keep-out cones. These last constraints are representative of scenarios in which some pointing directions are desired, such as the antenna to ground visibility during the maneuver, or the maximization of solar panels exposition to the sun, or in which some pointing directions are forbidden, such as payload and/or navigation instrument Sun/Earth/Moon avoidance. The algorithm is then coupled with a controller suitable for small-sat on-board implementation and tested in some realistic scenarios. As last step, the effect of flexible behavior of small-sat on the pointing performances is investigated through the adoption of a proper multi-body formulation of the equation of motion of the flexible small-sat.

Acknowledgements

Table of Contents

Abstract **V**

Acknowledgements **IX**

List of Figures **XV**

List of Tables **XVII**

Acronyms **XXI**

1 Introduction **1**
1.1 The Research Problem 2
1.2 Dissertation Overview 3
1.3 Bibliographic Disclaimer 4

2 Background & State of the Art **5**
2.1 Absolute Orbital Dynamics 5
2.1.1 The gravitational model and the environment 5
2.1.2 The Low Thrust Problem 6
2.1.3 Vehicle Routing Problem 8
2.2 Attitude dynamics 9
2.2.1 The attitude motion and the environment 9
2.2.2 Attitude guidance and attitude motion planning 9
2.3 Flexible Dynamics 10
2.4 Heuristic Optimization 13

3 Orbital Guidance **17**
3.1 Guidance algorithm . . . 17
3.1.1 Single Revolution Algorithm . . . 18
3.1.1.1 Requirements . . . 18
3.1.1.2 Low thrust equations parametrization . . . 19
3.1.1.3 Non-Linear Interpolation . . . 21
3.1.1.4 the algorithm architecture . . . 23
3.1.1.5 The interpolating functions . . . 25
3.1.2 Multi-revolution Approaches . . . 26
3.1.2.1 Interplanetary scenario . . . 26
3.1.2.2 Planeto-centric scenario . . . 27
3.1.3 Test Cases . . . 30
3.1.3.1 Very low thrust LEO-GEO transfer . . . 31
3.1.3.2 Orbital Raising to GEO . . . 31
3.1.3.3 Earth-Mars Rendezvous . . . 35
3.1.3.4 Earth-Nereus Rendezvous . . . 36
3.2 Low thrust multi-injection approach for constellation and multi-mission deployment . . . 38
3.2.1 Multi-deployment algorithm . . . 38
3.2.1.1 Routing . . . 38
3.2.1.2 Low-thrust transfer . . . 42
3.2.2 Optimization . . . 46
3.2.2.1 Optimization approach . . . 46
3.2.2.2 Bounding criteria . . . 50
3.2.2.3 Constraints . . . 51
3.2.3 Results and Discussion . . . 51
3.2.3.1 Heuristic-hybrid approaches comparison . . . 51
3.2.3.2 Constellation insertion . . . 54

4 Attitude Guidance **59**
4.1 Analytical derivation of the equation of motion . . . 59
4.2 Quaternion shaping . . . 63
4.3 Test Cases . . . 67
4.3.1 Case 1: Unconstrained slew maneuver with final and initial non-null angular velocity . . . 67
4.3.2 Time-Optimal Spacecraft Reorientation with Keep-Out Cones 73

5 Attitude Control Techniques **79**
5.1 6 DoF simulator . . . 79
5.2 Control Techniques . . . 80
5.2.1 Feed-Forward open loop control . . . 85
5.2.2 Classical PID controller . . . 86
5.2.3 Feed-forward PID controller with Gain-Scheduling . . . 90

5.2.4 Control Techniques Comparison 94
5.2.5 Effect of model uncertainties on the control techniques . . 95

6 Flexible Dynamics **99**
6.1 multi-body modeling 99
6.1.1 Central body 101
6.1.2 Booms 102
6.1.3 solar panels 105
6.1.4 EoM 108
6.2 Test Cases 110
6.2.1 Rest-to-rest fast slew 110

7 Conclusions **115**
7.1 Major Results 115
7.2 Future Works 118

A List of derivatives **123**
A.1 Departure orbit 123
A.2 Target orbit 125
A.3 Attractor distances 126

Bibliography **127**

List of Figures

2.1 Example of a Lumped Mass Model (LMM) modeling of a flexible appendages undergoing large planar motion 11
2.2 Example of a Lumped Parameters Model (LPM) modeling of a flexible appendages undergoing large planar motion 12
2.3 Example of a Distributed Parameter Model (DPM) modeling of a flexible appendages undergoing large planar motion 12

3.1 Spacecraft position 19
3.2 Non-linear interpolation 22
3.3 Architecture 23
3.4 Interp. function comparison 26
3.5 Example of multi-revolution application 31
3.6 Multi-revolution algorithm architecture 32
3.7 Orbit rising to Geostationary Equatorial Orbit (GEO) 33
3.8 Time optimal trajectory 35
3.9 Launch opportunities 35
3.10 Earth-Nereus trajectory 37
3.11 Routing algorithm main workflow. 39
3.12 Single low-thrust J2-exploiting transfer scheme. 44
3.13 Branch and bound based heuristic approach scheme. 48
3.14 Pareto front and partially dominated solutions. 50
3.15 Comparison of optimization approaches results 53
3.16 Pareto front for Starlink replacement mission 56
3.17 Pareto fronts with different number of thrusters for Starlink replacement mission 57

4.1 Quaternions shaping 64
4.2 Shaping function behavior with e_i ranging from -10 to $+10$. . . . 66
4.3 PSO convergence for case 1. 70
4.4 Optimal slew maneuvers considering different objectives compared with the first guess solution ($e_i = 0$ and $T_{man} = 55s$) 72
4.5 keep-out cones definition. 74
4.6 PSO convergence for case 1. 75
4.7 Optimal maneuver. Case 2 76
4.8 Optimal control torque (*left*) and angular velocity (*right*). Case 2 77
4.9 Time evolution of the quaternions and their derivatives for the optimal slew of case 2. 78

5.1 HERMES 3U CubeSat configuration 81
5.2 3-D representation of the minimum-time optimal solution 83
5.3 RWs control and status behavior of the minimum-time optimal solution 84
5.4 Quaternions and angular velocity behavior of the minimum-time optimal solution 84
5.5 Open Loop control scheme 85
5.6 Angular error during Feed-Forward Open Loop controlled slew considering different control frequencies 86
5.7 PID control scheme 87
5.8 Trajectory of the PD controlled slew - control frequency $= 10\,\mathrm{Hz}$. 88
5.9 Angular error during PD controlled slew considering different control frequencies 89
5.10 Comparison of nominal, ideal and real control actions during the slew maneuver 90
5.11 Feed-forward PID control scheme 91
5.12 Trajectory of the Feed-Forward Proportional-Derivative (FFPD) controlled slew - control frequency $= 10\,\mathrm{Hz}$ 92
5.13 Angular error during Feed-Forward PD controlled slew considering different control frequencies 93
5.14 Comparison of nominal, ideal and real control actions during the slew maneuver - Feed Forward PD 93
5.15 Identification of the phases during the time response: Nominal maneuver (red), Transient Phase (purple), Steady-State (green) . 94
5.16 Pointing error evaluation via Montecarlo analysis for the Proportional Derivative (PD) controller (*upper picture*) and the FFPD controller (*lower picture*) 96

6.1 Typical CubeSat form factors - Credits of Cal Poly – San Luis Obispo, CA 100
6.2 Antenna booms modeling 102
6.3 Selected Ritz-Galerking (RG) trial function. 106

6.4 Solar panels modeling 106
6.5 Comparison of rigid and flexible configuration during the slew ($t = 3s$). Elastic displacement enlarged for sake of visibility 111
6.6 Solar panel angular displacements (*left*) and booms tip displacements (*right*). 112
6.7 Angular velocity(*upper*), angular acceleration (*middle*) and flexible disturbance (*bottom*) 112
6.8 Pointing error - FFPD controller 113
6.9 Frequency analysis of the flexible disturbance 113
6.10 Comparison of nominal, ideal and real control actions during the slew maneuver - Feed Forward PD with flexible model 114

List of Tables

3.1 Simulation parameters 31
3.2 Comparison of results 33
3.3 Time-optimal constrained solutions 34
3.4 Search domain 36
3.5 Methods for routing algorithm validation. 41
3.6 Results comparison for algorithm validation. 42
3.7 Number of Lambert routine calls comparison. 42
3.8 Keplerian Parameters (KP) of the drifting orbit for low-thrust J2-exploiting transfer strategy. 44
3.9 Initial states for multi-deployment mission example. 52
3.10 Optimization tuning parameters for multi-deployment mission example. 52
3.11 Spacecraft characteristics for multi-deployment mission example. . 52
3.12 Optimization tuning parameters of hybrid approach for optimization approaches comparison. 53
3.13 Initial states for Starlink replacement mission. 54
3.14 Optimization tuning parameters for Starlink replacement mission. . 55
3.15 Lower and upper bounds for Starlink replacement mission. 55
3.16 Release details of the fastest solution for Starlink replacement mission. 56

4.1 Result Comparison. Case 1. 70
4.2 keep-out cones definition 73
4.3 literature comparison 75

5.1 Spacecraft orbital initial state at $t = 0$ 81

5.2 Determination and Control system specifications 82
5.3 Optimization algorithms comparison - fast slew maneuver 83
5.4 Feed-Forward PD optimized gains 91
5.5 Control technique comparison: Mean Quadratic Errors and transient phases duration 95
5.6 Mean quadratic pointing error variation adopting an uncertainty in the inertia model 96

6.1 Solar arrays flexible model parameters 110
6.2 Booms flexible model parameters 111

Acronyms

ACS	Attitude Control System
ADCS	Attitude Determination and Control System
ADR	Active Debris Removal
AKE	Attitude Knowledge Error
BF	Body-Fixed
CGM	Control Moment Gyroscope
CoM	Center of Mass
COTS	Commercial-Off-The-Shelf
CPU	Central Processing Unit
CR3BP	Circular Restricted Three-Body Problem
DCM	Direction Cosine Matrix
DoF	Degrees of Freedom
DPM	Distributed Parameter Model
EA	Evolutionary Algorithms
EoM	Equation of Motion
FEM	Finite Element Method
FFPD	Feed-Forward Proportional-Derivative

FF	Feed-Forward
GA	Genetic Algorithm
GEO	Geostationary Equatorial Orbit
GTOC	Global Trajectories Optimization Competition
IW	Inertia Wheel
I-P	Interior Points
KP	Keplerian Parameters
LEO	Low Earth Orbit
LTAN	Local Time of the Ascending Node
LMM	Lumped Mass Model
LOS	Line Of Sight
LPM	Lumped Parameters Model
LQR	Linear Quadratic Regulator
MEE	Modified Equinoctial Elements
MGA	Multiple Gravity Assists
MINLP	Mixed-Integer Nonlinear Programming
MOPSO	Multi-Objective Particle Swarm Optimization
MPC	Model Predictive Control
NEO	Near Earth Object
N-M	Nelder-Mead
OBDH	On-board Data Handling
ODE	Ordinary Differenzial Equation
PD	Proportional Derivative
PDE	Partial Differential Equations
PID	Proportional Integral Derivative
PIL	Processor In the Loop
PSO	Particle Swarm Optimization
PVW	Principle of Virtual Work
RAAN	Right Ascension of the Ascending Node

RF	Reference Frame
RG	Ritz-Galerking
RWs	Reaction Wheels
SI	Swarm Intelligence
SSO	Sun-Synchronous Orbit
TBP	Two Body Problem
TOF	Time Of Flight
TPA	Two Phase Algorithm
TSP	Travelling Salesman Problem
VRP	Vehicle Routing Problem

CHAPTER 1

Introduction

The interest in small-sat and CubeSat[1] missions is constantly increasing during the lasts years. They evolved from being mere teaching tools to true spacecraft capable of carrying out important scientific and technological demonstration missions[2, 3]. The continuous improvements in the hardware miniaturization leads to small satellites with potential performances that some years ago were only prerogatives of larger platforms, dramatically reducing the costs and the risks related to space exploration. Moreover small-spacecrafts open the possibility to exploit a fractionated or repeated payload architecture, mitigating the consequences of a failure of a single element in the constellation and increasing the scientific and/or commercial return of the whole mission. The use of Commercial-Off-The-Shelf (COTS) components and a standardized approach to the system design pose advantages in terms of development time and costs if compared with respect to larger satellites. Moreover, the possibility of launching several tens of small-sat with a single launch, makes it possible radically new mission scenarios and architectures consisting of very large constellations or clusters of small-sat for both scientific and commercial missions. At the same time the low-thrust based technologies, such as ion and Hall

effect thrusters, have become sufficiently mature to be installed on that kind of platforms for both orbital maneuvers and fine pointing purposes. These technologies are able to reduce the amount of fuel required, to extend the lifetime of the platform and to increase the pointing performances and the mission flexibility. Unfortunately, from one side the commercial small-sat and CubeSat hardware, especially for the Attitude Determination and Control System (ADCS) and On-board Data Handling (OBDH) subsystem, is not mature enough to follow the rapid increase in performances required, from the other side the low-thrust based technologies strongly affect the design of the whole platforms on which they are installed, requiring an elevate amount of power and a continuous thrusters fine pointing and making difficult to separate the mission analysis from the design of the platform itself: a multi-disciplinary approach is required in order to quickly converge to an optimal design [4]. For these reasons it is necessary to further investigate guidance and control techniques for both the orbital and attitude dynamics. In particular, differently from most of the algorithms available in literature, they should keep into account practical technical and mission constraint.

1.1 The Research Problem

The objective of the research is to investigate fast, robust and effective semi-analytical algorithms to support multidisciplinary-based guidance synthesis and on-board control. More in detail, four main goals have been identified as crucial. They are summarized as follows:

1. To derive a fast semi-analytical guidance algorithm for the Center of Mass (CoM) motion that is suitable to deal with the peculiarities of the small-sat and their mission scenarios as indicated in the introduction, that is able to overcome some of the typical limitation of the in-literature available algorithms. The method should be able to quickly identify near-optimal trajectories even in presence of very complex mission scenarios.

2. To derive a semi-analytical guidance algorithm for the attitude motion of the spacecraft, that is able to manage practical and mission related constraints, such as actuation limitation and attitude keep-out cones. A multi-disciplinary approach should be carried on to tailor the algorithm on small-sat and CubeSat needs and capabilities.

3. To identify the most suitable control schemes and algorithms that can fly on small-sat hardwares with very limited computational performances but able to follow the developed guidance in an high fidelity simulated environment. Within this framework, the developed guidance algorithms could be also tested.

4. To explore new multi-body modeling techniques tailored for small-sat and CubeSat, with the aim of investigate the flexible effect on the attitude performances.

1.2 Dissertation Overview

This research work focuses on analyses, methods, techniques and tools for small-sat guidance and control, with the purpose of enhance the attitude and orbital performances of such class of satellites. To do that, novel approaches and algorithms are proposed, in support of actual methodologies. The discussion of the developed techniques is accompanied by the presentation of some practical example, with the double aim of further clarification and comparison with the literature.

Chapter 2 deals with the background knowledge. In particular, the attention is focused on the state of the art for the attitude and orbital semi-analytical guidance algorithm, as well as the methodologies adopted for the multi-body modeling of small-sat, and in particular CubeSats.

Chapter 3 presents the advancements achieved on the semi-analytical CoM guidance. In particular, Sec. 3.1 is devoted to the analysis of a novel shape-based algorithm working on a non-linear interpolation between consecutive orbits developed by the author. In Sec. 3.2.1 the developed algorithm is generalized in more complex scenarios, such as orbital transfers with the help of J_2 drift and multi-deployment missions, with the introduction of a novel simple but effective approach to solve large vehicle routing problem through a branch and bound approach based on the exploitation of partial permutations.

Chapter 4 proposes a semi-analytical algorithm to design attitude maneuvers in presence of strong attitude and platforms constraints. In particular the algorithm, thanks to its inherent speed of execution, is able to locate near-optimal attitude trajectories even in presence of complex search-spaces. The algorithm follows a multi-disciplinary approach, keeping into account real hardware models and practical objectives and constraints related to the platform limitations and necessities .

Chapter 5 present a comparison between low computational cost control algorithms, that can be easily implemented on actual CubeSat hardware, with the double aim of test the guidance developed in Chapter 4 and to investigate the possibility to reach good performances in steady state error rejection and reference tracking with simple controllers.

Chapter 6 presents a novel multi-body model for the spacecraft motion specifically tailored for CubeSat. More in details, the developed model takes into

account flexible effects coming from booms-like structures, such as UHF/VHF antennas, and large bi-dimensional structures, such as multi-panel solar arrays.

Final comments and a possible road-map for future research and devel opment, are stated in Chapter 7

1.3 Bibliographic Disclaimer

During the years of my Philosophiae Doctor (Ph.D.), I presented updates of my work in many conferences and I also had the possibility to publish part of them in peer reviewed journals. Therefore, some of the work in this dissertation has already been presented in different articles. The most significant are listed below.

- Prinetto, J., & Lavagna, M. (2021). Elliptical shape-based model for multi-revolution planeto-centric mission scenarios. Celestial Mechanics and Dynamical Astronomy, 133(1), 1-24.
- Salvato, V. M., Prinetto, J., & Lavagna, M. (2021). Low Thrust Multi-Injection Approach for Constellation and Multi-Mission Deployment. In SpaceOps 2021 Virtual Edition-16th International Conference on Space Operations (pp. 1-18).
- Prinetto, J., Lunghi, P., & Lavagna, M. (2021). Fast and Accurate Re-Planning Tool Under Multidisciplinary Constraints Set. In SpaceOps 2021 Virtual Edition-16th International Conference on Space Operations (pp. 1-10).
- Silvestrini, S., Prinetto, J., Zanotti, G., & Lavagna, M. (2020). Design of robust passively safe relative trajectories for uncooperative debris imaging in preparation to removal. In 2020 AAS/AIAA Astrodynamics Specialist Conference (Vol. 175, pp. 4205-4222). Univelt.
- Colagrossi, A., Prinetto, J., Silvestrini, S., Orfano, M., Lavagna, M., Fiore, F., ... & Pirrotta, S. (2019). Semi-analytical approach to fasten complex and flexible pointing strategies definition for nanosatellite clusters: The HERMES mission case from design to flight. In 70th International Astronautical Congress (IAC 2019) (pp. 1-8).
- Colagrossi, A., Prinetto, J., Silvestrini, S., & Lavagna, M. R. (2020). Sky visibility analysis for astrophysical data return maximization in HERMES constellation. Journal of Astronomical Telescopes, Instruments, and Systems, 6(4), 048001.

CHAPTER 2

Background & State of Art

This section shows the background and the current level of development of the main pillars of the dissertation.

2.1 Absolute Orbital Dynamics

In this section the Absolute Orbital Dynamics is introduced, with a focus on the state of the art of the semi-analytical guidance algorithms.

2.1.1 The gravitational model and the environment

The framework within the Absolute Orbital guidance in this work is developed is the perturbed Two Body Problem (TBP). In particular, the motion of a controlled spacecraft around a primary object, defined as Attractor, is governed by the system of seven coupled non-linear equations presented in Eq. 2.1 [5]

$$\begin{cases} \dot{\mathbf{r}} = \mathbf{v} \\ \dot{\mathbf{v}} = -\frac{\mu}{r^3}\mathbf{r} + \frac{\mathbf{f}_{pert}}{m} + \frac{\mathbf{f}_{control}}{m} \\ \dot{m} = -\frac{f_{control}}{I_S g_0} \end{cases} \tag{2.1}$$

in which the first six equation describe the physical state of the spacecraft during the motion, wile the last describe the variation of the mass in time due to the presence of thrusters. The natural motion of the spacecraft if forced not only by the presence of actuators but also by natural perturbations. The perturbations, that are time and state dependent, arise from different natural phenomena [6], namely the *third body effects*, the *Atmospheric forces* (considered as drag only, in most of the cases), the *Solar radiation Pressure* and the *Non-uniformity in the gravity field of the attractor*. Even if from a purely theoretical point of view the selection of a specific Reference Frame (RF) doesn't affect the solution of the motion, if numerical methods are employed the choice of the correct RF becomes of primary importance. In most of the cases the Equation of Motion (EoM) are written in one of the following four set of coordinates: the *Cartesian* coordinates, the *polar* coordinates, the KP and the Modified Equinoctial Elements (MEE). The Cartesian coordinates are the simplest to be used, but unfortunately they are not optimal to describe the motion of a thrusting spacecraft, indeed a fast and periodic variation of the states and the control variables in such coordinate requires an high number of computational nodes to have an accurate trajectory. *Polar* coordinates partially solve this problem [5], since the variation of the variables is smoother, but un fortunately they suffer of a singularity for the out of plane motion [7] that can be avoided only by selecting a proper reference plane, variable for each scenario, as will be seen in Chapter 3. Also the KP, even if they are good in therm of smoothness variation of the state and control variables, shows important singularities for circular and equatorial orbits, that cannot be avoided, and therefore typically they are not used in numerical implementations. The MEE, introduced in 1972 by Cefola [8] are able to avoid any kind of singularity, while maintaining an extremely smooth variation of the state and control variables with the double drawback of the lack of physical meaning of those parameters and the complicated formulation of the equations. Within this work, *polar* coordinates and MEE are employed as needed, and in particular, the first are used to parametrize the EoM in Chapter 3, and the second are used both in Chapter 3 for the non-linear orbital interpolation and in Chapter 5 for the coupled attitude/orbital high fidelity simulator.

2.1.2 The Low Thrust Problem

In the last decades, Solar Electric Propulsion (SEP) has became of primary interest both for long and complex interplanetary missions (e.g.ESA Bepi-Colombo, NASA Dawn) and for Earth-centered satellite station keeping (e.g. ESA-Artemis, AlphaSat). The continuous progress in SEP related technologies and the flexibility they would offer the mission can extend the applicable domain of these thrusters to the main propulsion system of planeto-centric missions as well, such as tug vehicles and launchers' upper stages. Still, the

low thrust trajectory design and optimization represent a challenge. The low thrust trajectory optimisation entails solving quite a complex continuous optimal control problem [5], as the dynamics are formalised by a set of seven coupled non-linear differential equations: in turn, they are built up by the evolution of six state-vector components plus mass rate consumption. The modeling problem grows further in complexity whenever perturbations must be considered and realistic events are to be taken into account (e.g.vehicle separation, gravity assists, eclipses). Perturbations are time and state dependent, while realistic events are formalised as discontinuities. Analytical solutions exist but for simplified, and therefore unrealistic, cases [9] [10]. It is worth noting also that the structure of the solution is typically not a-priori known [4] [5], which makes the powerful direct and indirect optimization methods inapplicable. Moreover, they typically require a good first guess to properly converge as well as having a high computational burden that makes them unsuitable for large search spaces [5][11][12]. Shape-based algorithms aim at coping with the aforementioned limitations. They approach the core strategy and revert the continuous optimal control problem by imposing kinematic constraints to the dynamics: the trajectory shape is selected, with proper degrees of freedom exploited to tune the dynamics compliance. Typically some assumptions, such as the tangential direction of the thrust, are assumed to obtain faster analytical solutions[13][7]. These methods, working on a subspace of the problem, are only capable of giving a sub-optimal solution and are extremely fast if compared with others [14][4][13]. They are well suited for fast detection of sub-optimal solutions in wide search domains using heuristic algorithms, with the possibility to exploit multiple/multidisciplinary objectives [4]. These solutions can either be adopted as initial guesses for direct or indirect optimizations or to obtain independent results during the preliminary phases of a space mission design [4][14]. The first developed shape-based techniques could only solve simple planar problems without the possibility of imposing neither exact boundary conditions on positions and velocities or the time of flight [14]. These algorithms were only used to obtain a quick estimation of the low-thrust trajectory cost and to generate initial trajectories. Improvement in the flexibility of the trajectory and in the precision of the solution was proposed in [15]; the possibility to exactly impose the boundary conditions also provides the opportunity to include gravity assist maneuvers[16]. Conway and Wall developed a simple but effective shape for both 2D problems and approximated 3D problems with small displacements from the plane[7][13]. Recently, Xie et al. [17] and Zeng et al. [18] proposed full 3D shapes for interplanetary trajectory design. Novak and Vasile presented a new coupling between the analytic solution and a LQR controller [19]. Some authors apply Fourier series to tune more effective shapes in finding solutions closer to the optimum [20][21][22]. Gondelach and Noomen [23] developed a shape-based algorithm based on velocity shaping instead of the trajectory that showed good results

in interplanetary low thrust trajectory definition. All the above-mentioned shapes, and some other variations proposed by other authors, give very good results in interplanetary trajectory design: the flexibility of the approaches and the reduced computational burden allow designing and optimizing complex mission scenarios, with mixed integer and continuous variables, therefore settling multidisciplinary objectives [4]. Algorithms developed by Taheri and Abdelkhalik [21][24] can efficiently solve Earth-Centered mission scenarios, including rendezvous and phasing, for spacecraft with thrust acceleration in the order of $10^{-1} m/s^2$. Indeed, in this peculiar environment the proximity of the attractor makes the dynamics much more constrained: hundreds or even thousands of revolutions are typically needed to move the satellite between two different orbits, and during the single revolution the osculating elements remain almost unchanged. Purely geometrical shapes cannot fit this behavior, especially whenever eccentric orbits are considered [7][13][14]. Moreover, eclipses introduce a high number of discontinuities, definitely unmanageable with the algorithms mentioned so far.

2.1.3 Vehicle Routing Problem

Finding the optimal hopping path between different orbits means solving a variant of the Travelling Salesman Problem (TSP), which is a particular formulation of the Vehicle Routing Problem (VRP). The latter belongs to the class of the NP-hard problems, meaning that the computational time required to solve them dramatically increases with the size of the problem. Most of the VRP applications to space have been about on-orbit servicing [25], Active Debris Removal (ADR) [26], [27] or Multiple Gravity Assists (MGA) problem [28]. This variant of the TSP can be formulated as a problem that includes two different kinds of variables: continuous-valued variables, such as the ones describing the state of the spacecraft in time, and binary variables, such as the ones defining the visiting order of the hopping trajectory. In particular, this problem belongs to the Mixed-Integer Nonlinear Programming (MINLP), the area of optimization which deals with non-linear problems with continuous and integer variables. Alternatively, to solve the problem a two-layer optimization scheme can be adopted, as suggested by Conway and Wall [29]. These different options were investigated in the work by Zhang et al. [30], who showed that the two-level optimization presented the worst performance. In light of such results, this work focuses on finding a solution to the MINLP. Differently from previous works, a new way to approach the VRP will is proposed, that will be specifically applied to multi-deployment mission scenarios. Moreover, the transfers will not be modeled as solutions to the Lambert problem, like in most of the previous solutions of the VRP in space, but they will be continuous-thrust transfers computed through the adoption of a shape-based approach [31].

2.2 Attitude dynamics

The section briefly present the background and state of the art for the fast semi-analytical guidance algorithm for the attitude motion.

2.2.1 The attitude motion and the environment

The attitude motion of a rigid body in the 3D space is described by the Euler equation, as in Eq. 2.2

$$\boldsymbol{I}_{SC}\dot{\boldsymbol{\omega}} + \omega \times \boldsymbol{I}_{SC}\boldsymbol{\omega} = \boldsymbol{M}_{pert} + \boldsymbol{M}_{control} \tag{2.2}$$

in which the two forcing terms are the control torque $\boldsymbol{M}_{control}$, that can be applied using different types of actuators and the perturbation torque $\boldsymbol{M}_{pert}$, that is dependent of the spacecraft state and properties, as well as the epoch. In particular, the perturbation that act on the angular motion of the spacecraft are provoked by different phenomena [41], namely the *gravity gradient*, the residual *atmosphere*, the *magnetic field*, the *solar pressure* and any internal source of disturbances, such as flexible motion or sloshing. An important element in the attitude motion is the kinematic that is selected to describe the trajectory [41]. The first possibility is to use the Direction Cosine Matrix (DCM), that uniquely define the attitude of the spacecraft, and that is free from any kind of singularity. The drawback is that the DCM is composed by nine element, among which only 3 of them are independent, therefore the representation is not optimal from a computational point of view. The second possibility is to represent attitude using Euler angles, that unfortunately are only locally defined, limiting therefore the possible with of the maneuvering. The third possibility, that avoids singularity and that is globally defined are the quaternions. Even if quaternions are the most widely used methods to represent the kinematic of the attitude motion, they are not unique, and therefore each physical attitude state can be defined by different quaternions, leading to possible ambiguity in the definition of the attitude trajectory. The Rodrigues parameters could be used to represent the attitude kinematics[32], but lead to a undesirable complex formulation. In this work, quaternions will be extensively adopted in the formulation of the attitude guidance algorithm presented in Chapter 4 for the kinematic representation, and in Chapter 5 for the implementation of the 6 Degrees of Freedom (DoF) simulator. Anyway, the attitude constraint are implemented considering directly the DCM, to avoid ambiguity.

2.2.2 Attitude guidance and attitude motion planning

The definition of a proper attitude motion planning or a refined attitude guidance is necessary in some mission scenarios where the spacecraft, that

shall perform large slew maneuvers, has some requirement in terms of attitude constraint. These attitude constraints, that in literature and within this work are defined by keep-out cones, can be a consequences of the embarked payload, such as telescopes or optical/thermal payloads, or navigation instruments, such as star tracker or sun sensors. Moreover, practical constraints, such as the one coming from actuators limitation, shall be kept into account. Most of the existing methods for attitude motion planning that are capable to keep into account keep-out cones are classified into four main categories, listed below. The first group is based on potential functions having a minimum at the desired attitude and large values around the exclusion cones. These functions are adopted to design a control laws that are based on Lyapunov theory [32, 33]. The advantage of these family of algorithms is that they are computationally light, but the convergence on global minimum depends on the initial guess. The second group uses path planning strategies to look for a path given the initial and final attitude while avoiding the forbidden pointing regions. The path planning is achieved through a discretization of the trajectory, and then by looking for a feasible path [34, 35, 36]. The third group consists of algorithms that take advantages of geometric relations to design the trajectories that are able to avoid the keep-out cones [37, 38]. These methods are quite simple and benefit from a fast execution time. The drawback, in most cases, is that they are able to avoid only one keep-out cone. The last class includes algorithms in which the constrained attitude motion problem is written as an optimal control problem, obtaining a non-convex optimization problem to be solved using a global optimization approach, an indirect method or a gradient based one [39, 40]. Within this work, a novel semi-analytical guidance method for motion planning with attitude and practical constraints is presented in Chapter 4.

2.3 Flexible Dynamics

Even if CubeSat are typically considered as rigid body, the increase of the pointing performances combined with the growing size of potentially flexible elements, such as solar panels and antennas, the effect of flexibility on the attitude performance for this class of objects could become an interesting research field in the incoming years. Looking at the components that are generally investigated in the flexible motion of a large spacecraft, it is possible to identify some common structural elements, such as booms, plates, concentrated spring-mass systems and eventually partially filled tanks [41]. These elements, using a proper discretization model, are typically combined to derive the EoM of the flexible model of the spacecraft, following a multi-body approach [42, 43]. A key element in the inclusion of the flexible effect in the EoM of the spacecraft, that define the complexity and the fidelity of the model, is the discretization approach followed in the modeling of the structural elements. The different approaches that can be followed to modeling the structural elements above-

mentioned can be summarized in three major categories, based on the way in which the mass, stiffness and dumping parameter are considered, that are here reported in ascending order of model fidelity [44, 45]:

- *LMM.* Following this approach, lumped masses are connected to the rigid structure of the spacecraft using massless springs, to replicate the real dynamics of a pseudo vibrational mode. This method, which in general is simple to implement, unfortunately lacks of theoretical rigor, and can often lead to results that are not faithful to reality. Moreover, the selection of the stiffness of the springs, the value of the masses and the position/direction in which the oscillators are mounted is not straightforward except in the simples cases, since the real vibrational modes of the structure should be respected, as well as the forces exchanged. Even in this last case in which the vibrational modes and the torques/forces induced by the flexible motion are well approximated, the LMM approach is not able to recover the flexible displacement of the structures, that is of fundamental interest for some applications (e.g. payloads mounted on booms). Figure 2.1 shows an example of LMM modeling of a rigid hub with a cantilever flexible appendage, in which, given the number of modes that is desired to replicate, it is important to underline the lack of rigor in the definition of the position x where the spring is mounted as well as the value of the mass m_i and the elastic constant k_i.

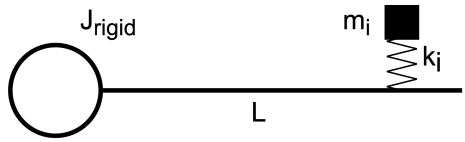


Figure 2.1: Example of a LMM modeling of a flexible appendages undergoing large planar motion

- *LPM.* This family of discretization procedures includes a quite large number of methods, sharing the fact of having a clear spatial separation between the elastic and inertial elements. One of the most commonly used method [44] relies on a representation of the system with discrete lumped masses interconnected by continuous and massless elastic elements. Once the discretization is defined, the equation of motion can be easily written using the Principle of Virtual Work (PVW), or the Lagrange equations. Figure 2.2 shows an example of LPM modeling of a rigid hub with a cantilever flexible appendage. Also this method lacks of a systematical and rigorous approach, due to the strong subjectivity in the positioning of the concentrated masses, that could lead to poor results.

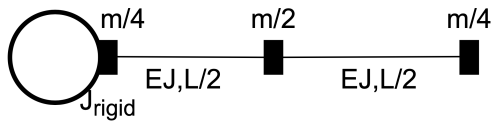


Figure 2.2: Example of a LPM modeling of a flexible appendages undergoing large planar motion

- *DPM.* Following this approach, the inertial and flexible properties are *distributed* along the structure, leading to a continuous problem governed by a system of Partial Differential Equations (PDE). Figure 2.3 shows an example of DPM modeling of a rigid hub with a cantilever flexible appendage.

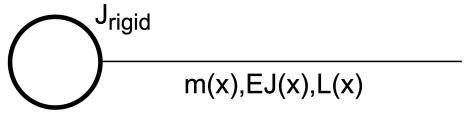


Figure 2.3: Example of a DPM modeling of a flexible appendages undergoing large planar motion

By their own nature, such systems posses infinite degrees of freedom (namely the elastic displacement and elastic displacement rate as function of the position $\boldsymbol{x}$ and time). A wide number of methods can be successfully applied in the discretization of DPM, among which the most famous are surely the RG discretization and the Finite Element Method (FEM). The first method relies on the approximation of the unknown elastic displacement field by means of a finite linear combination of appropriate spatial functions, typically stacked into the matrix of functions $\boldsymbol{N}(\boldsymbol{x})$. The coefficients of the linear combination $\boldsymbol{u}(t)$, that are time dependent, are the new set of coordinates describing the dynamics of the problem, as in Eq. 2.3 [44].

$$\boldsymbol{s}(\boldsymbol{x},t)_{flex}^{LocalRF} = \begin{bmatrix} \mathbf{u}(\mathbf{x},t) \\ \mathbf{v}(\mathbf{x},t) \\ \mathbf{w}(\mathbf{x},t) \end{bmatrix} = \boldsymbol{N}(\boldsymbol{x})\boldsymbol{u}(t) \tag{2.3}$$

The EoM are then derived adopting the PVW or the Lagrange equation, as will be largely discussed in Chapter 6. The real advantages of this method are the complete theoretical rigors, and its capability to obtain, in most of the cases and selecting the appropriate trials functions, a good

approximation of the reals dynamics, at least for the lower frequencies, using a reduced number of degrees of freedom. Following this approach, the refinement of the solution can be achieved increasing the number of trials functions, and consequently also the number of coordinates, used in the linear combination to approximate the spatial displacement field. If only one term is employed, it is known also as Ritz-Rayleigh method. The biggest disadvantage of the RG method is the difficulty to find appropriate trials functions for complex geometrical shape. To overcome this shortcut, the FEM is employed. Quite the contrary, the FEM abandons the wish discterize the structure with a linear combination of spatial functions covering the entire domain, preferring to divide it in portion sufficiently small to be suitable for piecewise continuous functions, that are only locally non-null. Therefore FEM can be seen as a particular application of the RG method [45]. The refinement of the solution, in this case, passes through a refinement of the geometrical mesh that drives the discretization of the displacements field. It turns out in a higher number of degrees of freedom with respect to the classical RG method, with an increased computational cost. Even if it is beyond the aim of this work, it is important to recall that the particular band structure of the resulting mass and stiffness matrices, combined with a wide number of methods that can be employed to reduce the computational load of FEM [46], make it nowadays the gold standard for structural analysis. Since the aim of the flexible modeling of this work is more related with the guidance and control, rather than the structural analysis, the RG discretization was selected for the flexible elements, as discussed in Chapter 6.

2.4 Heuristic Optimization

The implemented models and the scenarios that will be discussed within this work are, in general, very complex from a computational point of view. For example, a mission scenario with numerous satellites to release, as the one discussed in Chapter 3, or the optimization of a highly constrained maneuver, as the one presented in Chapter 4 would lead to problems with a really large and non mono-convex solution space in a not simply connected domain. The deterministic solution of such problems requires the knowledge of an initial guess that is in the small basin of the global optimum, that is typically unknown a-priori. This typically leads to the impossibility to converge on the optimal solution even in extremely large computational time. For these reasons, it is necessary to take optimization methods that trade optimality for speed into account, and that are able to efficiently look for optimal regions in such complex domains. Heuristic methods [47] do not grant to find the optimal solutions to the problem but the low computational effort required to achieve this near-optimal solution makes them very attractive for the purpose of the work.

Most complex problems like this variant of the VRP require the evaluation of an immense number of possibilities to determine an exact solution, with a consequent increase of the computational time. In addition to this, some problems may require to be solved much more than one time, depending on their use and goal. For this reason, it is useful to have an algorithm whose solution can be found in a relatively short amount of time. Heuristics play an effective role in such problems by finding a way to reach solutions with reasonable computational effort. In particular problem-independent algorithms are often referred to as "metaheuristic" algorithms. There is a relatively wide number of metaheuristic algorithms that could be adopted to solve the here proposed problems and it is important to select a performing one. Each algorithm is characterized by a particular set of tuning parameters that could be optimally set to maximize its efficiency. In the last years, several metaheuristic methods have been developed and they can be classified in many ways. One of the most significant classification [48] is the following:

- *Trajectory methods:* these algorithms work on one single solution at a time, describing a trajectory in the search space during the search process. They encompass local search-based metaheuristics.

- *Population-based methods:* these algorithms, on the contrary, deal in every iteration with a set of solutions, therefore providing a way to efficiently explore the search space.

While the first class is suitable for a more refined exploration of a promising area, the second ones are preferred to find near-optimal areas in the search space [48]. Due to the high dimension of the solution space of some of the problems treated in this work, population-based methods will be preferred to the trajectory ones for this application; among all the possible algorithms, the Genetic Algorithm (GA) and the Particle Swarm Optimization (PSO) are surely the most famous. The research on such family of algorithms starts in 1975 by Holland [49] and due to the promising results it was successfully addressed towards optimization problems. GAs take their inspiration from the biological evolution of species inside an environment. The natural environment is replaced by the problem itself and the individuals (also called chromosomes) represent the candidate solutions. At each iterations, new individuals are generated through recombination and mutation of previous ones, imitating Darwin's theory of natural selection. The individuals with higher fitness have more probability to survive to the next generation. Differently from GA, PSO does not rely on probability but on social behavior, implementing rules such as neighbor velocity matching and acceleration by distance [50]. PSO was indeed inspired by the movement of birds in flocks [51]. Even though being classified as an evolutionary algorithm at the beginning, its theoretical background along with its great potential gave origin to a new category of algorithms

known as Swarm Intelligence (SI). Among the population-based methods, the PSO was extensively adopted within this work. A multi-objective version of PSO, known as Multi-Objective Particle Swarm Optimization (MOPSO), opportunely modified to take into account integer variables, is adopted for the solution of multi-objective scenarios.

CHAPTER 3

Orbital Guidance

In this chapter a novel approach to the CubeSat CoM guidance is presented. More in detail, Section 3.1 is devoted to the presentation and analysis of the developed algorithm, with the help of some practical example. Section 3.2 is devoted to the application of the developed algorithm to the problem of multi-injection SmallSat constellations and multi mission.

3.1 Guidance algorithm

Within this chapter a novel shape-based algorithm applicable to Earth-centered trajectory optimization problems is proposed. The working principle of the algorithm is to map the complex multi-revolution problem into a high number of simpler single revolution trajectories, efficiently solved thanks to a new 3D shape-based method. To this end, a proper number of intermediate orbits is introduced and accurately located in order to not to exceed a threshold settled on the required thrust. The analysis of the proposed algorithm is structured as follows: in section 3.1.1 the developed single revolution algorithm is discussed; in section 3.1.2 the multi-revolution planeto-centric and interplanetary extensions are presented, while section 3.1.3 is devoted to the analysis of some test cases.

3.1.1 Single Revolution Algorithm

The section presents the developed single-revolution shape-based algorithm. The mathematical formulation is derived in both fixed and free Time Of Flight (TOF) formulations, starting from the equation of motion of a thrusting spacecraft via a nonlinear interpolation between departure and target orbits. The working principle of the algorithm lies in setting the spacecraft-to-attractor distance and the declination above the reference plane of the trajectory (respectively s and δ) by applying a non-linear interpolation between their values at the departure and target orbits. The kinematic and the dynamics are then recovered through a semi-analytic procedure. The discussion is organized so that, after analyzing the requirements, the equations of motion are parametrized, the non-linear interpolation is introduced, and the architecture of the algorithm is explained. Some kinematic relations, even if important in the numerical implementation, are not fundamental for the comprehension of the work. For sake of simplicity these equations are reported only in Appendix A.

3.1.1.1 Requirements

To drive the design of an algorithm which handles highly constrained dynamics, a peculiarity of the planeto-centric orbits whenever low thrust is selected, some requirements have been set. The most relevant are as follows:

1. The algorithm shall link two different states (expressed as MEE [8] or KP) with the possibility to impose the TOF.
2. The algorithm shall work with any couple of orbits that can be physically connected with zero radial thrust [13], including polar and retrograde orbits.
3. The thrust acceleration shall be exactly computed via an analytical procedure.
4. The thrust and mass profiles shall be computed via numerical integration of Tsiolkovsky equation [52].
5. The required thrust shall tend to zero as the distance between initial and final orbits decreases, regardless of the eccentricity and departure or arrival anomaly:

$$\lim_{\Delta KP \to 0} \max \left(\frac{|T|}{m} \right) = 0 \qquad \forall \quad e, i, \theta_1, \theta_2. \tag{3.1}$$

The last requirement is mandatory to obtain feasible solutions in planeto-centered mission scenarios: whenever not satisfied, the maximum thrust cannot be arbitrarily fixed .

3.1.1.2 Low thrust equations parametrization

This section formalizes the parametrization of the 3D EoM for a thrusting spacecraft in cylindrical coordinates. The assumption on small displacements for the out of plane motion, imposed by Wall in [13], is here removed. The exact EoM for a thrusting spacecraft [5] [6] can be written as

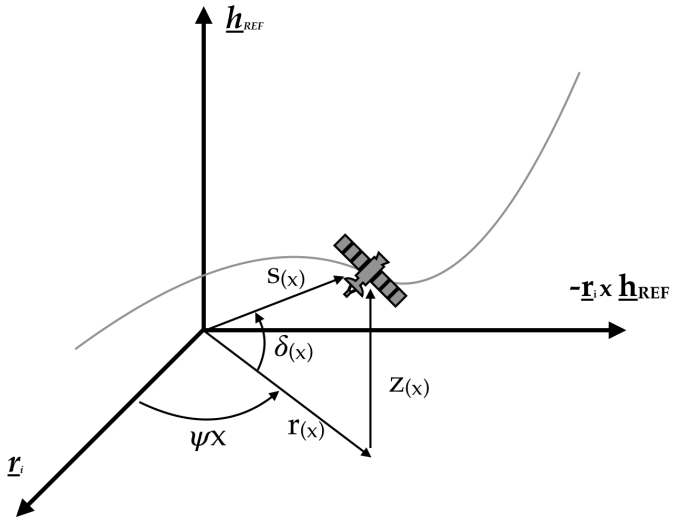


Figure 3.1: Spacecraft position

$$\begin{cases} \ddot{r} - r\dot{\theta}^2 = -\frac{\mu}{s^3}r + \frac{T_{IN}}{m}sin\alpha \\ r\ddot{\theta} + 2\dot{r}\dot{\theta} = \frac{T_{IN}}{m}cos\alpha \\ \ddot{z} = -\frac{\mu}{s^3}z + \frac{T_{OUT}}{m}. \end{cases} \tag{3.2}$$

The working principle that lies behind the proposed algorithm is to parametrize every quantity involved in Eq. 3.2 as a function of the non-dimensional anomaly x, defined as

$$x(t) = \frac{\theta(t)}{\psi}, \tag{3.3}$$

and to then compute the thrust and mass time history, adopting a reverse dynamic approach [7]. Physically, $x(t)$ is the angle between the initial position vector and the projection of the spacecraft position vector on the reference plane, normalized by the total transfer angle ψ, as reported in Figure 3.1.

The spacecraft 3D motion is defined by the parametrization of the in-plane projection of the radius r and the out-of-plane displacement z, as in

$$\begin{cases} r = r(x) \\ z = z(x). \end{cases} \tag{3.4}$$

This parametrization, which is not unique, is presented later on in this section. To parametrize the EoM the first and second-time derivatives for the in-plane angular displacement, the in-plane radius and the out-of-plane displacement are first computed, as reported in

$$\begin{cases} \dot{\theta} = \frac{d\theta}{dt} = \psi\dot{x} \\ \ddot{\theta} = \frac{d^2\theta}{dt^2} = \psi\ddot{x} \end{cases} \tag{3.5}$$

and

$$\begin{cases} \dot{r} = \frac{dr}{dt} = \frac{dr}{dx}\frac{dx}{dt} = r'\dot{x} \\ \ddot{r} = \frac{d^2r}{dt^2} = \frac{d}{dt}\left(r'\dot{x}\right) = r''\dot{x}^2 + r'\ddot{x} \\ \dot{z} = \frac{dz}{dt} = \frac{dz}{dx}\frac{dx}{dt} = z'\dot{x} \\ \ddot{z} = \frac{d^2z}{dt^2} = \frac{d}{dt}\left(z'\dot{x}\right) = z''\dot{x}^2 + z'\ddot{x}. \end{cases} \tag{3.6}$$

Then, all kinematic quantities can be substituted into Eq. 3.2, giving as result

$$\begin{cases} r''\dot{x}^2 + r'\ddot{x} - r\psi^2\dot{x}^2 = -\frac{\mu}{s^3}r + \frac{T_{IN}}{m}sin\alpha \\ 2\psi r'\dot{x}^2 + r\psi\ddot{x} = \frac{T_{IN}}{m}cos\alpha \\ z''\dot{x}^2 + z'\ddot{x} = -\frac{\mu}{s^3}z + \frac{T_{OUT}}{m}. \end{cases} \tag{3.7}$$

According to the first two equations listed in Eq. 3.7, the formulation for the second time derivative of the non-dimensional anomaly is extracted as follows

$$\begin{cases} \ddot{x} = \frac{1}{r'}\left[-\frac{\mu}{s^3}r + \frac{T_{IN}}{m}sin\alpha - r''\dot{x}^2 + r\psi^2\dot{x}^2\right] \\ \ddot{x} = \frac{1}{r\psi}\left[-2\psi r'\dot{x}^2 + \frac{T_{IN}}{m}cos\alpha\right]. \end{cases} \tag{3.8}$$

To analytically compute the x time derivative, the dependency from the thrust per unit mass in Eq. 3.8 must be removed: to this end, the in-plane thrust should be imposed as tangential only [7]. Indeed, in this last case, the thrust angle equals the flight path angle that can be easily computed using

$$\tan\alpha = \tan\gamma = \frac{v_r}{v_\theta} = \frac{\dot{r}}{r\dot{\theta}} = \frac{r'}{r\psi}. \tag{3.9}$$

By merging Eq. 3.9 and Eq. 3.8 and removing the thrust dependence, the square of x time derivative is made explicit, as follows

$$\dot{x}^2 = \frac{\mu r}{s^3\left(r\psi^2 - r'' + 2\frac{r'^2}{r}\right)} = \frac{Nu}{De}. \tag{3.10}$$

From Eq. 3.10, the second time derivative for the non-dimensional anomaly can be obtained as

$$\begin{cases} \ddot{x} = \frac{1}{2}\left(\frac{Nu' - \dot{x}^2 De'}{De}\right) \\ Nu' = \mu r' \\ De' = 3\frac{s'}{s} De + s^3\left(r'\psi^2 - r''' + \frac{2rr'r'' - r'^3}{r^2}\right). \end{cases} \tag{3.11}$$

By inserting the newly derived time-dependent quantity relationships into Eq. 3.7, the thrust per unit mass can be computed as

$$\begin{cases} \frac{T_{IN}}{m} = \frac{1}{\cos\gamma}\left(2\psi r'\dot{x}^2 + r\psi\ddot{x}\right) \\ \frac{T_{OUT}}{m} = z''\dot{x}^2 + z'\ddot{x} + \frac{\mu}{s^3}z. \end{cases} \tag{3.12}$$

The mass time history comes from numerically integrating the Tsiolkovsky equation, as [53]

$$\begin{cases} \left|\frac{T}{m}\right| = \sqrt[2]{\left(\frac{T_{IN}}{m}\right)^2 + \left(\frac{T_{OUT}}{m}\right)^2} \\ \frac{dm}{dt} = -\frac{\left|\frac{T}{m}\right|m}{I_s p g_0}. \end{cases} \tag{3.13}$$

Moreover, as highlighted in Eq. 3.14, the integral on the variation of the non-dimensional anomaly, which is numerically solved, leads to the time vector [7]

$$t = \int_0^t \mathrm{d}\tau = \int_0^1 \frac{1}{\dot{x}}\,\mathrm{d}x. \tag{3.14}$$

It is important to underline that, differently from Wall [13], the here proposed parametrization allows representing the exact motion of a thrusting spacecraft, removing any assumption on small displacements from the reference plane assumption.

3.1.1.3 Non-Linear Interpolation

The EoM parametrization discussed so far needs a parametric representation of the trajectory in cylindrical coordinates, as in Eq. 3.4. In the here presented algorithm the parametrization is obtained via a nonlinear interpolation between the departure and arrival orbits. This interpolation is set in spherical coordinates ($s(x)$, $\delta(x)$, ψx, see Figure 3.2) by means of an interpolating function

$\chi(x)$, as shown in

$$\begin{cases} s_{(x)} = \left(s_{2(x)} - s_{1(x)}\right)\chi_{(x)} + s_{1(x)} \\ \delta_{(x)} = \left(\delta_{2(x)} - \delta_{1(x)}\right)\chi_{(x)} + \delta_{1(x)}; \end{cases} \tag{3.15}$$

it is then mapped into cylindrical coordinates using

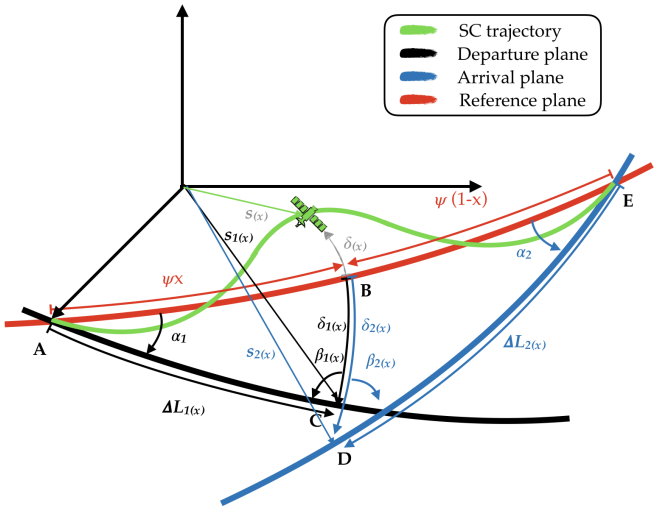


Figure 3.2: Non-linear interpolation

$$\begin{cases} r = s\cos\delta \\ z = s\sin\delta. \end{cases} \tag{3.16}$$

To solve the parametrized equation of motion, the first, second and third derivatives of $r(x)$ and the first two derivatives of $z(x)$ shall be calculated using

$$\begin{cases} r' = -s\delta'\sin\delta + s'\cos\delta \\ r'' = -\left(2s'\delta' + s\delta''\right)\sin\delta + \left(s'' - s\delta'^2\right)\cos\delta \\ r''' = -\left[3\left(s''\delta' + s'\delta''\right) + s\left(\delta''' - \delta'^3\right)\right]\sin\delta + \\ \qquad + \left[s''' - 3\delta'\left(s'\delta' + s\delta''\right)\right]\cos\delta \end{cases} \tag{3.17}$$

and

$$\begin{cases} z' = s\delta'\cos\delta + s'\sin\delta \\ z'' = \left(2s'\delta' + s\delta''\right)\cos\delta + \left(s'' - s\delta'^2\right)\sin\delta. \end{cases} \tag{3.18}$$

The computation of

$$\begin{cases} s' = \Delta s'\chi + \chi'\Delta s + s_1' \\ s'' = \Delta s''\chi + \chi''\Delta s + 2\Delta s'\chi' + s_1'' \\ s''' = \Delta s'''\chi + \chi'''\Delta s + 3\Delta s''\chi' + 3\Delta s'\chi'' + s_1''' \end{cases} \tag{3.19}$$

and

$$\begin{cases} \delta' = \Delta\delta'\chi + \chi'\Delta\delta + \delta'_1 \\ \delta'' = \Delta\delta''\chi + \chi''\Delta\delta + 2\Delta\delta'\chi' + \delta''_1 \\ \delta''' = \Delta\delta'''\chi + \chi'''\Delta\delta + 3\Delta\delta''\chi' + 3\Delta\delta'\chi'' + \delta'''_1 \end{cases} \tag{3.20}$$

enables the evaluation of the derivatives of the spherical coordinates included in Eq. 3.17 and Eq. 3.18.

The full set of equations needed to compute the geometrical quantities of the initial and final orbits required to solve Eq. 3.19 and Eq. 3.20 are reported in Appendix A.

3.1.1.4 the algorithm architecture

Figure 3.3 depicts the implemented algorithm architecture, made up of five main blocks plus two optional, activated whenever requested by the specific mission scenario.

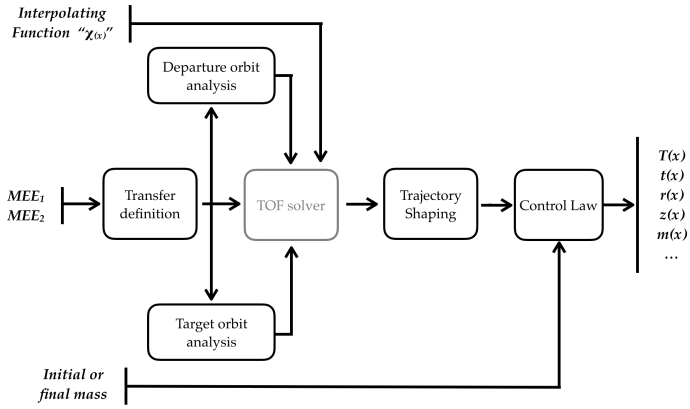


Figure 3.3: Architecture

A short description of each block is here reported:

- **Transfer definition:** This block takes as inputs the departure and arrival orbital states; it computes the reference frame and the transfer angle. The selected reference frame is composed by:
  - **First axis:** normalized initial position of the satellite.
  - **Third axis:** normalized cross product between the initial and final positions.
  - **Second axis:** Orthogonal to the first and third axis, to get a right-handed RF

- **Departure/Target orbit analysis:** These blocks take as input the data coming from the Transfer definition one. They compute some important geometrical quantities exploited for the shape interpolation. The output includes all geometric quantities of the initial and final orbits.

- **TOF solver:** This block manages the time of flight. It activates only whenever a TOF constrained mission scenario is considered: to this end, the shape itself shall contain a further degree of freedom. To force the imposed time of flight to be respected it is necessary to numerically solve

$$TOF - \int_0^1 \frac{1}{\dot{x}(a)} \, \mathrm{d}x = 0, \tag{3.21}$$

  through a Newton algorithm, initialized with $a = 0$.

  At each step of the Newton solver, adopted for integration of Eq. 3.21, a Cavalieri-Simpson integration scheme is adopted to preserve both a good precision and a limited Central Processing Unit (CPU) time. This block outputs the interpolating function, that is passed to the trajectory shaping block.

- **Trajectory shaping:** This block takes as input the geometry of departure and target orbits and the interpolating function. It computes the exact geometry of the transfer using the non-linear interpolation between departure and target orbits explained here-above. The output includes the kinematics of the trajectory.

- **Control Law:** This block takes as input the kinematics of the trajectory. The control law, as well as the time, are computed via a mixed numerical-analytical procedure. To compute the time vector, Eq. 3.14 is integrated using an high order multi-step predictor-corrector scheme (Adams-Bashford-3 Adams-Multon-4). The same integration scheme is adopted to integrate Tsiolkovsky equation [52] (Eq. 3.13) to get the mass history. It is here highlighted that both forward and backward integration schemes are supported, making the algorithm suitable to solve scenarios either with dry or wet mass imposed. The block is the most demanding in terms of computational time, as it includes two Ordinary Differenzial Equation (ODE) to be solved. In any case, it is important to underline that while typically an ODE solver spends most of the time evaluating the function to be integrated, in the framework of the proposed approach those information are directly fed into the solver (please refer to Eq. 3.14 and Eq. 3.13), being already computed by previous blocks; that helps in reducing the computational burden of the block itself. Moreover, the selection of an accurate integration scheme reduces the number of computation nodes while keeping the numerical errors as low as possible.

3.1.1.5 The interpolating functions

A key element of the developed algorithm lies in the ability of the interpolating function χ to respect the boundary conditions; to this end, the interpolating function $\chi(x)$ must be continuous with its derivatives up to the third order in the domain $[0; 1]$. Moreover, the interpolating function must satisfy the following requirements:

- **Position:** from Eq. 3.16 and Eq. 3.15 the initial and final conditions on position (in plane radius $r(x)$ and out-of-plane displacement $z(x)$) are automatically satisfied if the interpolating function satisfies:

$$\begin{cases} \chi(0) = 0 \\ \chi(1) = 1. \end{cases} \tag{3.22}$$

- **Velocity:** given the definition of the radial velocity ($\dot{r}$) and the out-of-plane velocity ($\dot{z}$) in Eq. 3.6, $r', z', \dot{x}$ must match the corresponding quantities of the initial state for $x = 0$ and final state for $x = 1$. If the previous conditions are verified, the boundary conditions on transverse velocity are automatically satisfied, being this velocity defined as $v_r = r\psi\dot{x}$. The requirements on r' and z' at the initial and final point can be directly derived from Eq. 3.17 and Eq. 3.18, and are summarized as follows:

$$\begin{cases} \chi'(0) = 0 \\ \chi'(1) = 0. \end{cases} \tag{3.23}$$

  For the initial and final conditions on $\dot{x}$, Eq. 3.10 must be considered: it contains the second derivatives of r as well. Therefore, boundary conditions on r'' must be imposed too. From Eq. 3.17 it is easy to derive the following constraints:

$$\begin{cases} \chi''(0) = 0 \\ \chi''(1) = 0. \end{cases} \tag{3.24}$$

Even if an extremely wide number of functions could satisfy those requirements, this work adopts the seventh order polynomial function

$$\chi(x) = -20x^7 + 70x^6 - 84x^5 + 35x^4, \tag{3.25}$$

and the eight-order polynomial

$$\begin{aligned} \chi(x, a_s) = a_s x^8 - (20 + 4a_s)x^7 + (70 + 6a_s)x^6 + \\ -(84 + 4a_s)x^5 + (35 + a_s)x^4 \end{aligned} \tag{3.26}$$

to solve the free TOF and the constrained TOF problems respectively.

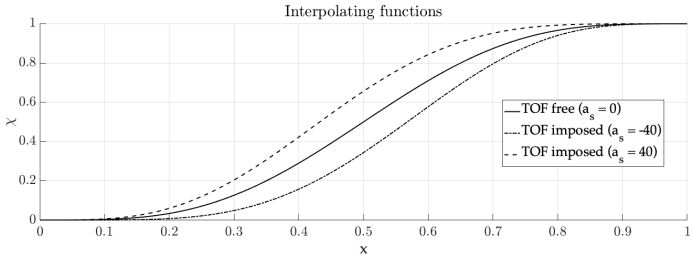


Figure 3.4: Interp. function comparison

It is easy to prove that these functions satisfy the above-mentioned boundary conditions. For sake of completeness they are plotted in Figure 3.4 for different values of the parameter a_s. Moreover, it is here remarked that those interpolation functions lead to solutions with null initial and final thrust, avoiding the typical undesired peaks at the beginning and the end of the transfer arc (see the thrust profile obtained in [13] and [7]).

The algorithm proposed runs in MATLAB R2020 on a laptop equipped with a sixth generation Intel i7 processor at 2.6 GHz, can evaluate more than 20 thousand revolution per second for free TOF scenarios and more than 8 thousand for imposed TOF problems.

3.1.2 Multi-revolution Approaches

The section discusses two possible solutions to extend the above-mentioned algorithm to the multi-revolutions trajectories. The first solves interplanetary transfers, and it is applicable whenever the number of revolutions is limited (no more than 2 or 3); the second, suitable for planeto-centric mission scenarios, increases in complexity: in fact, it manages thousands of revolutions, including also discontinuities such as eclipses.

3.1.2.1 Interplanetary scenario

Due to the low number of complete revolutions that are typically involved in low thrust interplanetary transfers, a simple but effective strategy lies in considering an 'augmented' transfer angle as given in[7][13]

$$\psi = \psi + 2\pi N_{rev}. \tag{3.27}$$

The solution is well suited for N limited to either 2 or 3, otherwise the shape of the interpolating function generates solutions with a bad distribution of the thrust peaks. This solution is extremely fast since it requires the evaluation of only one trajectory.

Interplanetary trajectory design and optimization introduces some other issues and constraints/objectives to identify and formalize. Regardless of the technology adopted, the thrust is linearly related with the electric power available on board: most of the thrusters on the market show a specific power consumption between 15 $\frac{W}{mN}$ and 40 $\frac{W}{mN}$. During an interplanetary transfer, the distance from Sun varies significantly and, in most cases, it drives the time history of the generated electric power, if photovoltaic technology is adopted on board as the primary power source [54].

Moreover, solar panels are affected by aging effects that reduce the amount of power produced during the mission. These effects can be merged together to formalize a unique constraint that, to speed up the optimizer convergence and its flexibility in the mission design, is included in the cost functions vector and is to be minimized. The solar panel area needed to accomplish the mission is the physical quantity that synthesizes all above-mentioned aspects, which is sized thanks to [54]

$$A_{SA}(x) = \frac{kT + P_{ss}}{\eta_{tot} \cos\phi \, (1-\beta)^t \, \frac{\phi_{Earth}}{s^2}}, \tag{3.28}$$

at each computational node.

The sun-angle can be either a-priori imposed or computed point-by-point from the control law, if the geometry of the spacecraft is known. Eq. 3.28 can be simpler as

$$obj = MAX \left[\frac{T(x)s(x)^2}{(1-\beta)^{t(x)}} \right], \tag{3.29}$$

to generate an objective function less dependent on the spacecraft specific sizing.

This objective function can be successfully included in a multi-objective multi-disciplinary heuristic optimization together with the fuel mass minimisation. Moreover, the admissible maximum thrust [4] can be included, formulated either as a constraint or as a further element of the objective functions vector, and is to be minimized. If the latter scheme is adopted, the Pareto front allows directly selecting the solution that fits at the best both the mission requirements and the platform constraints.

3.1.2.2 Planeto-centric scenario

The basic strategy proposed to solve the planeto-centric scenarios consists of introducing a set of intermediate keplerian orbits. That scheme opens the

possibility to formalize the actual operational case of switching off thrusters during eclipses. The problem is formalized first: the initial and final states are imposed as MEE:

$$\begin{cases} MEE_i = [p_i, f_i, g_i, h_i, k_i, L_i] \\ MEE_f = [p_f, f_f, g_f, h_f, k_f, L_f]. \end{cases} \tag{3.30}$$

Since the algorithm can work both forwards and backwards in time, two different but similar formulations are available: for sake of brevity only the forward algorithm is deeply analyzed. The steps are the following:

1. **Problem initialization:** at the first algorithmic step the spacecraft is assumed to be in its initial state, while the final state represents the desired condition, therefore

$$\begin{cases} MEE_1 = MEE_i \\ MEE_2 = MEE_f \\ m_k(0) = M_{initial} \\ t_k(0) = 0. \end{cases} \tag{3.31}$$

   with $k = 1$, as the first trajectory has still to be sized.

2. **k-th intermediate orbit setting:** the spacecraft holds in position described by MEE_1 with mass $m_k(0)$ at time after departure mapped in $t_k(0)$. The goal stays in localizing the k^{th} intermediate orbit such that the required maximum thrust equals the maximum actually available. According to

$$\begin{cases} p_k = (p_2 - p_1)\,\eta_k + p_1 \\ f_k = (f_2 - f_1)\,\eta_k + f_1 \\ g_k = (g_2 - g_1)\,\eta_k + g_1 \\ h_k = (h_2 - h_1)\,\eta_k + h_1 \\ k_k = (k_2 - k_1)\,\eta_k + k_1, \end{cases} \tag{3.32}$$

   the intermediate orbit positioning depends on the η_k parameter: the higher the η_k, the higher the gap between the current and the intermediate orbit, entailing the required thrust to be higher as well.

   By numerically solving

$$max\,(T(\eta_k)) - T_{available} = 0, \tag{3.33}$$

η_k is quantified. The term $max\,(T(\eta_k))$ in Eq. 3.33 refers to the maximum thrust required during the k^{th} trajectory, computed through the TOF free algorithm presented in 3.1.1.

Generally speaking, Eq. 3.33 numerical solution is not straightforward since continuity and solution existence is not guaranteed. An especially developed hybrid Newton-Bisection algorithm is here adopted: the algorithm tries solving the equation with the Newton method first; whenever that fails, it switches to the Bisection method; if a predefined tolerance cannot be satisfied within a given number of iterations, the equality turns into the following inequality:

$$max\,(T(\eta_k)) - T_{available} < 0. \tag{3.34}$$

This last inequality can always be solved since the developed shape-based algorithm fulfills the constraints expressed by

$$\lim_{\Delta MEE \to 0} \max\left(\frac{|T|}{m}\right) = \lim_{\eta_k \to 0} \max\left(\frac{|T|}{m}\right) = 0. \tag{3.35}$$

A computed η_k either equal or larger than one, means that the available on-board thrust suffices to reach the final position MEE_2, as per Eq. 3.32: therefore, η_k is automatically switched to one and the k^{th} trajectory is re-computed; the algorithm then stops. A computed η_k between 0 and 1 asks to set an iterative loop. The new starting position is represented by the exit from the eclipse of the k^{th} intermediate orbit, and the initial mass of the spacecraft on the $k+1$ trajectory equals the final mass of the k trajectory, namely:

$$\begin{cases} MEE_1 = MEE_k \\ MEE_2 = MEE_f \\ m_{k+1}(0) = m_k(1) \\ t_{k+1}(0) = t_k(1) + \Delta t_{eclipse}. \end{cases} \tag{3.36}$$

The time after departure at the beginning of the $k+1$ trajectory equals the arrival time of the k trajectory plus the time spent in shadow ($\Delta t_{eclipse}$).

k is then increased and the algorithm loops up to point 2. The cycle stops as soon as η_k either equals or gets larger than one.

3. **Trajectory analysis**: to preserve computational speed, the previous block limits outputs to the initial/final mass and the TOF. If more information is needed once the intermediate orbits are placed, the trajectory analysis block is activated to compute all kinematic and dynamic

quantities related to the trajectory. The rough order of magnitude for the algorithm computational time is 50%, increased whenever the current block takes place; therefore, while complex mission scenarios optimization holds, it should run on a limited number of solutions.

The backward version of the algorithm involves the same steps, with some important differences:

- The initial state is represented by the arrival. The departure state plays the role of the desired conditions. Therefore

$$\begin{cases} MEE_1 = MEE_f \\ MEE_2 = MEE_i \\ m_k(1) = M_{final} \\ t_k(1) = 0, \end{cases} \tag{3.37}$$

  is adopted instead of Eq. 3.31.

- Using Eq. 3.37, M_{final} is the final mass, and $t_k(x)$ now means 'time before arrival' instead of 'time after departure' and it is a negative quantity.

- The k^{th} intermediate orbit is integrated itself backward in time, therefore states are updated according to

$$\begin{cases} MEE_1 = MEE_i \\ MEE_2 = MEE_k \\ m_{k+1}(1) = m_k(0) \\ t_{k+1}(1) = t_k(0) - \Delta t_{eclipse}, \end{cases} \tag{3.38}$$

  which is adopted instead of Eq. 3.36.

  The final mass to be imposed to the $k^{th} + 1$ single revolution trajectory corresponds to the initial mass computed for the k^{th} revolution, as done to manage the time variable.

To include both forward and backward directions into the trajectory design, the algorithm broadens applicability to a wider set of scenarios. The algorithm block diagram and flow is offered in Figure 3.6. Figure 3.5 sketches how the algorithm works for a multi-revolutionary geocentric trajectory with a high thrust to weight ratio.

3.1.3 Test Cases

The section discusses some test cases. All runs are accomplished with the MATLAB R2017b coded algorithm on a laptop equipped with a sixth generation Intel i7 processor working at 2.6 GHz with no parallelization.

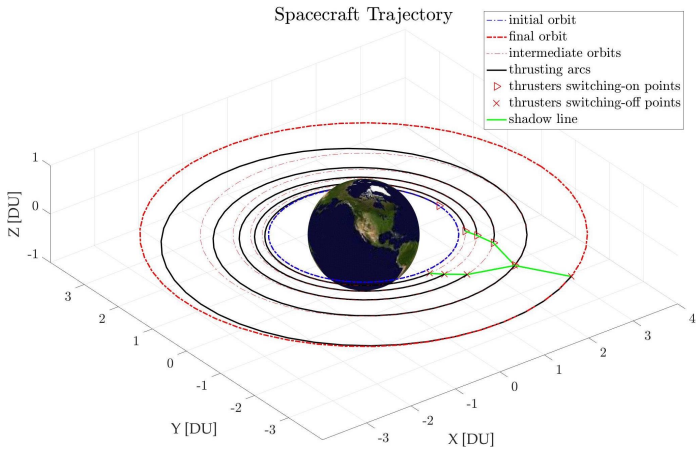


Figure 3.5: Example of multi-revolution application

3.1.3.1 Very low thrust LEO-GEO transfer

To perform a comparison between the proposed algorithm verification and literature results, a very low thrust acceleration ($2 \cdot 10^{-4} m/s^2$) planar Low Earth Orbit (LEO) to GEO transfer scenario has been implemented [55]. Parameters are tuned according to Sreesawet et al. [55], and are reported in Table 3.1.

Table 3.1: Simulation parameters

Parameter	Value
Initial Mass [kg]	5000
Thrust [N]	1,16
Departure orbit elevation [km]	2000
Specific impulse [s]	1788

The comparison between the solution found by the presented shape-based algorithm and the trajectories sized by Sreesawet et al. [55] is summarized in table 3.2. The shape-based algorithm converges to very similar results in terms of fuel consumption (1% better than the solution found by Sreesawet et al.) with a computational time that is two orders of magnitude lower. A gap between 10% and 20% exists on TOF. This undesired gap is mainly due to the limitation imposed by the continuity of the thrust profile.

3.1.3.2 Orbital Raising to GEO

The scenario explores the case of raising a satellite to GEO exploiting electric propulsion for energy increase. A satellite dry mass of 800 [kg], equipped with a propulsion unit with 3800 [s] specific impulse and 0.5 [N] thrust, departs from

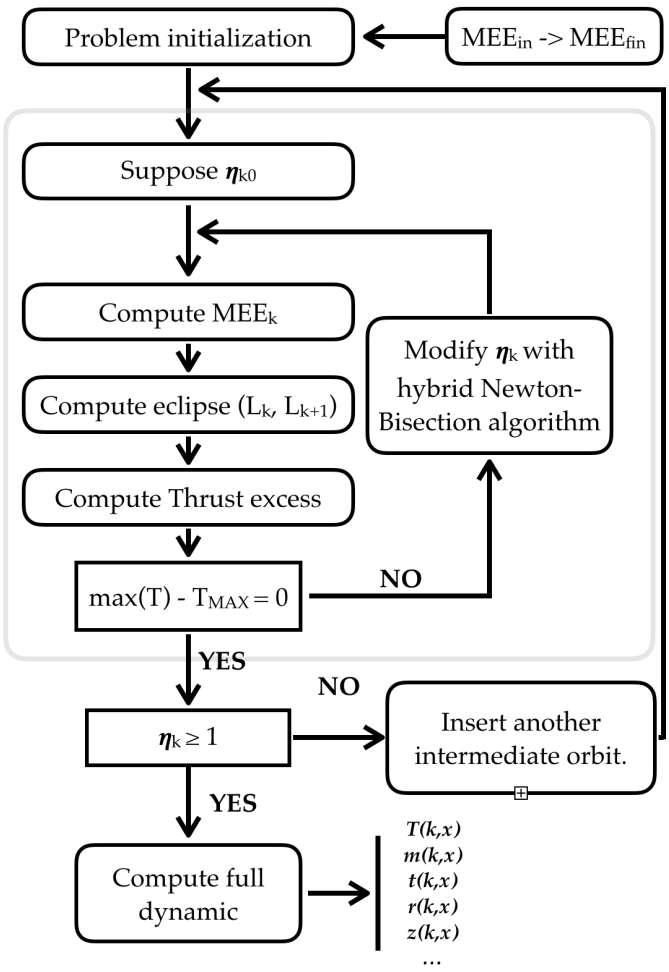


Figure 3.6: Multi-revolution algorithm architecture

a parking orbit gained by the European VEGA launcher [56], to move to GEO with low thrust. The inclination of the parking orbit is fixed at 5.4 $[deg]$, the minimum reachable from Kourou without a plane change, and the standard parking orbit plane for VEGA [56]. The apocenter and pericenter radii are degrees of freedom for the optimization process. Their values can range from 1.03 $[DU]$ to 6.6108 $[DU]$, including therefore any possible intermediate orbit between LEO and GEO. Due to the Earth rotation axis inclination, eclipses encountered by a satellite above LEO orbits are strongly affected by the period of the year: a satellite in GEO goes in Earth shadow only nearby the equinoxes [6]. Since in this example the spacecraft is supposed to thrust only in sunlight, the solution depends on the season at GEO arrival too, therefore the two opposite cases (arrival at the equinoxes or at the solstices) have been analyzed. To better highlight the relevant effect the eclipses insertion has on the final

Table 3.2: Comparison of results

	Prinetto et al.		Sreesawet et al.	
Eclipses	on	off	on	off
Final mass [kg]	4026	4025	3980	3993
ToF [days]	267	220	216	202
CPU time [s]	0.37	0.18	15.2	12.5

solution, comparison is offered with results with the shadowing constraint removed.

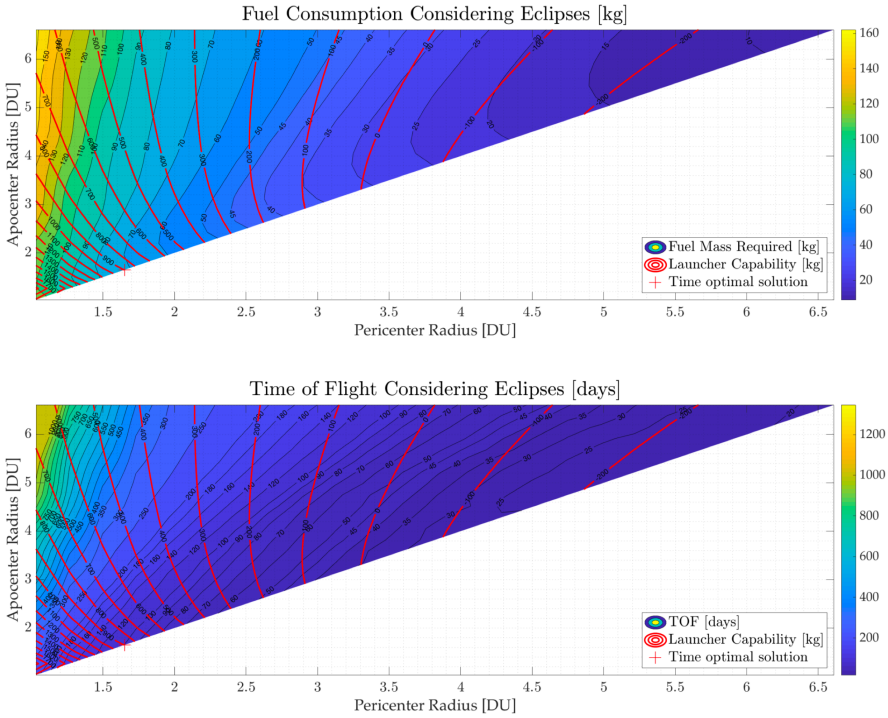


Figure 3.7: Orbit rising to GEO

Figure 3.7 depicts the most significant scenario: the GEO injection epoch occurs in the Solstice proximity, and an impulsive disposal maneuver is included in the optimization process to take into account the mandatory reenter of the upper stage 'AVUM' in atmosphere, according to debris mitigation guidelines [57]. Red lines represent the launchable mass with VEGA launcher: since a complete set of information for the launcher is not available, data is extrapolated applying the Tsiolkovsky equation to the launcher upper stage (AVUM) from the reference orbit available on the user-manual [56]. The time-optimal problem is solved for all the above-mentioned cases adopting a Nelder-Mead simplex algorithm [58] modified with a penalty method to force the solution detecting

an initial mass lower than the launchable mass on the same orbit. Being the number of revolutions discrete, the time of flight is not continuous: therefore the adoption of a derivative free algorithm is mandatory.

Table 3.3: Time-optimal constrained solutions

Eclipse	Yes		Yes	No
De-orbiting	No	No	Yes	No
Epoch	Equinox	Solstice	Solstice	—-
Fuel Mass [kg]	62.30	61.72	71.5	61.59
TOF [days]	83.89	79.84	98.47	66.5
Revolutions [-]	280	279	425	214
r_p park. [DU]	1.8994	1.9019	1.6517	1.8816
r_a park. [DU]	1.9030	1.9029	1.6524	1.9207
CPU time [s]	23	31	41	12

Table 3.3 compares solutions obtained for similar scenarios by slightly changing the framework in terms of arrival epoch, eclipse and disposal maneuver constraints activation. The fuel consumption is similar for scenarios in the table, while the TOF increases between 20% and 25% whenever the no-thrust in shadow constraint is active. Sensitivity to the injection season is contained: the reason why is that, even if nearby equinox eclipses occur at any distance from ground, the fraction of time spent in shadow decreases with the radius. The reported CPU time highlights the algorithm to very quickly find sub-optimal solutions for multi-revolutions discontinuous trajectories: as far as table 3.3 scenarios are considered, 50 computational nodes per revolution have been used for the optimization processes and 100 for plotting the final trajectory. The different CPU times reflect the different number of required revolutions to get to GEO, depending on the no-thrust in shadow constraint activation: in fact, it increases the solver complexity and asks for longer thrusting arcs to recover for time lost during the shadowed ballistic arcs. All optimization processes have been initialized with the reference VEGA parking orbit (200 $[km]$ x 1500 $[km]$ height LEO orbits). Literature offers no database with time-optimal solutions of LEO-GEO raising problems including eclipses in the model, therefore bench-marking is feasible with the no-thrust in shadow constraint disabled.

While the computed optimal solution is slightly more expensive than those from literature, as will be shown in the section dedicated to the test cases, the CPU time is orders of magnitude lower[59]. Figure 3.8 shows the attainable 3D trajectory, with a quasi-circular switching orbit with a radius of 2 [DU], with no-thrust in shadow and disposal maneuver constraints active; that indeed, represents the most challenging scenario among those listed in Table 3.3. It is worth noting the intermediate orbits higher density in Earth proximity. Moreover, the shadow region (in green) is distorted because of the Earth

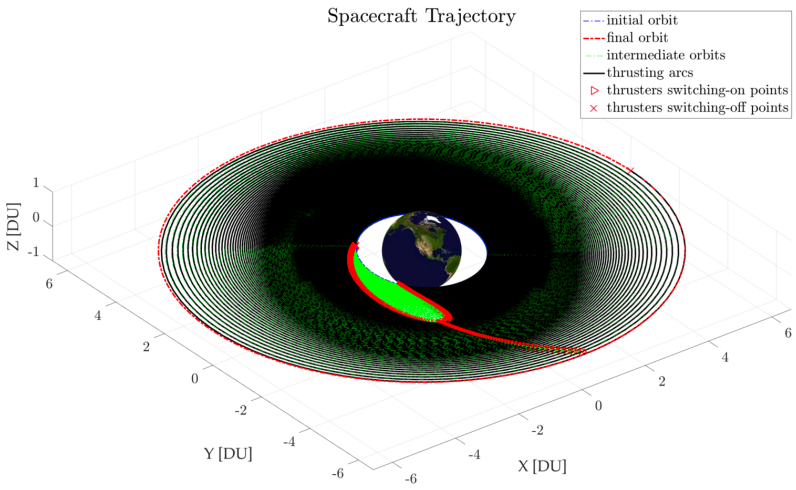


Figure 3.8: Time optimal trajectory

revolution and, less significant, because of the plane changes; the no-thrust region shrinking with altitude, can also be appreciated.

3.1.3.3 Earth-Mars Rendezvous

The Earth-Mars rendezvous problem is a classical scenario for the validation of low thrust algorithms [15].

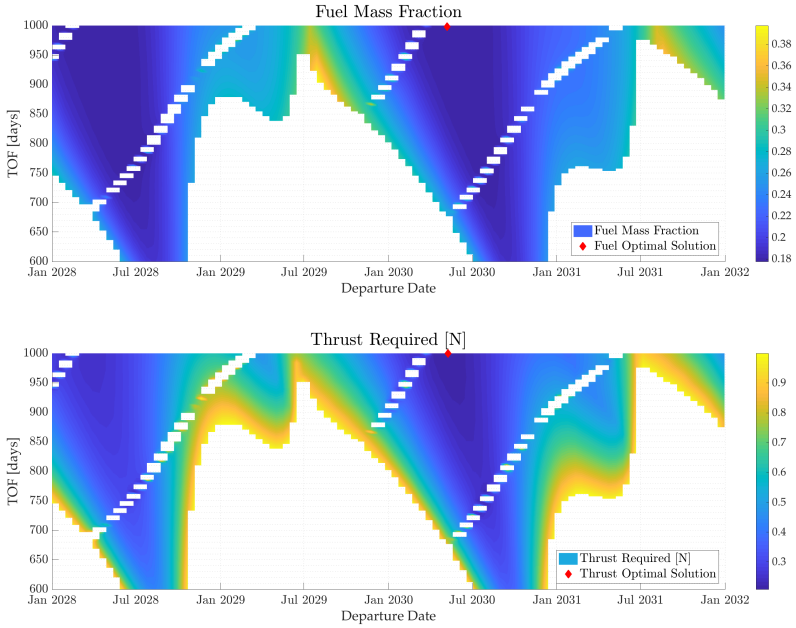


Figure 3.9: Launch opportunities

A spacecraft dry mass of 1000 [kg], a specific impulse of 3000 [s] and a maximum thrust of 0.22 [N] are settled. The requested thrust and the fuel mass fraction over the whole search domain are reported in Figure 3.9; regions where either the thrust exceeds 1 $[N]$ or the fuel mass fraction exceeds 0.5 $[-]$ are white coloured. The optimal thrust and fuel mass fraction are also reported, marked in red: it clearly appears that the search domain proposed by Vasile [19] must be enlarged, being the found optima located at its borders. Anyway, staying stuck on that search domain, two convenient regions exist, in which thrust and fuel mass fraction, together with TOF, keep low. The computed fuel optimal solution approaches Vasile and De Pascale results, with the same fuel mass fraction of 0.177 [-] and a 1000 days TOF, slightly higher than the reference; the departure date is the 8^{th} of May 2030. The CPU time, exploiting the standard MATLAB genetic algorithm library with a 100 individuals population for the optimization and a stopping criteria tuned on the average change of the cost function, is lower than 10 seconds.

3.1.3.4 Earth-Nereus Rendezvous

The asteroid scenario was selected to underline the ability of the shape-based algorithm to find near optimal solution when high elliptical orbits are considered. Nereus is a Near Earth Object (NEO) with an highly elliptical orbit on a plane slightly above the ecliptic. Its pericenter is located near the Earth's pericenter, while the apocenter is at 1.5 [AU]; therefore, a quasi-ballistic solution with a non-zero escape velocity is expected whenever either the spacecraft fuel mass fraction or the requested thrust level is selected as a cost function. To facilitate the best trajectory finding, an extremely wide search space is considered: the degrees of freedom and their ranges are reported in Table 3.4.

Table 3.4: Search domain

	Range	Optimum
Dep. Date [dd/mm/yyyy]	2030 to 2050	09/02/2042
Time of Flight [days]	500 to 1500	690.5
Number of revolutions [-]	0 to 2	1
v_{inf} departure [km/s]	0 to 6	5.93
v_{inf} in-plane angle [d]	-90 to +90	-8.07
v_{inf} out-plane angle [d]	-90 to +90	44.79

The optimization is accomplished through the MATLAB GA, with a population of 1000 individuals; the optimal trajectory, according to the stop criterion already introduced, is obtained in 5 minutes with 100 computational nodes.

As can be seen from the output trajectory reported in Figure 3.10, the launcher directly inserts the spacecraft in a quasi-ballistic orbit, as expected from theory: thrusters are exploited only on arrival at Nereus; slightly more than

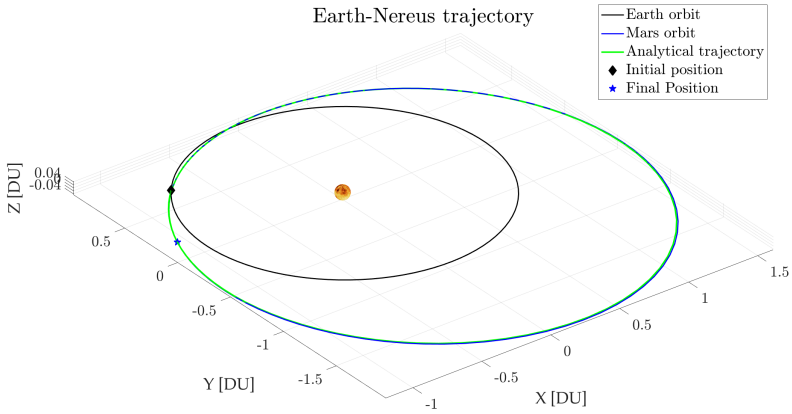


Figure 3.10: Earth-Nereus trajectory

10 mN suffice, and only 0.0052 fuel mass fraction is requested. This example shows that the proposed algorithm can manage also highly elliptical orbits in interplanetary trajectories: that is enabled by the the shape to be a non-linear interpolation between arrival and departure orbits.

3.2 Low thrust multi-injection approach for constellation and multi-mission deployment

With the advent of CubeSats and SmallSats, which range from 0.01 to 180 kg [60], the need for new techniques to launch in space these classes of satellites is arising. In the incoming years, a substantial increment in the number of small satellites to be launched has been forecasted [61]. Studies suggest that the number of satellites launched in the decade 2019-2028 will have a x4 growth rate compared to the previous decade and that satellites with a launch mass $< 500\ kg$ will account for 87% of such number. In addition to this, it was found that "despite a growing number of operational dedicated launch vehicles, the majority of nano/microsatellites in 2019 chose to leverage rideshare alternatives" [62]. The drawback of piggyback launches is in the fact that usually the small satellites are released on the target orbit of the main payload or in its proximity. Consequently, these satellites would need their propulsive system to be capable to allocate themselves on the correct orbit and with the desired phasing. Also, they would need to wait for a launch whose main payload has a target orbit as similar as possible to their final one. Being such satellites the largest market share, new ways to facilitate their access to space are being investigated. A possible solution to overcome the drawbacks of piggyback payload launches is to develop the last stages of launchers or dedicated vehicles able to carry the small satellites directly on their operational orbit. The objective of the work is to develop an algorithm that, given a set of N satellites to release in different positions around Earth with a vehicle with a a low-thrust control authority, is capable to define the optimal releasing order and transfer strategy. In Section 3.2.1 the algorithm developed to solve the VRP and the transfer strategy selected are presented, while in Section 3.2.2, the optimization approach adopted to solve the problem is reported. The most relevant results are discussed in Section 3.2.3.

3.2.1 Multi-deployment algorithm

First, in Section 3.2.1.1, the main structure of the routing solving algorithm is presented. Even though it is here used to find the releasing order of the multi-deployment mission, this routing algorithm is suitable to any routing problem. Afterward, in Section 3.2.1.2, the transfer strategy selected for the multi-deployment scenario is described.

3.2.1.1 Routing

Routing architecture The main workflow of the algorithm is shown in Fig. 3.11.

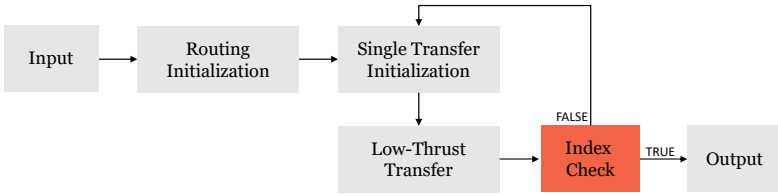


Figure 3.11: Routing algorithm main workflow.

Input. The main inputs of the algorithm are the initial wet mass of the vehicle M_0, the initial time JD_0, the specific impulse I_{sp}, the initial state KP_0 and the two matrices **K** and **P**. The former is a matrix of dimensions $6 \times N$, where N is the number of orbits onto which to deploy the satellites, or generically the number of destinations of the routing problem. Each column of the matrix contains the KP of each orbit. Matrix **P** is presented in the next item.

Routing Initialization. In this block, the releasing order is defined. Given N orbits onto which to release the satellites, the optimal path will be one of the possible permutations of the vector $[1, 2, ..., N]$, where each number identifies one of the columns of **K**. The matrix **P** is built outside the algorithm through the operator in Eq. (3.39), where $x = [1, 2, 3, .., N-1, N]$. The operator builds the matrix **P** such to contain all the possible permutations of x and its size will be $N \times N!$.

$$\mathbf{P} = \texttt{perms}(x) \tag{3.39}$$

Once the matrix **P** is built, only one discrete variable identifying one of the columns of **P** is enough to define the releasing order. The selected column will be referred to as p. Arithmetic overflow might occur when building the matrix **P**. Such issue and more details about the choice and advantages of introducing **P** to solve the VRP are discussed in Section 3.2.1.1.

Single Transfer Initialization. Once the releasing order is fixed by selecting a column of **P**, the first of the N transfers must be initialized. In this block, the initial mass, current date and target state position are updated. The latter, due to the presence of environmental perturbations, depends on the epoch. In particular, only the J_2 secular contribution of the Earth's gravitational assimetry is taken into account. The target state Right Ascension of the Ascending Node (RAAN) is updated according to Eq. (3.40).

$$\dot{\Omega}_{sec} = -\frac{3nR_{\oplus}^2 J_2}{2p^2} cos(i) \tag{3.40}$$

Low-Thrust Transfer. In this block, the transfer between the current state and the next one in the releasing order is computed.

Index Check. In this block, a check about the progress of the releasing mission takes place. If the last released satellite does not correspond with the last element of p, the next transfer is initialized. Otherwise, the mission is completed and the final output can be computed.

Output Once the last element of p has been reached, the total propellant consumption and duration of the transfer can be computed. In particular, the total values are the sum of the partial contributions of each transfer.

VRP solution As aforementioned, the problem belongs to the MINLP family, presenting both discrete and continuous variables. The latter are the ones optimizing the transfers, while the discrete variables are needed to define the route of the releasing vehicle. Since the multi-deployment algorithm must run inside an optimizer in order to find the optimal or near-optimal solutions, it was necessary to find a way to help the optimizer to efficiently evaluate different releasing orders. To speed up the algorithm and guarantee convergence onto a feasible solution, the introduction of the matrix $\mathbf{P}$ was considered. This solution resulted to be particularly efficient (see Section 3.2.1.1) since thanks to this choice only one discrete variable identifying the column of $\mathbf{P}$ is sufficient to define the visiting order of the orbits, allowing to drastically reduce the search space. This variable will be the only discrete variable of the optimization and shall adopt values from 1 to N!, which is the number of columns of matrix $\mathbf{P}$.

On the one hand, the introduction of $\mathbf{P}$ speeds up the convergence of the algorithm with respect to considering N discrete variables to define the visiting order. On the other hand, the building of $\mathbf{P}$ can require large storage space with increasing values of N. For these reasons the matrix $\mathbf{P}$ is built only once outside the multi-deployment algorithm and this approach proves ineffective for values of N larger than 11 (e.g. the storage of matrix $\mathbf{P}$ for $N = 12$ would require 42.8 GB of space). To overcome this limitation, a hybrid optimization approach between a branch and bound method and a heuristic method was implemented, explained in Section 3.2.2.

Routing validation It was interesting to compare the results obtained applying the routing algorithm to a problem whose solution was known in order to validate it and also to assess the algorithm quality and efficiency. The algorithm validation was carried on by solving the problem faced in the 5^{th} Global Trajectories Optimization Competition (GTOC), that was already used for validation in several papers [30] [63]. The scenario of this problem is different from the multi-deployment mission which is the main focus of the algorithm developed in this thesis. However, the two problems share similar features and therefore by applying only really little changes to the algorithm it was possible to apply it to this different problem. The problem deals with three

versions of a multiple asteroids rendezvous task, respectively with 4, 8 and 16 targets. The optimization approach explained in Section 3.2.2.1 to enlarge the capabilities of the algorithm only suits multi-objective optimization (due to the branching and bounding criteria chosen). For this reason, the algorithm will only be tested on the 4 and 8 targets cases. The details of the GTOC problem are reported in the work by Zhang et al [30] and are not here reported for the sake of brevity.

The workflow of the algorithm stays the one of the routing algorithm represented in Fig. 3.11, but the transfers will be computed as a single Lambert transfers. PSO was chosen as computational method for the single-objective optimization to minimize the total Δv of the hopping journey. In particular, the `particleswarm` function of MATLAB Global Optimization Toolbox [64] was used. The default tuning parameters of the algorithm were adopted in this case; a research of the optimal ones might further increase the performances reported in the next paragraphs.

For both the 4 and the 8 targets cases, respectively identified as Case 1 and Case 2, the problem was solved 10 times and the results are shown in Table 3.6. The results of the routing algorithm (M1) here proposed are compared to the ones found by solving it through two different methods (M2 and M3). The three methods are summarized in Table 3.5.

M1	Routing algorithm
M2	GA with search enhancement
M3	Two Phase Algorithm (TPA)

Table 3.5: Methods for routing algorithm validation.

The results are compared to the ones found by Zhang et al [30], who tried different computational methods to solve the problem. Only the method which provided the best results is here reported, which are the ones found by performing the minimization through the adoption of a GA with search enhancement (M2). The proposed method will be compared also to the one proposed by Bang and Ahn [63] which consists in a TPA (M3) characterized by a first phase in which some elementary solutions are found which are later used as starting point to solve the TSP.

The results are shown in Table 3.6, which confirms the validity of the algorithm and also proves its quality in performances. Only the information about the best results found by adopting M3 was available.

It is interesting to compare not only the results but also the rapidity with which the algorithm finds its final solution. The computation procedure presented

Method	Case 1 [km/s]			Case 2 [km/s]		
	Best	Mean	STD	Best	Mean	STD
M1	6.068	7.558	0.779	17.350	20.989	1.921
M2	6.397	7.307	0.570	19.153	22.978	3.044
M3	6.360	-	-	16.400	-	-

Table 3.6: Results comparison for algorithm validation.

in this report ran on a personal computer with an Intel(R) Core(TM) i7-7500 (2.7 GHz) processor and a 16 GB RAM. However, the rapidity of the codes is measured through the evaluation of how many times the Lambert functions are called to find the final solution and therefore independently on the computing machine. For the proposed method M1 the averages of the 10 runs are reported in Table 3.7 and compared to the number of Lambert calls of the other methods [63]. The results indicate that the computational resource spent by the proposed method is about one order of magnitude smaller than adopting M2 and two than the M3, even though the latter was capable to find a better solution for Case 2.

Method	Number of Lambert routine calls	
	Case 1	Case 2
M1	88,000	348,320
M2	960,000	7,680,000
M3	3,526,933	25,392,677

Table 3.7: Number of Lambert routine calls comparison.

All things considered, it is possible to deduce that the algorithm works and finds reliable solutions to the problem. Also, it is competitive with respect to similar algorithms in terms of solutions found and especially of computational cost.

3.2.1.2 Low-thrust transfer

Single transfer The 3-dimensional shape-based algorithm described in Sec. 3.1 [31] was selected for the single transfers since particularly suited to planetocentric mission scenarios. The shape of the transfer is proposed a priori as a non-linear interpolation of similar and consecutive orbits. A completely analytical shape based approach was necessary due to its really low computational complexity, which makes it possible to evaluate several different

trajectories in few seconds. This is of paramount importance for the scope of the multi-deployment algorithm since a really large number of transfers have to be computed as fast as possible. When dealing with orbits in the LEO region, environmental perturbations have an important role, especially in low-thrust since the order of magnitude of the perturbing accelerations are sometimes the same as (if not higher than) the thrust acceleration [65]. For this reason, the impact of the perturbations on the spacecraft trajectories, especially the J_2 effect which is the main contribution, must be taken into account when planning the transfers, since it cannot be simply counteracted and canceled by the spacecraft. The shape-based algorithm is then modified to take into account the J_2 perturbation disturbances. The most accurate way to consider this perturbation would be considering its punctual effect on each revolution around the main attractor. However, this approach would need the integration of the equations of motion which would dramatically increase the computational load. For this reason, only the integral of the effects of the J_2 perturbation are considered in the trajectory computation. After each revolution around the Earth the current state is corrected with the secular effect of the perturbation over that revolution. While correcting the current state with the secular effect of the perturbation introduces some discontinuities in the trajectory and control low, it provides a good estimation of the propellant consumption and transfer duration with a really low computational load, which is the main driver of the algorithm development.

Transfer strategy To perform the transfers in the most efficient way, a transfer strategy exploiting the secular effect of the J2 perturbation was planned and is here explained. The main idea behind this approach is to change the RAAN of the spacecraft not by thrusting the spacecraft but only exploiting the gravitational perturbation due to the not spherically symmetric mass distribution of the central attractor [66]. The RAAN of a spacecraft on a given orbit can be changed for free by waiting the necessary amount of time without counteracting the J2 perturbation. By doing so, the rate of change of the RAAN would be the one in Eq. (3.40). Alternatively, the spacecraft can also move to another orbit in order to make the desired change of RAAN happen faster at a cost of a little propellant consumption. This two-legs transfer option turns the transfer problem into an optimization problem whose variables are the KP of the intermediate orbit onto which to stationary for the change of RAAN. The workflow of the transfer strategy is reported in Fig. 3.12 and structured in the blocks described below:

Inputs. The starting orbit KP_1 and the target orbit KP_2 are the two main inputs to the function.

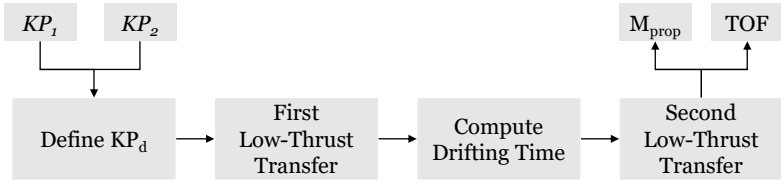


Figure 3.12: Single low-thrust J2-exploiting transfer scheme.

Drifting Orbit Definition. The drifting orbit is defined by the six KP reported in Table 3.8. As it can be seen from Eq. (3.40), the three parameters

a_d	e_d	i_d	Ω_d	ω_d	θ_d
var	0	*var*	*free*	$\frac{1}{2}(\omega_1 + \omega_2)$	*free*

Table 3.8: KP of the drifting orbit for low-thrust J2-exploiting transfer strategy.

which affect the RAAN variation are the semi-major axis a, the eccentricity e and the inclination i. For this reason, not all the KP are considered as variables in order to reduce the size of the optimization problem:

- The semi-major axis is the one which has the greatest impact on the RAAN variation and it is also an element whose value is relatively cheap to control and change, making it the main variable of the optimization.
- Eccentricity affects the RAAN change since a high value means an orbit perigee at a lower altitude and therefore more subject to the gravitational asymmetries of the central planet. However, it is preferred to have a uniform influence of such asymmetry on the spacecraft over the revolutions and therefore this value is chosen to be 0.
- The inclination is a parameter that also greatly affects the RAAN variation but also extremely expensive to change. Therefore, it is considered as variable but only when a change of inclination is necessary from original to target orbit. In that case, the value of the inclination of the drifting orbit is selected between i_1 and i_2. In the case in which i_1 and i_2 have the same value, the inclination is kept the same as the ones of the two orbits unless polar orbits are dealt with. Polar orbits do not experience the RAAN secular variation and, for this reason, it is necessary to depart from 90° inclination to shift the orbital plane.
- Saying that Ω_d is left *free* means that no control is considered on the final value of the RAAN when moving to the drifting orbit. The RAAN of the spacecraft is left to vary according to the J_2 effect. There would

be no point or convenience in controlling the RAAN since the goal itself of reaching the drifting orbit is to change such parameter by exploiting the natural perturbations instead of the propellant.

- The argument of perigee ω of an orbit does not affect the secular variation of the RAAN. For this reason, ω_d was not taken as variable of the problem and was arbitrarily set to have a halfway value between ω_1 and ω_2.
- θ_d is *left* free since it affects neither the secular RAAN variation nor the cost or duration of the multi-revolution low-thrust transfer.

First Transfer. Once the drifting orbit is defined, the cost and duration of the first low-thrust transfer are computed.

Drifting Time Computation. Once the target orbit has been reached, it is necessary to compute the amount of time necessary to close the RAAN gap. Generally speaking, knowing the elements of the drifting orbit KP_d and the ones of the target orbit KP_2 it is possible to compute the RAAN gap (Eq. (3.41a)) at the moment of the arrival on the drifting orbit and the relative drift rate (Eq. (3.41b)). Once these two quantities are known, it is possible to compute the waiting time (Eq. (3.41c)) necessary to have $\Omega_2 = \Omega_d$, at a value different from the two original ones.

$$\Delta\Omega = \Omega_2 - \Omega_d \tag{3.41a}$$

$$\Delta\dot{\Omega} = \dot{\Omega}_d - \dot{\Omega}_2 \tag{3.41b}$$

$$t_{wait} = \frac{\Delta\Omega}{\Delta\dot{\Omega}} \tag{3.41c}$$

While this is true for impulsive maneuvers, when dealing with low-thrust transfers the computation of the exact Ω_d at which to depart from the drifting orbit is slightly more complicated. Since low-thrust transfers between the orbits have large times of flight, an amount of relative RAAN shift happens also during the transfer from the drifting to the target orbit. For this reason, departing when the two values of the RAAN are already equal would require some degree of control on such parameter during the transfer to the target orbit to keep such values equal. The most efficient and less expensive way to perform the transfer would be to estimate the relative drift which takes place during the transfer from the drifting to the target orbit and to take it into account when computing the drifting time. To adopt this approach the following steps must be taken:

- A fictitious target orbit is defined with the same parameters as the true target orbit KP_2 apart from the RAAN which is left free, to estimate the amount of shift that would take place during the transfer. Such shift is referred to as $\Delta\Omega_{t,1}$.

- Once computed the transfer to the fictitious target orbit, knowing the TOF of the transfer it is possible to estimate also how much the target orbit shifts during the duration of the transfer. This amount of RAAN shift is defined as $\Delta\Omega_{t,2}$.
- From these two quantities it is possible to compute the amount of relative shift $\Delta\Omega_t$ (Eq. (3.42a)) and use it to correct the RAAN gap to close (Eq. (3.42b)). The time necessary is then computed according to Eq. (3.41c) and with Eq. (3.40) the new values of the RAAN at the end of the waiting time are updated.

$$\Delta\Omega_t = \Delta\Omega_{t,1} - \Delta\Omega_{t,2} \tag{3.42a}$$

$$\Delta\Omega = \Omega_2 - \Omega_d + \Delta\Omega_t \tag{3.42b}$$

Shortly, this correction on $\Delta\Omega$ allows finding the correct Ω_d at which to depart from the drifting orbit, which does not coincide with Ω_2 unless the two orbits are so close or so high in altitude that the J2 effect during the transfer can be considered negligible. Thanks to this estimation, no propellant has to be consumed to control the RAAN because the transfer is planned in such a way to meet the target orbit at the correct value of this parameter.

Second Transfer. Once the RAAN of current and target orbits have been updated after the stationing onto the drifting orbit, the second low-thrust transfer to the target orbit can effectively be computed.

Outputs. At the end, the whole cost of the two-legs transfer is computed. In terms of propellant, the total amount needed to reach the target orbit is the sum of the propellant consumed in the two legs of the transfer. In terms of time, the total duration of the transfer is the sum of the TOFs of the two legs and also the waiting time onto the drifting orbit.

3.2.2 Optimization

First, the optimization approach adopted is presented in Section 3.2.2.1. Later, more details about how some solutions are discarded and about the introduction of constraints are discussed in Section 3.2.2.2 and Section 3.2.2.3.

3.2.2.1 Optimization approach

A multi-objective optimizations was preferred to minimize the propellant consumption and time duration of the mission. A heuristic optimization method was necessary due to the large search space of the problem. Heuristic methods [67] do not grant to find the optimal solutions to the problem but the low computational effort required to achieve this near-optimal solution makes them very valuable. In particular, population-based methods are more suitable

to find promising areas in a large search space [68]. For these reasons, the MOPSO [69] method, specially modified to work with discrete variables, will be used to perform all the multi-objective optimizations. The two main tuning parameters of the method are N_p and M_{gen}.

However, while the optimization method selected was the MOPSO, a particular approach to deal with the optimization of the problem was developed and is presented in this section. This approach was at first developed to deal with problem with N$>$ 11, which was a problem after the introduction of $\mathbf{P}$. However, it proved to provide better results even when applied to cases of smaller dimension (see Section 3.2.3.1).

Instead of dealing with the whole hopping trajectory at once, the problem is broken up into smaller and consequential subproblems. A parameter r which defines the dimension of the subproblem is chosen, defining also the number of iterations necessary to solve the problem of reaching the N orbits. The number of iterations N_{iter} needed to solve the problem can be computed by Eq. (3.43), where the operator rounds the result of the fraction to the nearest integer greater than or equal to it. In the case in which the remainder after the division of N by r is different from zero, the last iteration is performed with a subset of size r equal to the remainder.

$$N_{iter} = \texttt{ceil}\left(\frac{N}{r}\right) \tag{3.43}$$

When adopting this hybrid approach, the matrix $\mathbf{P}$ to give as input to the routing algorithm is built through the use of a different operator than Eq. (3.39). The new operator is shown in Eq. (3.44) and builds a matrix which, given a set of N orbits to reach, only contains the permutations of a subset of r elements of the vector x.

$$\mathbf{P} = \texttt{subperms}(x, r) \tag{3.44}$$

The number of possible permutations N_{perms}, in this case, is not anymore N! but can be found through Eq. (3.45). The size of the matrix $\mathbf{P}$ will be $r \times N_{perms}$.

$$N_{perms} = \frac{N!}{(N-r)!} \tag{3.45}$$

The pseudo code of the optimization approach is presented in Algorithm 1, whose steps are described below, and is represented by the scheme in Fig. 3.13. In the latter, each circle represents a single solution. The grey solutions actually represent several solutions, whose number is unknown a priori due to the heuristic nature of the optimization.

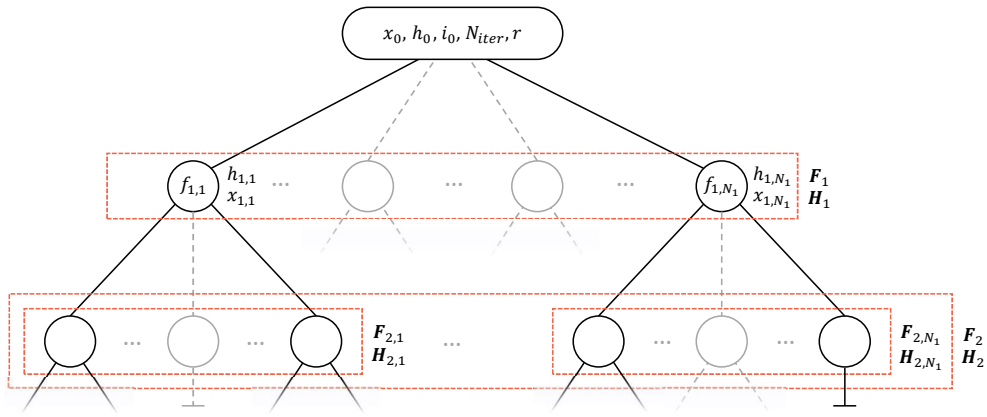


Figure 3.13: Branch and bound based heuristic approach scheme.

Algorithm 1: Branch and bound based heuristic approach

Data: x_0, i_0, h_0, N_{iter}, r
Result: $\mathbf{F}_{N_{iter}}$, $\mathbf{H}_{N_{iter}}$

begin
  $[\mathbf{F}_1, \mathbf{H}_1] = \texttt{branching}(x_0, i_0, h_0, r)$
  for $j = 2$ **to** N_{iter} **do**
    $N_{j-1} = \texttt{length}(\mathbf{H}_{j-1}(:, 1))$
    for $k = 1$ **to** N_{j-1} **do**
      $h_{j-1,k} = \mathbf{H}_{j-1}(k, :))$
      $x_{j-1,k} = \texttt{exclude}(x_0, h_{j-1,k})$
      $i_{j-1,k} = \mathbf{F}_{j-1}(k, :))$
      $[\mathbf{F}_{j,k}, \mathbf{H}_{j,k}] = \texttt{branching}(x_{j-1,k}, i_{j-1,k}, h_{j-1,k}, r)$
    end
    $\mathbf{H}_j = [\mathbf{H}_{j,1}; \mathbf{H}_{j,2}; \ldots ; \mathbf{H}_{j,N_{j-1}}]$
    $\mathbf{F}_j = [\mathbf{F}_{j,1}; \mathbf{F}_{j,2}; \ldots ; \mathbf{F}_{j,N_{j-1}}]$
    $[\mathbf{F}_j, \mathbf{H}_j] = \texttt{bounding}(\mathbf{F}_j, \mathbf{H}_j)$
  end
  $[\mathbf{F}_j, \mathbf{H}_j] = \texttt{boundingpareto}(\mathbf{F}_j, \mathbf{H}_j)$
end

- The inputs to the algorithm are the vector of all the elements still to be reached x, the vector of initial conditions i, the vector containing the indices of the destinations already reached h. At the first iteration $x_0 = [1, 2, .., N]$ and h_0 is initialized as an empty vector. In addition to these, the number of iterations N_{iter} computed in Eq. (3.43) and the sub-search dimension r are also given as input to the algorithm.

- The first iteration is performed outside the **for** loop. The operator `branching`, starting from one initial condition, runs the heuristic optimizer and finds a set of partial solutions which make the first branches of the algorithm. The matrix $\mathbf{P}$ at this iteration is computed through Eq. (3.44), with x_0 as input. The heuristic algorithm only gives as output the solution which respect the bounding criteria, explained in Section 3.2.2.2. Each solution is characterised by $f_{j,1}$, $h_{j,1}$ and $x_{j,1}$ vectors. The survived solutions are stored in two matrices:
  - $\mathbf{F}_j = [f_{j,1};\ f_{j,2};\ ...\ ;\ f_{j,N_j}]$ is the matrix containing the final conditions of each solution, including the values of the objectives.
  - Each row of $\mathbf{H}_j = [h_{j,1};\ h_{j,2};\ ...\ ;\ h_{j,N_j}]$ contains the indices of the elements reached by the respective partial solution.
- Now the first **for** loop begins. At each iteration, N_{j-1} is computed, which is the number of the branches coming from the previous iteration. The operator `length` computes the number of elements of the vector of input. Each of the branches represents a solution that must be expanded in the following **for** loop.
- One branch at a time, the iteration is initialized defining $h_{j-1,k}$, $i_{j-1,k}$ and $x_{j-1,k}$. The latter in particular is defined by the operator `exclude`. This operator cancels from x_0 all the elements already reached by that partial solution, identified by $h_{j-1,k}$. The new vector $x_{j-1,k}$ will have dimension $N - r \cdot j$.
- From each branch, a new set of branches is found again through the operator `branching`. At each iteration a different $\mathbf{P}$ is given as input to the routing algorithm, again computed through Eq. (3.44), each time with $x_{j-1,k}$ as input. The branches are stored in $\mathbf{F}_{j,k}$ and $\mathbf{H}_{j,k}$.
- Once all the branches have been expanded, the solutions are stored in the matrices $\mathbf{F}_j$ and $\mathbf{H}_j$, which include all the branches born from the current iteration.
- Before starting the next iteration, the bounding criteria must be applied to $\mathbf{F}_j$ and $\mathbf{H}_j$. While it is true that each $\mathbf{F}_{j,k}$ is composed by solutions which survived the bounding inside the single optimization, now they must be compared to the all the other set of solutions of the iteration. The bounding criteria are applied by the operator `bounding` and the new $\mathbf{F}_j$ and $\mathbf{H}_j$, containing only the survived solutions, are given as output. In Fig. 3.13 the solutions which do not survive the bounding criteria are indicated with the symbol $\perp$.
- The solutions of the iteration $j-1$ which survived the `bounding` operator, are now expanded themselves.

- After the last iteration, if necessary, a more stringent bounding criterion is applied to the final set of solutions. In particular, through the use of the operator `boundingpareto` only the non-dominated solutions are kept.

3.2.2.2 Bounding criteria

The operator `bounding`, given a set of solutions and some bounding criteria, only keeps the solutions that respect the latter while discarding all the others. Choosing to discard all the solutions not belonging to the Pareto front of the sub-problem would lead to the risk of discarding some partial solutions really close to the Pareto front, which might eventually become optimal solutions when extending them. Therefore, a region of solutions must be selected: all the solutions belonging to the Pareto front or within a certain percentage range from one of the solutions in the front can be kept and then extended. These solutions outside the Pareto front but which survive the bounding criterion are referred to as "partially dominated solutions", even though they are actually fully dominated according to the definition of dominated solutions. The value r_{perc}, chosen between 0 and 1, sets the percentage range within which a dominated solution is considered to be only partially dominated. A schematic representation of such solutions is reported in Fig. 3.14. Indicating

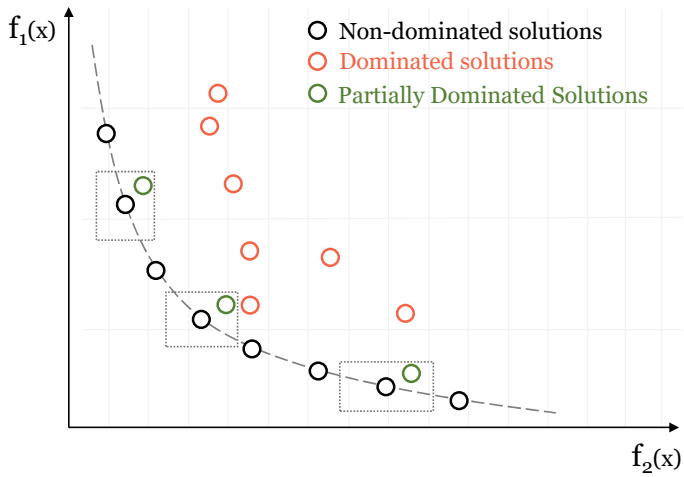


Figure 3.14: Pareto front and partially dominated solutions.

with $f_{1,D}$ and $f_{2,D}$ the objectives of a dominated solutions and with $f_{1,P}$ and $f_{2,P}$ the ones of the closest non-dominated solution, a dominated solutions is considered to be partially dominated if both the criteria in Eq. (3.46) are met.

$$\frac{f_{1,D} - f_{1,P}}{f_{1,P}} < r_{perc} \tag{3.46a}$$

$$\frac{f_{2,D} - f_{2,P}}{f_{2,P}} < r_{perc} \tag{3.46b}$$

Graphically, each Pareto solution has a rectangle with sides of length $r_{perc} \cdot f_{1,P}$ and $r_{perc} \cdot f_{2,P}$ which defines the range into which a dominated solution is considered to be partially dominated.

3.2.2.3 Constraints

When dealing with engineering problems, it is of paramount importance that an algorithm allows the introduction of constraints to some of the variables. In the routing algorithm, the constraints were introduced by the use of penalty functions.

Penalty functions allow treating of constrained problems as unconstrained problems, introducing an artificial penalty when the constraint is violated, but may create severe slope changes or discontinuities in the solution space, which could interfere with a heuristic optimization algorithm. To help the latter converge onto feasible solutions, it was chosen to introduce a penalty function whose penalty is proportional to the amount of violation of the constraint. For instance, considering a generic objective f_1 and an upper limit L_1, the objective function is modified adding the quadratic loss function $\lambda(f_1, L_1)$ in Eq. (3.47).

$$\lambda(f_1, L_1) = \max(0, f_1 - L_1)^2 \tag{3.47}$$

The new formulation of the objective is presented in Eq. (3.48),

$$f_1 = f_1 + F \cdot \lambda(f_1, L_1) \tag{3.48}$$

where F is the penalty factor, a scalar greater than 0 and arbitrarily chosen depending on the order of magnitude of f_1.

3.2.3 Results and Discussion

In Section 3.2.3.1 an example of multi-deployment is reported, solved with both a traditional pure heuristic approach and the branch and bound based heuristic one. Afterward, in Section 3.2.3.2, a case of deployment of an existing constellation is dealt with. Due to the relatively high mass of the satellites belonging to existing constellations around Earth, the deployment of only one portion of the constellation is considered.

3.2.3.1 Heuristic-hybrid approaches comparison

A generic set of 6 satellites to deploy in LEO was considered. The KP of the orbits into which to insert each of them are reported in Table 3.9, together

ID	a [DU]	e [-]	i [deg]	Ω [deg]	ω [deg]
0	1.05	0.02	66	0	0
1	1.06	0.01	67	10	5
2	1.07	0.02	66	8	0
3	1.08	0.01	67	328	3
4	1.09	0.03	66	161	0
5	1.10	0	68	22	0
6	1.11	0.05	66	159	20

Table 3.9: Initial states for multi-deployment mission example.

with the starting orbit, indicated by the index 0, from which the vehicle starts the journey. DU is equal to the radius of the Earth, which was used to non-dimensionalize the semi-major axes. All the KP reported are the ones at the time of the departure from orbit KP_0. The lower and upper bounds of the optimization for this example are the ones in Table 3.10. The inclinations of the drifting orbits are allowed to vary between the minimum and maximum value of the inclinations of the sets of orbits into which to release the satellites.

Parameter	lb	ub
a_d [DU]	1.0314	1.300
i_d [deg]	66	68

Table 3.10: Optimization tuning parameters for multi-deployment mission example.

The characteristics of the deploying vehicle chosen for this example are presented in Table 3.11. All the 6 satellites to be released were considered to be nano-satellites of 5 kg mass each.

M_0 [kg]	T [N]	I_{sp} [s]
100	0.5	3500

Table 3.11: Spacecraft characteristics for multi-deployment mission example.

First, the problem was solved with a pure heurstic approach addressing the $N = 6$ problem directly. $N_p = 20$ and $M_{gen} = 20$ were chosen for the only iteration of the optimization. Afterward, the hybrid approach was carried on with $r = 3$, meaning that two iterations were needed to find the final solutions. Two different values of N_p and M_{gen} were chosen for the first and second iteration, reported in Table 3.12. This was done since the second iteration has

	Iter 1	Iter 2
N_p	10	5
M_{gen}	20	10

Table 3.12: Optimization tuning parameters of hybrid approach for optimization approaches comparison.

a solutions space smaller than the first iteration. While the first iteration has 120 possible permutations according to Eq. (3.45), the second iteration only has 6, since the matrix $\mathbf{P}$ in this case only has N! possible columns (Eq. (3.39)), with this time $N = r = 3$.

Since the final result of a heuristic optimization method strongly depends on the initial solution, comparing only one optimization run to compare the results would not be really significant. For this reason, in order to truly assess whether or not one approach is better than the other, more than one run per each must be considered. For both of them, 10 runs were performed and the 10 Pareto fronts have been merged into one, where only the non-dominated solutions survived. The two Pareto fronts are shown in Fig. 3.15, while the

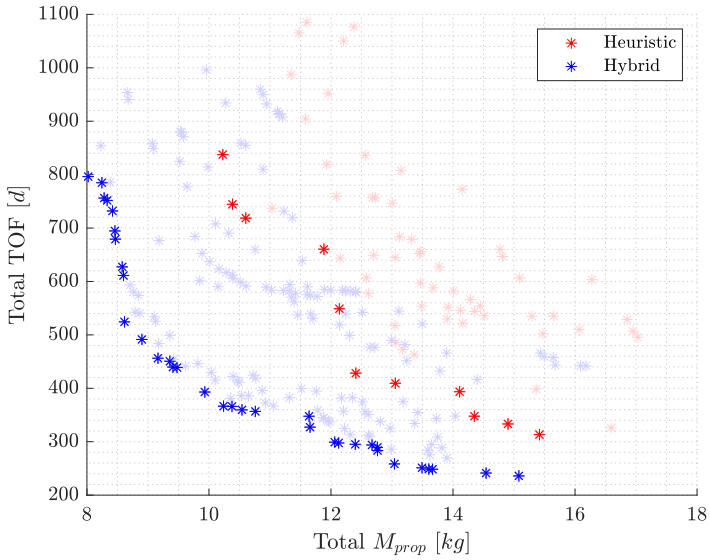


Figure 3.15: Comparison of optimization approaches results ($M_0 = 100\ kg$; $I_{sp} = 3500\ s$).

transparent markers represent the clouds of solutions found by the 10 runs of the optimization for both the approaches. It is clear how the Pareto front found

from the hybrid optimization approach is composed by far better solutions with respect to the pure heuristic one. Indeed, the blue Pareto front dominates the red one in all its solutions found.

It is important when comparing the two optimization approaches to compare not only the final results but also the computational effort necessary to achieve them. The tuning parameters N_p and M_{gen} were chosen to have a similar duration of the runs between the two approaches. The time is indeed proportional to the function evaluations necessary to find the solution. The average duration of the 10 runs of the hybrid approach was 225.5 s, while the heuristic one needed an average of 282.3 s per run. It is possible to conclude that the hybrid approach outperforms the pure heuristic one both in terms of results and computing effort. In light of such results, the hybrid optimization approach will be chosen as the standard to perform the minimization of the objectives, also when dealing with problems characterized by $N < 11$.

3.2.3.2 Constellation insertion

Here, a possible real application is dealt with: the deployment of a portion of the satellites belonging to the Starlink [70] constellation was considered. The satellites of this constellation have a mass of 260 kg each. The Starlink spacecraft constellation will be spread into 24 orbital planes with an inclination of 53°, on circular orbits with an altitude of 550 km. A possible case may be the replacement of 6 satellites of the constellation, each on a different orbital plane. Supposing that the multi-deployment vehicle is already released on one of the orbital planes of the constellation (the one referred to with ID 0), the planes onto which to release the satellites are reported in Table 3.13.

ID	a [DU]	e [-]	i [deg]	Ω [deg]
0	1.0862	0	53	0
1	1.0862	0	53	15
2	1.0862	0	53	30
3	1.0862	0	53	45
4	1.0862	0	53	60
5	1.0862	0	53	90
6	1.0862	0	53	105

Table 3.13: Initial states for Starlink replacement mission.

The initial wet mass M_0 of the releasing vehicle was considered to be 2000 kg, according to mass estimating statistical relationships [71]. The vehicle is considered to be provided with ten RIT 2X Series [72] thrusters, each with a nominal thrust of 171 mN and constant specific impulse of 3500 s.

The optimization of the multi-deployment mission was carried on through the use of the branch and bound based heuristic approach. With the size of the problem $N = 6$ and the size of the subsearch $r = 3$, two iterations were needed to find the final solutions. The tuning parameters of the optimization are reported in Table 3.14, while the lower and upper bounds of the research are shown in Table 3.15.

	Iter 1	Iter 2
N_p	15	10
M_{gen}	30	20

Table 3.14: Optimization tuning parameters for Starlink replacement mission.

Parameter	lb	ub
a_d [DU]	1.0314	1.250
i_d [deg]	53	53

Table 3.15: Lower and upper bounds for Starlink replacement mission.

The results of one run of the algorithm are presented in the top Pareto front in Fig. 3.16. A constraint on a maximum mission time of 2 years was imposed, reason why all the results do not exceed 700 d of duration. The bottom subfigure in Fig. 3.16 represents a focus on the fastest solutions of the complete Pareto front of the top one.

It can be interesting to focus on one solutions to go deeper in detail on the releasing strategy. For instance, the fastest solution is considered. As it could be expected from Table 3.13, the fastest solutions are characterized by the releasing order $p = [1, 2, 3, 4, 5, 6]$. The releasing details of this solution are reported in Table 3.16.

Some considerations arise by analyzing the results:

- In four out of the six transfers, the value of a_d was set equal to the upper bound of the solutions space. This suggests that faster solutions, if desired, can be achieved by setting a higher upper bound.
- It can be noticed how the propellant mass required for the last transfers is smaller in magnitude than the one of the initial transfers. This is due to the lightening of the vehicle, whose main contribution is given by the release of the satellites rather than by the propellant consumption. Together with the propellant mass, also the times of flight of the transfers

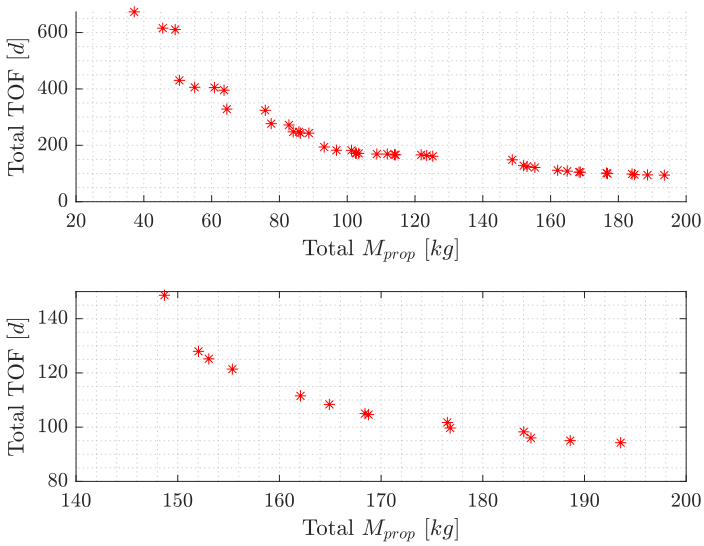


Figure 3.16: Top: Pareto front for Starlink replacement mission. **Bottom**: focus on fastest solutions ($M_0 = 2000\ kg$; $I_{sp} = 3500\ s$).

	R1	R2	R3	R4	R5	R6
a_d [DU]	1.2136	1.2500	1.1898	1.2500	1.2500	1.2500
i_d $[deg]$	53.00	53.00	53.00	53.00	53.00	53.00
$M_{prop,1}$ $[kg]$	23.44	24.91	13.40	16.14	11.86	7.68
TOF_1 $[d]$	7.1962	7.7478	4.2896	5.0999	3.7778	2.5871
t_{wait} $[d]$	2.8250	0.3743	7.7710	3.1846	13.1871	5.8489
$M_{prop,2}$ $[kg]$	23.16	24.54	13.27	15.92	11.67	7.57
TOF_2 $[d]$	7.1659	7.6668	4.2363	5.0745	3.7877	2.4532

Table 3.16: Release details of the fastest solution for Starlink replacement mission.

decrease due to the higher acceleration peaks that can be achieved, being the maximum available thrust constant.

- Finally, it is possible to notice how the waiting times t_{wait} on the drifting orbits grow bigger with the going on of the mission. Due to the smaller times of flight, a smaller portion of the RAAN gap is covered during the two legs of the transfers. For this reason, more time must be spent on the drifting orbit to obtain the desired RAAN shift.

Due to the really high initial mass of the vehicle, an extreme case with 10 thrusters simultaneously firing was considered to get the previous results. It is interesting to study the behaviour of the solutions when changing the number of thrusters the vehicle is provided with. A run of the optimization with the same tuning parameters as before has been performed with 7, 5 and 2 thrusters. The resulting Pareto fronts are plotted in the same figure in Fig. 3.17, where

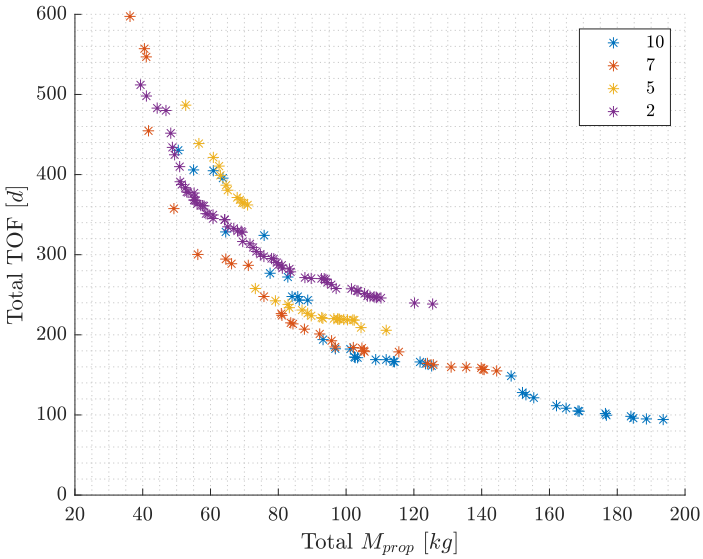


Figure 3.17: Pareto fronts with different number of thrusters for Starlink replacement mission ($M_0 = 2000\ kg$; $I_{sp} = 3500\ s$).

the numbers in the legend represent the number of thrusters that can be fired simultaneously. Respectively, the three configurations lead to a maximum thrust available of 1.1970 N, 0.8550 N and 0.3420 N. As expected, there is no difference in the slow solutions, even due to the maximum mission time duration set to 2 years. The solutions with low propellant consumption and high mission duration are characterized by the choice of a a_d really similar to the semi-major axis of the orbits of the constellation, solutions that can be carried out with each propulsive configuration. The real difference is in the fastest solutions, since the higher the maximum thrust available the faster an orbit with different a_d can be reached. Ten thrusters allow the deployment of the whole set of satellites in about 100 days, while two thrusters configuration requires at least about 250 days to release the six of them.

This comparison is important to understand how the thrust authority does not deeply impact the time duration of the missions, since the greatest contribution to it is given by the amount of time to wait on the drifting orbits to change the RAAN.

CHAPTER 4

Attitude Guidance

In this section a novel approach to the definition of the attitude guidance trough an innovative shape-based method is proposed. In particular the algorithm, that solve the exact EoM in an inverse formulation described below in this section,can be successfully adopted to generate attitude guidance of a wide range of maneuvers, even in presence of attitude or actuation practical constraints.

4.1 Analytical derivation of the equation of motion

The aim of this section is to write an analytical parametrization of the attitude motion for a rigid spacecraft, equipped with a certain number of momentum exchange devices. In this case, the angular momentum of the spacecraft can be written using a barycentric Body-Fixed (BF) RF as reported in Eq. 4.1

$$\boldsymbol{\Gamma} = \boldsymbol{\Gamma}_{SC} + \boldsymbol{\Gamma}_{RWs} = \boldsymbol{I}_{SC}\boldsymbol{\omega} + \boldsymbol{A}_{RWs}\boldsymbol{h}_{RWs} \tag{4.1}$$

In which $\boldsymbol{I}_{SC}$ is the inertia tensor of the rigid spacecraft, including the Reaction Wheels (RWs) at rest, $\boldsymbol{h}_r$ is the vector containing the scalar value of the angular

momentum of each inertia exchange device and $\boldsymbol{A}_{RWs}$ is the assembly matrix in which each column of the matrix contains the mounting direction of the corresponding momentum exchange device. The EoM of the spacecraft can be computed using the second cardinal equation of the dynamics as in Eq. 4.2.

$$\frac{d\boldsymbol{\Gamma}}{dt} + \boldsymbol{\omega} \times \boldsymbol{\Gamma} = \boldsymbol{M}_{pert} \tag{4.2}$$

which, considering a generic set of inertia exchange actuators (RWs, Control Moment Gyroscope (CGM) or Inertia Wheel (IW)) lead to the EoM reported in Eq. 4.3

$$\boldsymbol{I}_{SC}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times \boldsymbol{I}_{SC}\boldsymbol{\omega} + \dot{\boldsymbol{A}}_m\boldsymbol{h}_r + \boldsymbol{A}_{RWs}\dot{\boldsymbol{h}}_r + \boldsymbol{\omega} \times \boldsymbol{A}_{RWs}\boldsymbol{h}_r = \boldsymbol{M}_{pert} \tag{4.3}$$

If the analysis is restricted to the classical architecture consisting of RWs only, in which their directions of the angular momentum with respect to the BF RF is constant in time, the EoM can be rewritten as presented in Eq. 4.4, in which the dynamics of the control devices is grouped in the variable $\boldsymbol{M}_{control}$ for sake of simplicity.

$$\begin{cases} \boldsymbol{I}_{SC}\dot{\boldsymbol{\omega}} + \omega \times \boldsymbol{I}_{SC}\boldsymbol{\omega} = \boldsymbol{M}_{pert} + \boldsymbol{M}_{control} \\ \boldsymbol{M}_{control} = -\left(\boldsymbol{A}_{RWs}\dot{\boldsymbol{h}}_r + \boldsymbol{\omega} \times \boldsymbol{A}_{RWs}\boldsymbol{h}_r\right) \end{cases} \tag{4.4}$$

The system of attitude EoM for the rigid spacecraft equipped with RWs can be rewritten as in Eq. 4.5, in which $\boldsymbol{A}_{RWs}^{-1}$ is the Moore-Penrose Pseudo-inverse matrix of the RWs assembly matrix $\boldsymbol{A}_{RWs}$, to find out the dynamics of the actuators.

$$\dot{\boldsymbol{h}}_r = -\boldsymbol{A}_{RWs}^{-1}\left[-\left(\boldsymbol{I}_{SC}\dot{\boldsymbol{\omega}} + \omega \times \boldsymbol{I}_{SC}\boldsymbol{\omega} - \boldsymbol{M}_{pert}\right) + \boldsymbol{\omega} \times \boldsymbol{A}_{RWs}\boldsymbol{h}_r\right] \tag{4.5}$$

In particular, it is important point out that, while it is possible to have a complete analytical representation for the control torque $\boldsymbol{M}_{control}$, to find out the exact guidance for the RWs, it is necessary to numerically integrate the dynamical system as presented in Eq. 4.5.

In order to pursue the objective of creating an analytical representation of the dynamics of the satellite's attitude using a quaternion shaping approach, it is important to analytically derive the angular velocity $\boldsymbol{\omega}$ and the angular acceleration $\dot{\boldsymbol{\omega}}$ starting from the quaternion themselves.

In particular, the angular velocity is linked with the quaternions by the quaternions derivative law, as in Eq. 4.6, in which $\boldsymbol{\Omega}$ is defined as in Eq. 4.7

$$\frac{d\boldsymbol{q}}{dt} = \frac{1}{2}\boldsymbol{\Omega q} \tag{4.6}$$

$$\boldsymbol{\Omega} = \begin{bmatrix} 0 & \omega_z & -\omega_y & \omega_x \\ -\omega_z & 0 & \omega_x & \omega_y \\ \omega_y & -\omega_x & 0 & \omega_z \\ -\omega_x & -\omega_y & -\omega_z & 0 \end{bmatrix} \tag{4.7}$$

that can be rewritten in an explicit form as in Eq. 4.8.

$$\begin{cases} \dot{q}_1 = \frac{1}{2}\left(q_2\omega_z - q_3\omega_y + q_4\omega_x\right) \\ \dot{q}_2 = \frac{1}{2}\left(-q_1\omega_z + q_3\omega_x + q_4\omega_y\right) \\ \dot{q}_3 = \frac{1}{2}\left(q_1\omega_y - q_2\omega_x + q_4\omega_z\right) \\ \dot{q}_4 = \frac{1}{2}\left(-q_1\omega_x - q_2\omega_y - q_3\omega_z\right) \end{cases} \tag{4.8}$$

The system of equation presented in Equation 4.8 can be solved to find out the relation between the angular velocity of a rigid body and the values of its quaternions and quaternions time derivative, as reported in Eq. 4.9. It is important to point out that, since the system consist of four scalar equation among which only three are linearly independent, and there are three unknowns, it is possible to consider only three of the four equation to solve the system in the angular velocity unknowns, as reported in Eq. 4.9

$$\begin{cases} \omega_x = \frac{1}{q_4}\left(\dot{q}_1(q_1^2 + q_4^2) + \dot{q}_2(q_1q_2 + q_3q_4) + \dot{q}_3(q_1q_3 - q_2q_4)\right) \\ \omega_y = \frac{1}{q_4}\left(\dot{q}_1(q_1q_2 - q_3q_4) + \dot{q}_2(q_2^2 + q_4^2) + \dot{q}_3(q_1q_4 + q_2q_3)\right) \\ \omega_z = \frac{1}{q_4}\left(\dot{q}_1(q_1q_3 + q_2q_4) + \dot{q}_2(q_2q_3 - q_1q_4) + \dot{q}_3(q_2^2 + q_4^2)\right) \end{cases} \tag{4.9}$$

to find out the relation between quaternions and angular velocities, is sufficient to make the time derivative of Eq. 4.9, leading to Eq. 4.10

$$\begin{cases} \dot{omega}_x = & \frac{1}{q_4}(\ddot{q}_1(q_1^2+q_4^2)+\ddot{q}_2(q_1q_2+q_3q_4)+\ddot{q}_3(q_1q_3-q_2q_4)+ \\ & +2\dot{q}_1(q_1\dot{q}_1+q_4\dot{q}_4)+\dot{q}_2(\dot{q}_1q_2+q_1\dot{q}_2+\dot{q}_3q_4+q_3\dot{q}_4)+ \\ & +\dot{q}_3(\dot{q}_1q_3+q_1\dot{q}_3-\dot{q}_2q_4-q_2\dot{q}_4)-\dot{q}_4\omega_x) \\ \dot{omega}_y = & \frac{1}{q_4}(\ddot{q}_1(q_1q_2-q_3q_4)+\ddot{q}_2(q_2^2+q_4^2)+\ddot{q}_3(q_1q_4+q_2q_3)+ \\ & +\dot{q}_1(q_1q_2+q_1q_2-q_3q_4-q_3q_4)+2\dot{q}_2(q_2q_2+q_4q_4)+ \\ & +\dot{q}_3(q_1q_4+q_1q_4+q_2q_3+q_2q_3)-\dot{q}_4\omega_y)) \\ \dot{omega}_z = & \frac{1}{q_4}(\ddot{q}_1(q_1q_3+q_2q_4)+\ddot{q}_2(q_2q_3-q_1q_4)+\ddot{q}_3(q_2^2+q_4^2)+ \\ & +\dot{q}_1(\dot{q}_1q_3+q_1\dot{q}_3+\dot{q}_2q_4+q_2\dot{q}_4)+\dot{q}_2(\dot{q}_2q_3+q_2\dot{q}_3-\dot{q}_1q_4-q_1\dot{q}_4)+ \\ & +2\dot{q}_3(q_2\dot{q}_2+q_4\dot{q}_4)-\dot{q}_4\omega_z) \end{cases} \tag{4.10}$$

Now that all the kinematic relations have been obtained, it is possible to notice that in order to have a completely analytical formulation of the guidance, it is necessary to impose on the quaternions a shape that possesses the following requirements:

- The shaping function for the quaternions $q_i = q_i(t)$ shall be continuous and differentiable up to the second derivative in the domain $0 \leq t \leq t_{fin}$.
- The shaping function for the quaternions shall posses an analytical representation up to the second derivative in the domain $0 \leq t \leq t_{fin}$.
- The shaping function for the quaternions shall posses a sufficiently wide number of degrees of freedom to impose initial and final conditions on the quaternions value, as in Eq. 4.11, in which $\bar{q}_{i,ini}$ and $\bar{q}_{i,fin}$ are respectively the initial and final prescribed quaternions.

$$\begin{cases} q_i(0) = \bar{q}_{i,ini} \\ q_i(t_f) = \bar{q}_{i,fin} \end{cases} \tag{4.11}$$

- The shaping function for the quaternions shall posses a sufficiently wide number of degrees of freedom to impose initial and final conditions on the first time derivative of the quaternions value, as in Eq. 4.12, in which $\bar{\dot{q}}_{i,ini}$ and $\bar{\dot{q}}_{i,fin}$ are respectively the initial and final prescribed time derivative of the quaternions, that can be easily computed starting from the prescribed initial and final angular velocities using Eq. 4.8.

$$\begin{cases} \dot{q}_i(0) = \bar{\dot{q}}_{i,ini} \\ \dot{q}_i(t_f) = \bar{\dot{q}}_{i,fin} \end{cases} \tag{4.12}$$

Section 4.2 describes the quaternion shaping strategy developed to match the requirements here-above described.

4.2 Quaternion shaping

The aim of this section is to describe the developed shaping strategy for the quaternions, to be inserted in the reverted dynamics presented in Section 4.1. The kinematic, expressed using the quaternions, is parametrized with respect to the non-dimensional time, defined as in Eq. 4.13

$$x = \frac{t}{T} \quad \rightarrow \frac{df(t)}{dt} = \frac{1}{T}\frac{df(t)}{dx} \quad \text{with} \quad 0 \leq x \leq 1 \tag{4.13}$$

In this section, for sake of simplicity, the derivative with respected to time and non-dimensional time are marked as in Eq. 4.14

$$\begin{cases} \frac{df}{dt} = \dot{f} \\ \frac{df}{dx} = f' \end{cases} \tag{4.14}$$

The quaternion shaping is based on a non-linear interpolation between the natural evolution of the initial and final quaternions, $q_i^{ini}(t)$ and $q_i^{fin}(t)$ respectively. The natural evolution of the initial and final attitude is computed outside the optimization loop, to increase the computational efficiency of the algorithm, and can include the natural perturbations more relevant in a LEO environment [41]:

- Gravity gradient
- Solar pressure torque
- Magnetic torque
- Atmospheric drag torque

The non-linear interpolation adopted to shape each quaternion component is performed following Eq. 4.15, as shown in Fig. 4.1.

$$q_{i,nn}(x) = \Delta q_i \chi(x) + q_i^{ini} \quad \text{with} \quad \Delta q_i = \left(q_i^{fin}(t) - q_i^{ini}(t)\right) \tag{4.15}$$

Following this approach, it is important to point out that, for a general set of interpolating functions $\chi(x)$ the norm of the interpolated quaternion $\boldsymbol{q}_{nn}(x)$ is not unitary, and therefore a dedicated normalization step will be needed, as described later in this Section.

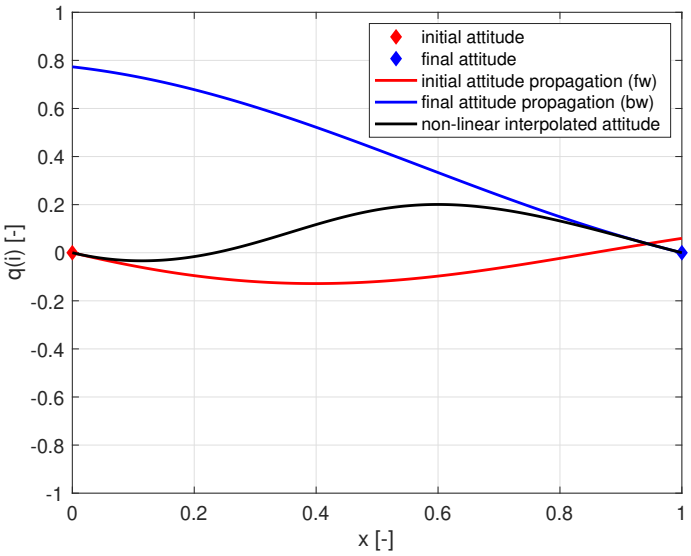


Figure 4.1: Quaternions shaping

To compute the full kinematic of the rigid body attitude motion, the first and second time derivative of the quaternion shall be computed, as in Eq. 4.16 and Eq. 4.17 respectively

$$\begin{aligned} \dot{q}_{i,nn}(x) =& \frac{d\left(\Delta q_i \chi(x) + q_i^{ini}\right)}{dt} = \Delta\dot{q}_i\chi(x) + \frac{\Delta q_i}{T}\chi(x)' + \dot{q}_i^{ini} \\ & \text{with} \quad \Delta\dot{q}_i = \left(\dot{q}_i^{fin} - \dot{q}_i^{ini}\right) \end{aligned} \tag{4.16}$$

$$\begin{aligned} \ddot{q}_{i,nn}(x) =& \frac{d\left(\dot{q}_{i,nn}(x)\right)}{dt} = \Delta\ddot{q}_i\chi(x) + 2\frac{\Delta\dot{q}_i}{T}\chi(x)' + \frac{\Delta q_i}{T^2}\chi(x)'' + \ddot{q}_i^{ini} \\ & \text{with} \quad \Delta\ddot{q}_i = \left(\ddot{q}_i^{fin} - \ddot{q}_i^{ini}\right) \end{aligned} \tag{4.17}$$

The boundary conditions to be satisfied are listed in Eq (4.18)

$$\begin{cases} q_i(0) = \bar{q}_{i,ini} \\ q_i(1) = \bar{q}_{i,fin} \\ \dot{q}_i(0) = \bar{\dot{q}}_{i,ini} \\ \dot{q}_i(1) = \bar{\dot{q}}_{i,fin} \end{cases} \tag{4.18}$$

in which the initial and final values of quaternions can be computed starting from the initial and final cosine matrix, as in Eq. 4.19

$$\begin{cases} q_1 = \frac{1}{4q_4}\left(A_{2,3} - A_{3,2}\right) \\ q_2 = \frac{1}{4q_4}\left(A_{3,1} - A_{1,3}\right) \\ q_3 = \frac{1}{4q_4}\left(A_{1,2} - A_{2,1}\right) \\ q_4 = \pm\frac{1}{2}\sqrt{1 + A_{1,1} + A_{2,2} + A_{3,3}} \end{cases} \tag{4.19}$$

while the derivative of quaternions at the initial and final instant can be computed using Eq. 4.8. To compute the time evolution of the second derivative of the quaternions, that are required to compute the variation of angular velocity, as shown in Eq. 4.10, it is necessary to compute the second order time derivative of the initial and final quaternions, as can be seen looking at Eq. 4.17. These values can be computed taking the time derivative of Eq. 4.8, as shown in Eq. 4.20

$$\begin{cases} \ddot{q}_1 = \dot{q}_2\omega_z + q_2\dot{\omega}_z + \dot{q}_3\omega_y + q_3\dot{\omega}_y + \dot{q}_4\omega_x + q_4\dot{\omega}_x \\ \ddot{q}_2 = -\dot{q}_1\omega_z - q_1\dot{\omega}_z + \dot{q}_3\omega_x + q_3\dot{\omega}_x + \dot{q}_4\omega_y + q_4\dot{\omega}_y \\ \ddot{q}_3 = \dot{q}_1\omega_y + q_1\dot{\omega}_y + \dot{q}_2\omega_x + q_2\dot{\omega}_x + \dot{q}_4\omega_z + q_4\dot{\omega}_z \\ \ddot{q}_4 = -\dot{q}_1\omega_x - q_1\dot{\omega}_x + \dot{q}_2\omega_y + q_2\dot{\omega}_y + \dot{q}_3\omega_z + q_3\dot{\omega}_z \end{cases} \tag{4.20}$$

One of the advantages of the proposed method to parameterize the equation of motion is that the boundary conditions can be applied directly on the $\chi(x)$ function, and are identical for all the attitude maneuvers, with a substantial beneficial effect on the CPU time. In particular, the boundary conditions identified in Eq. 4.18 turns into the $\chi(x)$ function behavior as in Eq. 4.21

$$\begin{cases} \chi(0) = 0 \\ \chi(1) = 1 \\ \chi'(0) = 0 \\ \chi'(1) = 0 \end{cases} \tag{4.21}$$

An extremely wide number of function could be used to parameterize the quaternions, the most important requirement is that, in order to impose the four boundary conditions expressed in Eq. 4.21, at least four parameters are needed. For sake of simplicity and to include a degree of freedom in the maneuver design, a fourth order polynomial is selected, as in Eq. 4.22,

$$\begin{cases} \chi(x) = a + bx + cx^2 + dx^3 + ex^4 \\ \chi'(x) = b + 2cx + 3dx^2 + 4ex^3 \end{cases} \tag{4.22}$$

in which the e_i parameters are the degrees of freedom, while the parameter a_i, b_i, c_i, d_i, belong to the imposition of the boundary conditions stated in Eq.4.21, as in Eq. 4.23

$$\begin{cases} a_i = 0 \\ b_i = 0 \\ c_i = 3 + e_i \\ d_i = -2(1 + e_i) \end{cases} \tag{4.23}$$

therefore the shaping function and its derivatives reads as in Eq. 4.24. The behavior of the function itself, spanning the e_i from -10 to $+10$ is presented in Fig.4.2.

$$\begin{cases} \chi_i(x) = 3x^2 - 2x^3 + e_i(x^2 - 2x^3 + x^4) \\ \chi_i'(x) = 6x - 6x^2 + e_i(2x - 6x^2 + 4x^3) \\ \chi_i''(x) = 6 - 12x + e_i(2 - 12x + 12x^2) \end{cases} \tag{4.24}$$

It is easy to verify that the boundary conditions on the quaternions and the time derivative of the quaternions are always respected adopting this family of shaping functions regardless of the specific maneuver.

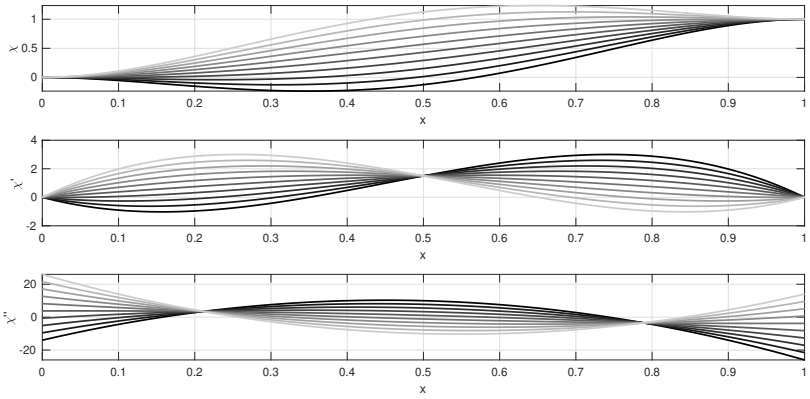


Figure 4.2: Shaping function behavior with e_i ranging from -10 to $+10$.

Since each component of quaternion is shaped independently with respect to the others, as described in Eq.4.15 , the norm of the shaped quaternion is not unitary. For this reason a dedicated normalization shall be applied to the shaped quaternions and to their derivatives, as described in Eq. 4.25

$$\begin{cases} q(i) = \frac{q_{nn}(i)}{|q_{nn}|} \\ \dot{q}(i) = \frac{\dot{q}_{nn}(i) - q(i)|\dot{q}_{nn}|}{|q_{nn}|} \\ \ddot{q}(i) = \frac{\ddot{q}_{nn}(i) - 2\dot{q}(i)|\dot{q}_{nn}| - q(i)|\ddot{q}_{nn}|}{|q_{nn}|} \end{cases} \tag{4.25}$$

in which the norm of the quaternions and the first and second derivative of the norm are reported in Eq. 4.26.

$$\begin{cases} |q_{nn}| = \sqrt{q_{nn}(1)^2 + q_{nn}(2)^2 + q_{nn}(3)^2 + q_{nn}(4)^2} \\ |\dot{q_{nn}}| = \frac{1}{|q_{nn}|}\left(\sum_{i=1}^{4} q_{i,nn}\dot{q}_{i,nn}\right) \\ |\ddot{q_{nn}}| = \frac{1}{|q_{nn}|}\left(\sum_{i=1}^{4}\left(q_{i,nn}\ddot{q}_{i,nn} + q_{i,nn}\dot{q}_{i,nn}^2\right) - |\dot{q_{nn}}|^2\right) \end{cases} \tag{4.26}$$

The parametrization here presented respect the exact kinematic and dynamic of the attitude motion of a rigid spacecraft for any type of maneuver. The analytical nature of the method is beneficial for the computational effort, making the algorithm suitable for solving even complex problems in a reduced amount of time. In the following, some relevant test cases are investigated to test the algorithm.

4.3 Test Cases

In this section some relevant test cases are presented and discussed. In particular, two different scenarios are investigated:

- *Unconstrained slew maneuver with final and initial non-null angular velocity.* In this scenario, a CubeSat shall perform a large slew maneuver starting and targeting a non null angular velocity. The CubeSat implements realistic actuator
- *Time-Optimal Spacecraft Reorientation with Keep-Out Cones.* A large spacecraft shall perform a slew maneuver, avoiding some forbidden pointing directions. This is a classical test case that can be found in literature [73].

4.3.1 Case 1: Unconstrained slew maneuver with final and initial non-null angular velocity

In this scenario, a CubeSat shall perform a large slew maneuver starting and targeting a non null angular velocity. In particular the CubeSat has an inertia matrix as reported in Eq. 4.27

$$\boldsymbol{I}_{rigid}^{global} = \begin{bmatrix} 0,065 & 0 & 0 \\ 0 & 0,055 & 0 \\ 0 & 0 & 0,012 \end{bmatrix} \text{kg m}^2 \tag{4.27}$$

and is actuated by four RWs with a maximum torque $\dot{h}_r = 2\,\text{mN m}$, and a maximum sortable angular momentum $h_r = 0.0019\,\text{N m s}$. The mounting direction of the RWs in the BF RF is reported is Eq. 4.28.

$$\boldsymbol{A}_{RWs}^{assembly} = \begin{bmatrix} 1 & 0 & 0 & \frac{1}{\sqrt{3}} \\ 0 & 1 & 0 & \frac{1}{\sqrt{3}} \\ 0 & 0 & 1 & \frac{1}{\sqrt{3}} \end{bmatrix} \tag{4.28}$$

Moreover, the spacecraft is equipped with body-mounted and wings solar panels, that will be considered in some of the objective of the optimization. These solar panels are mounted in the following configuration:

- $0.12\,\text{m}^2$ along the normal direction $\boldsymbol{i}_{N,1} = [-1, -1, 0]^T$
- $0.02\,\text{m}^2$ along the normal direction $\boldsymbol{i}_{N,2} = [-1, 0, 0]^T$
- $0.02\,\text{m}^2$ along the normal direction $\boldsymbol{i}_{N,3} = [0, -1, 0]^T$

The maneuver itself consist of a slew of 230° around the axis $\boldsymbol{i}_{rot} = [-1, -0.5, 1]^T$. At the initial instant, the body-fixed reference frame coincides with the inertial one, and therefore the DCM is $\boldsymbol{A}_{ini} = diag(1, 1, 1)$. The maneuvering time is in the range 10–110 s. The initial and final angular velocities are respectively $\boldsymbol{\omega}_{ini} = [-0.06, -0.04, 0.04]$ rad s^{-1} and $\boldsymbol{\omega}_{fin} = [-0.04, 0.01, 0.02]$ rad s^{-1}. This scenario, that is implemented in MATLAB 2021b, was optimized using three different algorithms: Interior Points (I-P) [74][75], PSO [76] and Nelder-Mead (N-M) [77], and run on a single core of a 2.6 GHz Intel i-7 processor.

The problem was optimized for three different Objective, namely:

- Obj A: *Minimum torque.* The aim of this objective is to minimize the peak of torque required to the RWs, namely $J = max(\dot{\boldsymbol{h}}_{RWs}(t))$.
- Obj B: *Maximum Sun energy.* The aim of this objective is to maximize the integral flux received by the solar panels, namely $J = -\sum_{j=1}^{3} \left(\int_0^{t_f} \psi \boldsymbol{n}_{Sun} \cdot \boldsymbol{A}^T(t) \boldsymbol{i}_{N,j} dt \right)$.
- Obj C: *Minimum Time.* The aim of this objective is to minimize the maneuvering time, namely $J = t_f$.

The actuator limitations, namely the maximum torque $\dot{h}_r$ and the maximum stored momentum h_r, are imposed by means of penalty functions directly on the objective functions, as in Eq. 4.29

$$J_{constr} = J + f^{\dot{h}_r}_{penalty}(\dot{h}_r) + f^{h_r}_{penalty}(h_r) \tag{4.29}$$

Among all the possible penalty functions, parabolic ones have been selected to ensure faster convergence to feasible solution in a reduced number of iterations, as in Eq. 4.30

$$f^{x}_{penalty}(x) = \begin{cases} 0 \quad if \quad |x| - x_{max} \leq 0 \\ a\left(\frac{|x|-x_{max}}{x_{max}}\right)^2 + b\left(\frac{|x|-x_{max}}{x_{max}}\right) + c \quad if \quad |x| - x_{max} > 0 \end{cases} \tag{4.30}$$

with a,b and c positive constants to be tuned accordingly to the specific problem. In particular, for the test cases under analysis the value of the constants are selected based on the value of the initial guess function J_{guess} as in Eq. 4.31

$$\begin{cases} a = 100 J_{guess} \\ b = 10 J_{guess} \\ c = J_{guess} \end{cases} \tag{4.31}$$

The PSO run with a population of 50 elements, with a randomly defined initial population, while the I-P and the N-M Simplex Method are initialized with $e_i = 0$ and $T_{man} = 60s$. The maneuvering time T_{man} is constrained to be in the range 10–110 s. The results are compare in Table 4.1. The solution found with the PSO algorithm, that in all the scenarios are the best, are compared in Fig. 4.4. In particular, it is possible to point out that the *Minimum Torque* and *Maximum Sun Energy* optimal solutions both try to maximize the maneuvering time, that results equal to the upper boundary. The *Maximum Sun Energy* optimal solution shows two peak of torque at the end and at the beginning of the maneuver, trying to spend as much time as possible in a quasi-sun-pointing condition, as can be seen from the right picture of Fig. 4.4, in which it is possible to see that the surface normal to the sun is close to the maximum value (0.148 m^2) for a large portion of the slew; quite the opposite,the *Minimum Torque* optimal solution, try to avoid peaks in the control action, preferring to distribute torque as uniformly as possible throughout the maneuver.

From Table 4.1, it is possible to see that the PSO algorithm is able to fin better solutions with respect to the N-M and to the I-P algorithms for all the analyzed cases. The reason of this fact is that the PSO algorithm, differently from the others, doesn't require a guess solution that is in the basin radius

	alg.	optimal value	fun. eval.	n° iter	CPU time
Obj A $J(x0) = 0.26mNm$	*I-P*	0.142 mN m	216	5	1.86 s
	PSO	0.0475 mN m	1900	37	15.26 s
	N-M	0.0594 mN m	178	99	1.73 s
Obj B $J(x0) = 1.18m^2s$	*I-P*	7.32 m^2 s	438	5	3.78 s
	PSO	10.63 m^2 s	6250	124	50.53 s
	N-M	5.00 m^2 s	208	108	1.92 s
Obj C $J(x0) = 60s$	*I-P*	20.58 s	468	1	3.65 s
	PSO	15.09 s	12300	245	100.55 s
	N-M	19.58 s	211	110	1.69 s

Table 4.1: Result Comparison. Case 1.

of the global minimum to converge on it. On the other hand, the CPU time required by the PSO is one to two order of magnitude higher with respect to the other, at least for the *swarm size* and *stopping criteria* here adopted, due to the higher number of function evaluation required. A complete analysis on the influence of these parameter to the optimality of the solution is still missing, and therefore the CPU time could be lowered, as can be appreciate by the convergence plot in Fig. 4.3 , but the computational cost will still remain higher if compared with I-P and N-M algorithms.

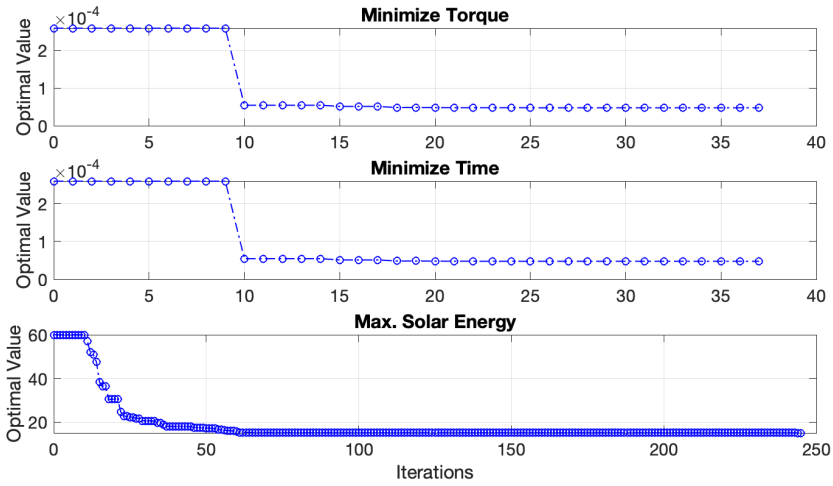


Figure 4.3: PSO convergence for case 1.

The I-P algorithm in particular, stops after a very limited number of iterations, with a cost function value that is sensibly higher with respect to the PSO.

Looking at the slew trajectories depicted in Fig. 4.4 and comparing the optimal solutions in Table 4.1 with the guesses solutions, it is possible to understand the flexibility of the developed shape-based algorithm, and its ability to locate near-optimal solutions. However, the control action computed, that is exact and that can be easily constrained, as can be seen in the Time-Optimal solution of Fig. 4.4, is forced to be continuous by the nature of the kinematic and the shape selected for the quaternion parametrization. Since optimal solutions typically shows *bang-bang* structures, the developed algorithm isn't able to locate the real optimal one. Anyway, the shape selected for the quaternion interpolation is sufficiently flexible to be able to 'simulate' the peak of control action, as can be seen in the *Minimum-Time* and Maximum-Sun-Energy examples in Fig. 4.4.

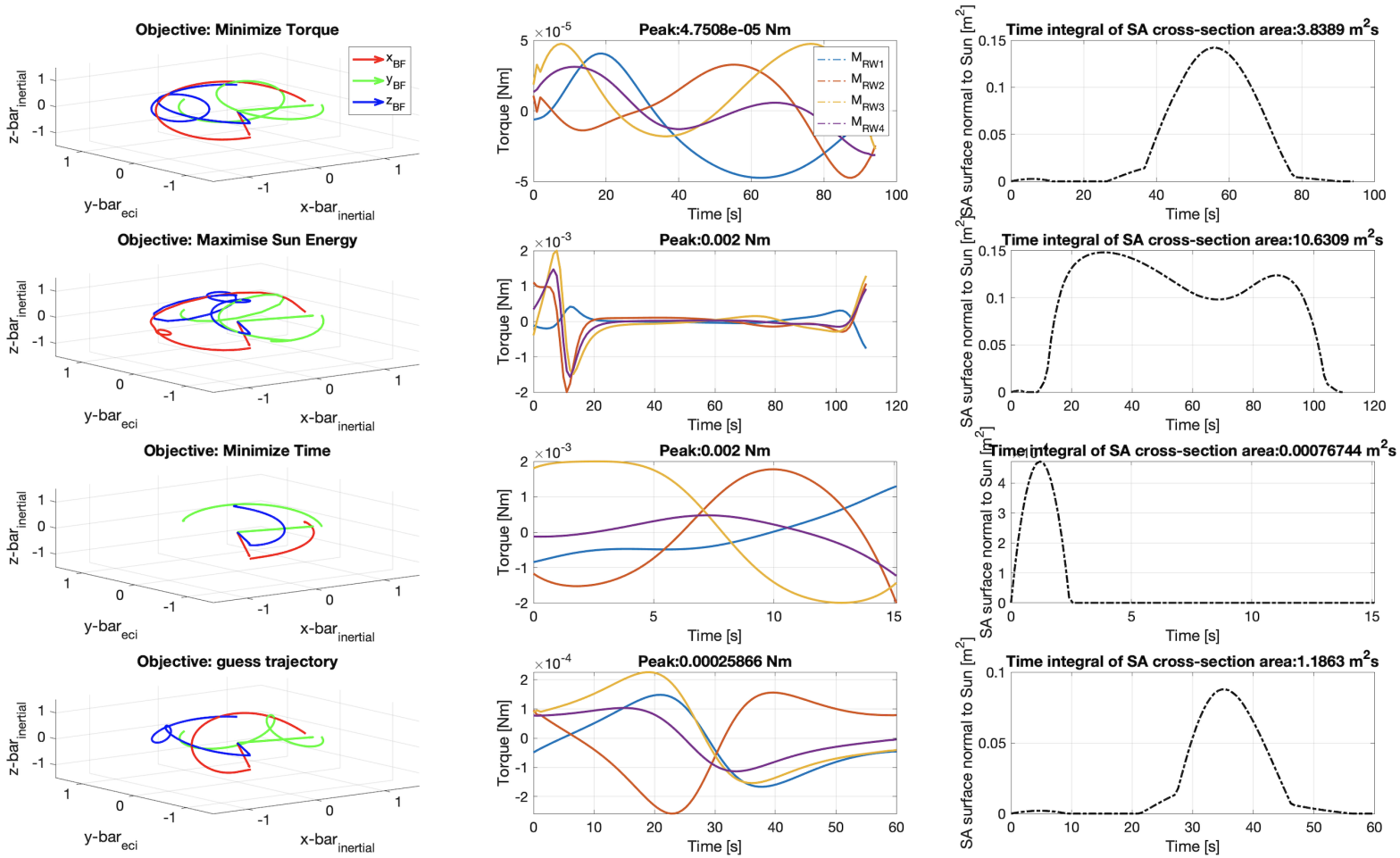


Figure 4.4: Optimal slew maneuvers considering different objectives compared with the first guess solution ($e_i = 0$ and $T_{man} = 55s$)

4.3.2 Time-Optimal Spacecraft Reorientation with Keep-Out Cones

This case study has been presented in [78] and [73]; it includes an Earth observation satellite in LEO. The satellite shall perform a roll rotation to switch the pointing direction on the other side of its track. Namely, the satellite, whose initial BF principal of inertia RF is aligned with the inertial one ($\boldsymbol{A}_{ini} = diag(1,1,1)$), shall perform a rotation of 135° around the inertial direction $\mathbf{i}_{rot} = [1,0,0]^T$. The attitude determination during the slew is ensured by a star tracker, with mounting direction in the BF RF coincident with the y axis, namely $\mathbf{i}_{ST,BF} = [0,1,0]^T$. The start tracker, to work properly, needs to avoid the Sun and the Moon, with conical keep-out zones, as reported in Table 4.2.

Keep-out cone	**Direction**	**half angle**
Sun	$\boldsymbol{i}_{Sun} = [-0.180.560.81]^T$	$\hat{\theta}_{Sun} = 47°$
Moon	$\boldsymbol{i}_{Moon} = [0.9500.31]^T$	$\hat{\theta}_{Moon} = 36°$

Table 4.2: keep-out cones definition

The satellites inertia matrix is $\boldsymbol{I}_{SC} = \text{diag}[3000, 4500, 6000] kgm^2$. In this scenario it is supposed to have on-board ideal actuators able to ensure a maximum torque along each BF axis of 0.25 N m. The objetive of the optimization is to minimize the maneuver time ($J = t_f$), while respecting the attitude constraints (initial attitude, final attitude, and keep-out cones) and the actuators limitations. In this scenario, the constraints are expressed by means of penalty functions as in Eq. 4.32

$$J_{constr} = J + f^{M_{max}}_{penalty}(\boldsymbol{M}) + f^{koc}_{penalty}(\boldsymbol{\theta}) \tag{4.32}$$

In which θ is the angle between the pointing direction of the instrument and the central direction of the keep-out cone. In particular, the unfeasible pointing direction that violates the keep-out cones are defined as in Fig. 4.5

Among all the possible penalty functions, parabolic ones have been selected to ensure faster convergence to feasible solution in a reduced number of iterations, as in Eq. 4.33

$$f^{x}_{penalty}(x) = \begin{cases} 0 \quad if \quad |x| - x_{max} \leq 0 \\ a\left(\frac{|x|-x_{max}}{x_{max}}\right)^2 + b\left(\frac{|x|-x_{max}}{x_{max}}\right) + c \quad if \quad |x| - x_{max} > 0 \end{cases} \tag{4.33}$$

the values of the constants a,b and c are defined as in the text case 1 and reported in Eq. 4.31.

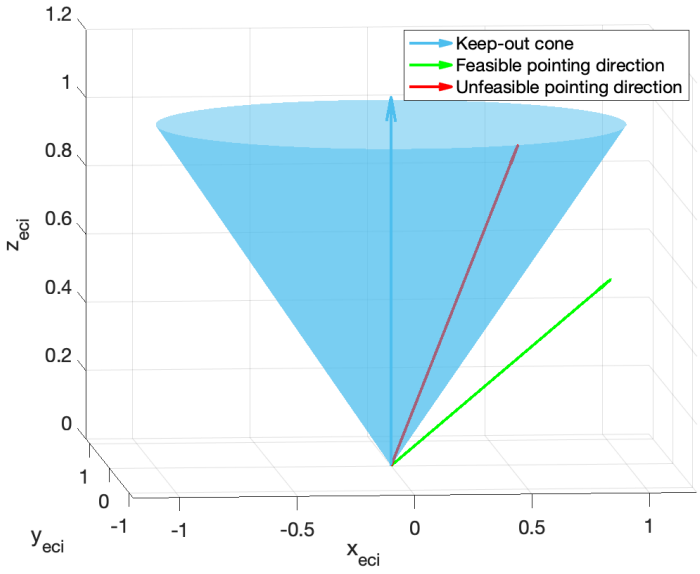


Figure 4.5: keep-out cones definition.

The test case under discussion is interesting to point out an important singularity that could reduce the effectiveness of the developed method, and a possible solution to mitigate it. As first, it is important to underline that the maneuver here described, being a rotation around the *inertial* x direction starting from the identity DCM, is characterized by the fact that the second and the third component of the initial and final quaternions are null. Moreover, the starting and desired angular velocities are zero, being a rest to rest maneuver. Since the trajectory is shaped via an interpolation between the initial and final quaternions propagation, the interpolated quaternion that describe the kinematic of the slew will surely present a second and third null components at any time of the maneuver. It turns into an undesired reduction of the search space, with a consequently possible reduction of the optimality of the solution and/or the impossibility to satisfy the attitude constraints. To prevent this effect, it is sufficient to solve the problem with a *support* inertial reference frame that is rotated about an axis different from the canonical base. Indeed, in such RF the rotation is no more aligned with an axis of the RF itself, with the consequence that all the components of the initial and final quaternions will change. In the here-presented solution, the *support* reference frame $\boldsymbol{A}_{sup}$ is rotated with respect to the *eci* of 10° around the axis $\mathbf{i}_{sup,eci} = [0, 2, -1]^T$.

The time-optimal problem was implemented in MATLAB 2021b and solved using a PSO algorithm running on a single core of a 2.6 GHz Intel i-7 processor. The PSO has been set with a swarm composed by 100 elements, and converges after 166 iterations and 107 s to a feasible solution (i.e. the keep-out cones are

avoided and the control action is within the thresholds) with a maneuvering time $t_f = 553.27s$. The convergence history of the PSO is reported in Fig. 4.6.

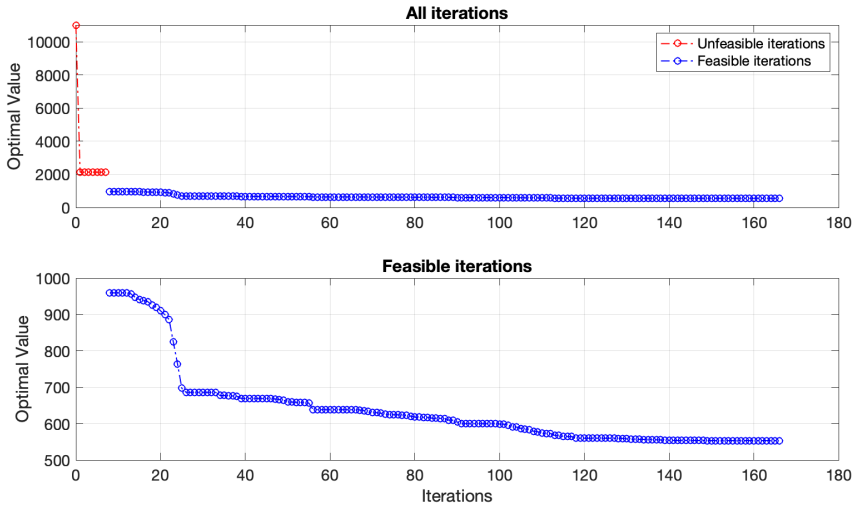


Figure 4.6: PSO convergence for case 1.

It is interesting to point out that the selected penalty functions approach is able to reach feasible trajectories in a very limited number of iterations: in the test here reported the first feasible solution appears after 7 iterations of unfeasible trajectories even in presence of a very complex and non mono-connected search domain. In the resulting optimal trajectory presented in Fig. 4.7 it is possible to appreciate the large slew required to pass through the narrow alley between the two keep-out cones.

A comparisons of the designed slew with the literature is presented in Table 4.3.

Algorithm	**Optimal value** (t_f)	**Computational Time**
Spiller et al. [78]	404 s	1523 s (324 s with a best guess)
Celani et al. [73]	755 s	8160 s
Prinetto	553 s	107 s

Table 4.3: literature comparison

.

The resulting optimal maneuvering time of $t_f = 553.27s$ found here is significantly lower with respect to the optimal value of $t_f = 755s$ found with

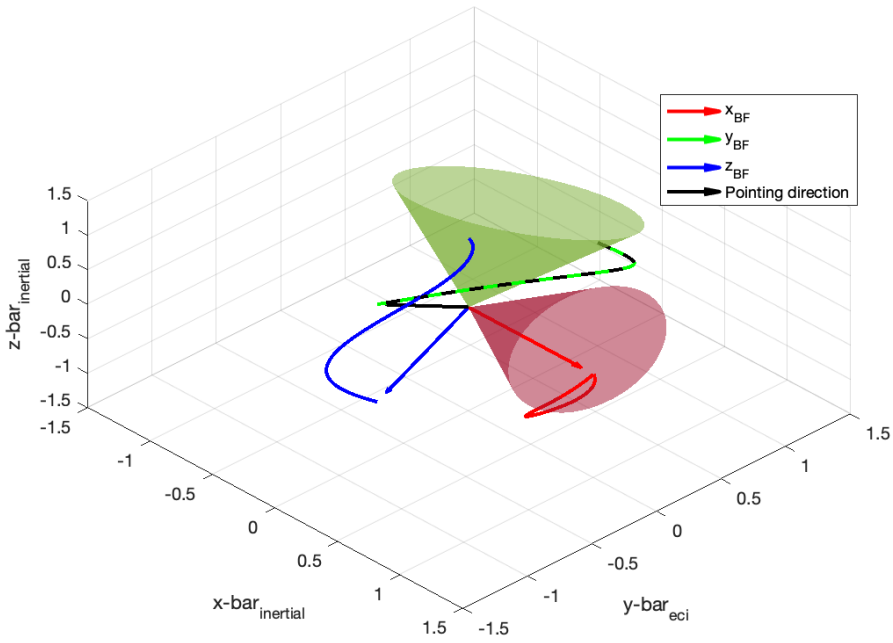


Figure 4.7: Optimal maneuver. Case 2

the shape-based algorithm presented in [73], but remains far from the real optimal value of $t_f = 404s$ found in [78]. Unfortunately in [78] the control action and angular velocity profile, as well as the trajectory representation, are missing since only the value of the optimal result is reported, therefore it is not possible to have a complete comparison between the two maneuvers. The small computational cost of the here reported algorithm turns out in a strong reduction of the CPU time required to run on similar processors.

The quaternion shaping approach here presented is able to generate trajectories that match exactly the initial and final attitude states, as can be seen from the quaternions evolution reported in Fig. 4.9. The resulting control action, reported in Fig. 4.8 is continuous and derivable, and shows a smooth variation, that is beneficial for the real actuators dynamics. It is easy also to include a constraint on the torque rate by means of a dedicated penalty function. Differently from the control action found in [73], in the work here presented the starting value is not null, since there are no contraints on the initial and final value of the second derivative of quaternions (see Sec. 4.2 for further details). To include this further constraint, it is sufficient to implement an interpolating function $\chi_{(x)}$ with a null second derivative with respect to the non-dimensional time x at the beginning and at the end of the maneuver. To do that, it is

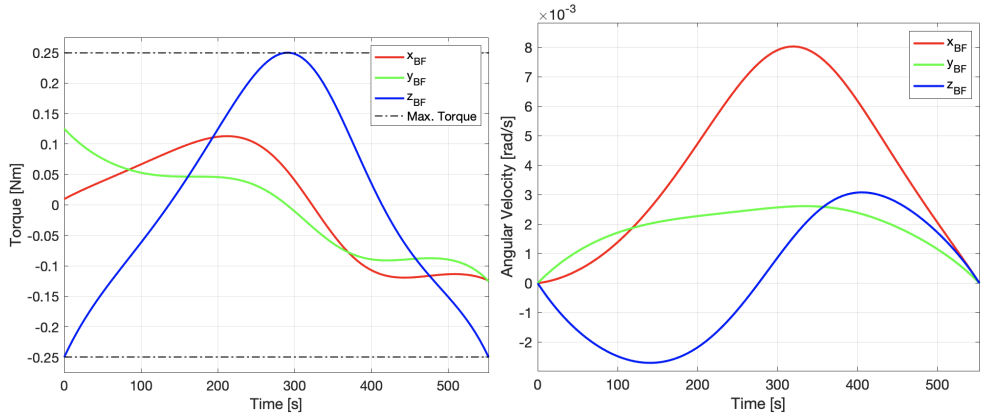


Figure 4.8: Optimal control torque (*left*) and angular velocity (*right*). Case 2

sufficient to use a proper sixth order polynomial instead of the fourth order used for this maneuver.

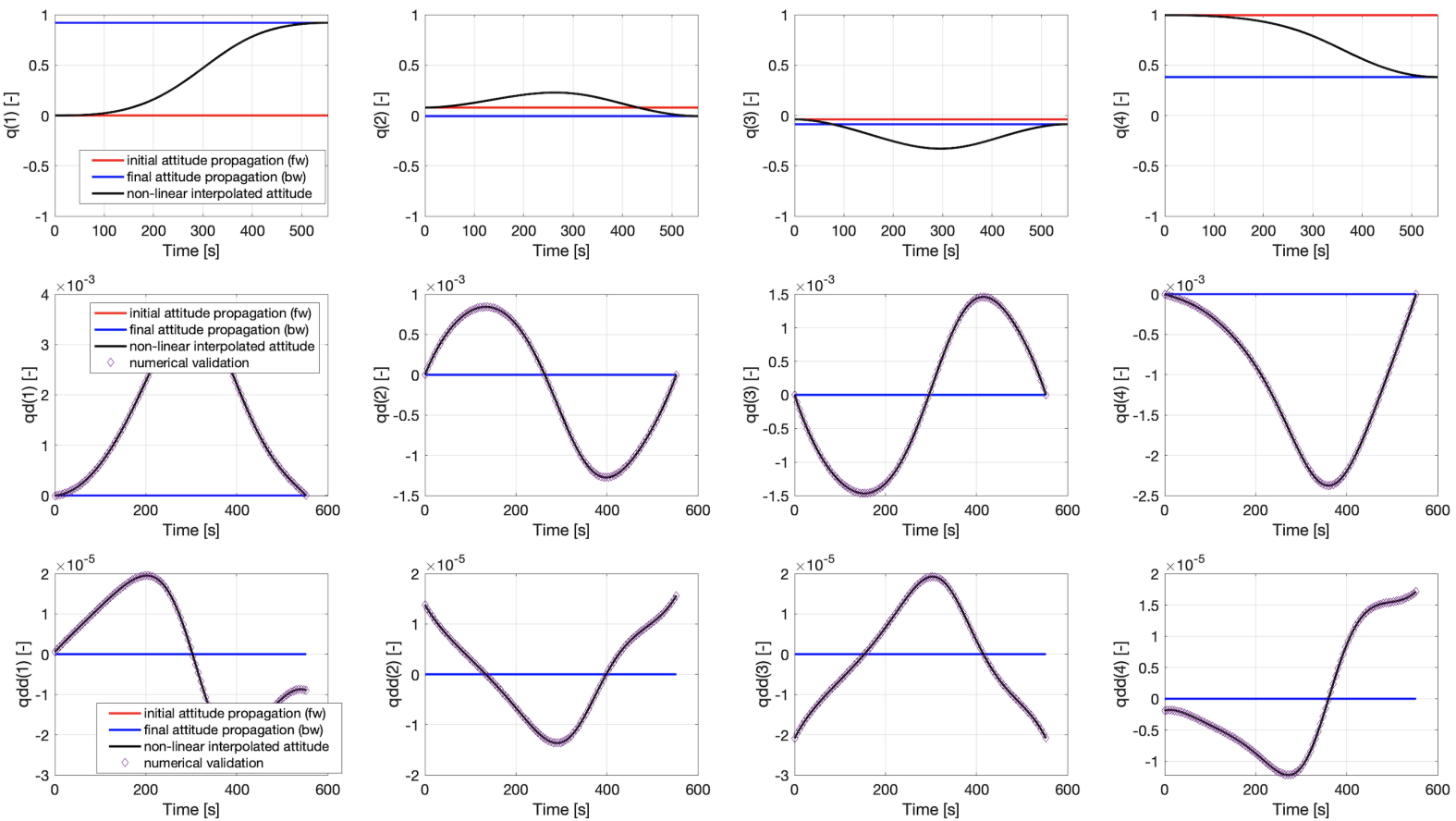


Figure 4.9: Time evolution of the quaternions and their derivatives for the optimal slew of case 2.

CHAPTER 5

Attitude Control Techniques

The aim of this chapter is to describe the on-board control strategies investigated to use the attitude guidance developed in Sec. 4. In particular, the control strategies and algorithm investigated are tailored on CubeSat Hardware and Software capabilities, therefore advanced and optimal but computationally heavy controllers, such as Linear Quadratic Regulator (LQR) or Model Predictive Control (MPC) are not considered within this dissertation, since they require an on-line optimization, as discussed in Sec. 2, that is beyond the real capabilities of actual CubeSat Hardware.

5.1 6 DoF simulator

The Attitude Control System (ACS) design is supported by a 6 DoF simulator, developed and validated by the author. It allows to simulate the coupled attitude and orbital dynamics of a spacecraft, including a high-fidelity representation of the environment [79][80]. The environment model takes into account the following perturbing forces:

- Earth geopotential based on EGM96 model.

- Solar gravitational perturbation (based on DE431 Ephemeris model).
- Lunar gravitational perturbation (based on DE431 Ephemeris model)
- Atmospheric drag perturbation based on Jacchia-Roberts model (F107-AP index 2020 update).
- Solar Radiation Pressure (SRP) (F107-AP index 2020 update).
- Gravity Gradient based on EGM96 mode.
- Magnetic torque based on IGRF.

The spacecraft is geometrically modeled by means of an assembly of planar surfaces representing its faces and solar arrays (if deployed). The simulator is implemented in the Simulink environment, The simulator does not implement any navigation algorithms, but a proper random error is added to the full state determination to replicate the navigation performances. On the other hands, it implements a wide set of actuators, including:

- A *generic ideal actuator*, capable to generate a control action $\mathbf{M_{control}}$, on which virtually any type of constraint (magnitude, rate, control frequency, state and time dependent constraints, etc..) can be added by means of non-linear control functions.
- *Reaction wheels*, with customizable assembly defined by the matrix $\boldsymbol{A}_{RWs}^{assembly}$, and with constraints on the momentum exchange magnitude $\dot{\mathbf{h}}_r$ (as function of the stored momentum $\mathbf{h}_r$), saturation, and control frequency. A control error on the full-scale torque is added to replicate the behavior of the internal reaction wheels control errors.
- A customizable set of *Thrusters* can be added, with user defined body-fixed thrusting direction. The thruster has constant thrust, tunable in Pulse Width Mode, with a customizable minimum impulse duration and control frequency. A Gaussian error is added on the control action.
- *Magneto-torquers*. The torque produced is a function of the current (the control variable) and the position of the spacecraft. A Gaussian error and a control frequency are implemented to replicate the real behavior of the actuators.

The environment and the dynamic of the simulator was validated during the develpement comparing the obtained results with literature [79, 80, 81]

5.2 Control Techniques

In the following sections, the three developed control technique are presented and discussed, with also the help of a test case. In particular the maneuver

consists of a rest-to-rest slew maneuver of 170° around the axis $\mathbf{i}_{rot} = [1, 1, 2]$, knowing that at the initial time $t = 0$ the body-fixed reference frame is aligned with the eci. The initial orbital state of the spacecraft is reported in Table 5.1.

Parameter	**Value**
Semi-major axis (a)	7328 km
Eccentricity (e)	0.0409
Inclination (i)	30°
Right ascension of the ascending node (RAAN)	100°
Argument of perigee (ω)	20°
True Anomaly (θ)	75°
Epoch	21/03/2023
Time (UTC)	0 h 0 min 0 s

Table 5.1: Spacecraft orbital initial state at $t = 0$

The maneuver is optimized with the attitude guidance algorithm described in Chapter 4, with the objective of minimizing the slew time. The maneuver is performed by the HERMES 3U CubeSat [82, 83, 84] depicted in Fig. 5.1., whose inertia properties $\boldsymbol{I}_{rigid}^{global}$, expressed in the principal axis of inertia, are reported in Eq. 5.1

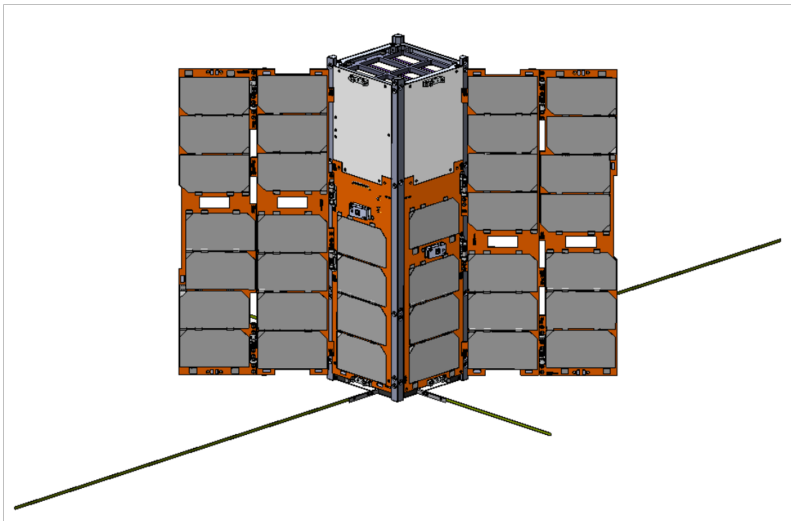


Figure 5.1: HERMES 3U CubeSat configuration

$$\boldsymbol{I}_{rigid}^{global} = \begin{bmatrix} 0,065 & 0 & 0 \\ 0 & 0,055 & 0 \\ 0 & 0 & 0,012 \end{bmatrix} \text{kg m}^2 \tag{5.1}$$

HERMES is actuated by four RWs, three of them are mounted along the principal axis of inertia and the fourth is along the diagonal, leading to an assembly matrix $\boldsymbol{A}_{RWs}^{assembly}$ as in Eq. 5.2.

$$\boldsymbol{A}_{RWs}^{assembly} = \begin{bmatrix} 1 & 0 & 0 & \frac{1}{\sqrt{3}} \\ 0 & 1 & 0 & \frac{1}{\sqrt{3}} \\ 0 & 0 & 1 & \frac{1}{\sqrt{3}} \end{bmatrix} \tag{5.2}$$

The specification of the ADCS performances are reported in Table 5.2. The errors, as well as the ADCS frequency, are considered only for the Control, while for the Guidance definition only ideal actuators are employed. In particular, due to the fastness of the maneuver, a relatively high ADCS frequency range was considered.

ADCS		
Frequency	5–20 Hz	
Control system		
Max. torque	2 mN m	for each RW
Momentum Storage	19 mN m s	for each RW
Control accuracy	0,015 %	full-scale
Determination system		
Error on position	5 m	3σ on each axis
Error on velocity	1 cm s^{-1}	3σ on each axis
Error on attitude	0.5°	3σ around each axis
Error on angular velocity	0.5 ° s^{-1}	3σ around each axis

Table 5.2: Determination and Control system specifications

The above mentioned scenario, from a mathematical point of view, includes five degrees of freedom: the four e_i parameters of the quaternions shape and the maneuvering time t_f. It was solved using the attitude guidance illustrated in Chapter 4 with three standard MATLAB optimization algorithms, representative of different families. The simulations run in MATLAB 2021b, on a machine equipped with a 2.6 GHz *Intel i7* processor using a single core. The PSO converges after 120 iteration with a population of 25 elements, with a randomly defined initial population, while the I-P and the N-M are initialized with $e_i = 0$ and $T_{man} = 17.5s$. The results are compare in Table 5.3

Generally speaking the solution found with the PSO algorithm shows better results in term of optimality, especially for highly non mono convex domains and/or when the maneuvering time is left free to be optimized, being able

Algorithm	Optimal slew time	CPU time
interior-points [74][75]	11.8527	2.617
particleswarm [76]	10.3218	18.716
Nelder-Mead Simplex Method [77]	10.8524	1.046

Table 5.3: Optimization algorithms comparison - fast slew maneuver

to reach easily the global optimum, but with a substantial increase in the computational time if compared with the others algorithms. The resulting optimal slew maneuver is presented in Fig. 5.2.

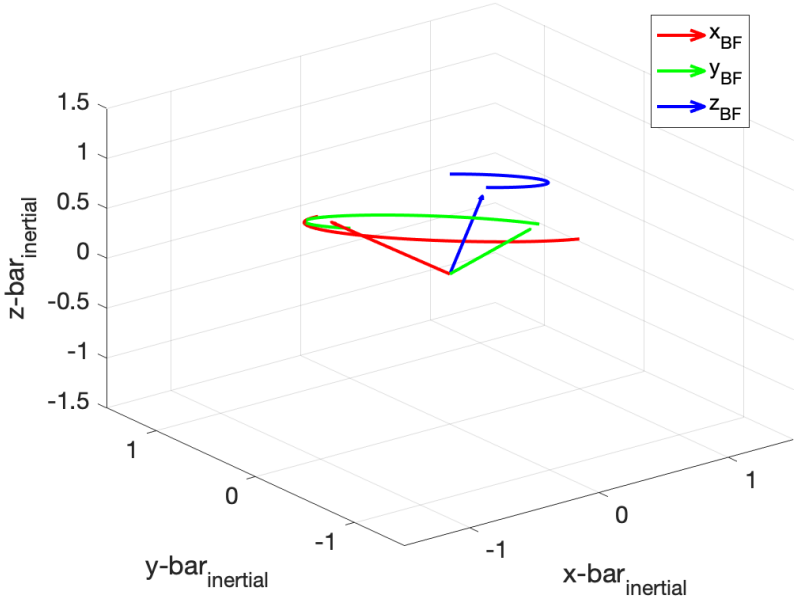


Figure 5.2: 3-D representation of the minimum-time optimal solution

Looking at the control action reported in Fig. 5.3 it is possible to see how the quaternion shaping strategy, with the shape selected, is able to generate a control law that is someway similar to the typical bang-bang structure of the real time-optimal control laws [39, 85]. The graph shows also that the reaction wheels never reach the saturation condition during this maneuver.

The quaternion and angular velocity behaviors are shown in Fig. 5.4 for sake of completeness.

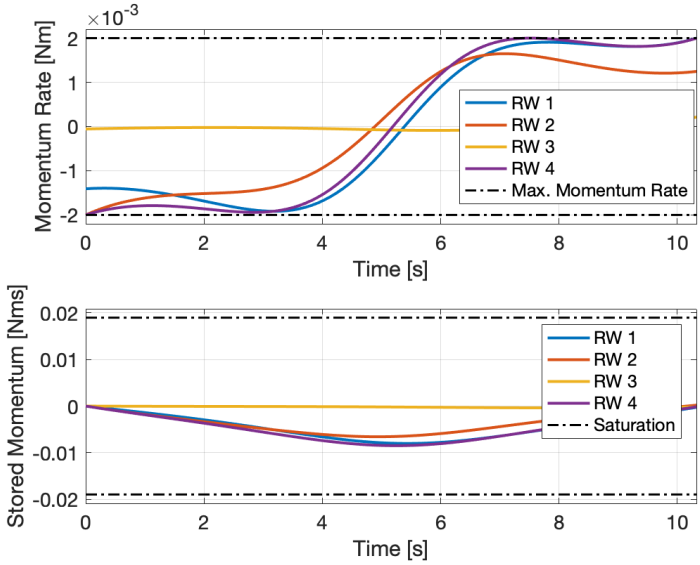


Figure 5.3: RWs control and status behavior of the minimum-time optimal solution

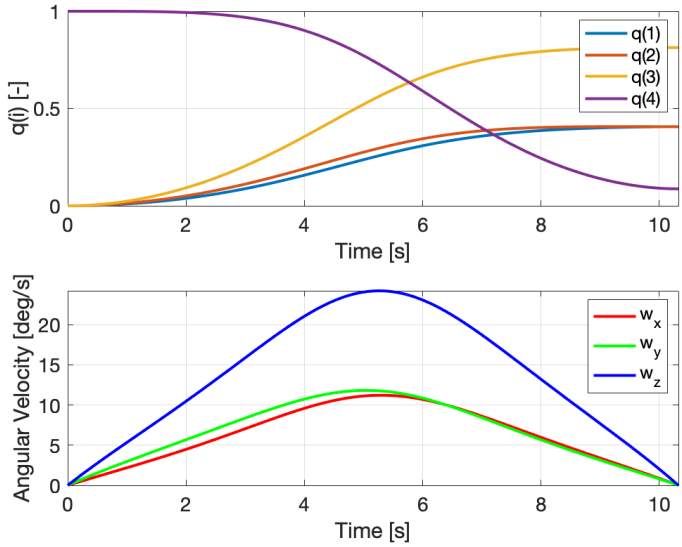


Figure 5.4: Quaternions and angular velocity behavior of the minimum-time optimal solution

The presented rest-to-rest slew maneuver is employed as support during the exposition and comparison of the developed control techniques, described in the following sections.

5.2.1 Feed-Forward open loop control

The simplest control law developed to test the attitude guidance algorithm described in Chapter 4 was an open loop control, as in presented in Fig. 5.5.

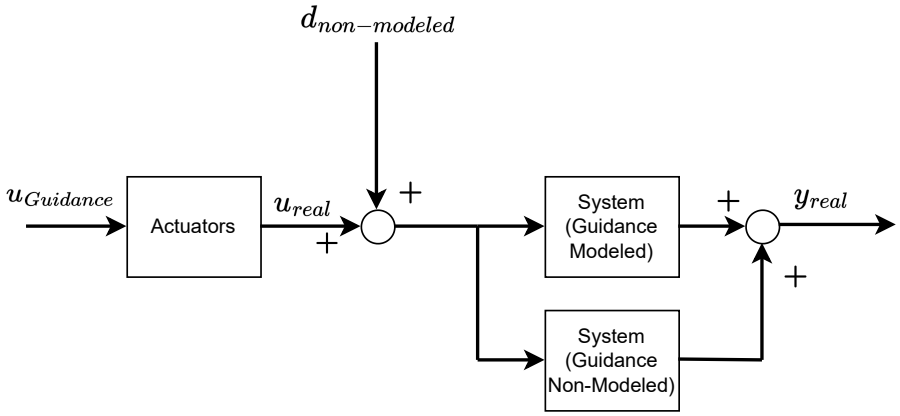


Figure 5.5: Open Loop control scheme

Practically speaking, the control action, defined and optimized on ground to satisfy all the requirements, and developed within a mid-fidelity environment with the algorithm described in Chapter 4, is updated directly to the platform. The CubeSat, then, apply the control law in an open loop cycle. Even if open loop control for attitude maneuvers is of very little (almost null) interest [86], it was decided to analyze this case to evaluate the performance of the developed guidance algorithm, once it is inserted in a perturbed high-fidelity environment with discrete time state determination and control. Figure 5.6 shows the behavior of the angular error with respect to the quaternion guidance, for different values of controller frequencies.

In particular can be noticed that, for controller frequencies much higher with respect to the slew one (the case of infinite or 10 Hz frequencies in Fig.5.6) the errors remain contained during the maneuver itself, even in absence of a closed loop control. It proves that the control action provided by the guidance described in Chapter 4 is very faithful to the real dynamics. On the other hands, over long periods it is impossible to control the system in open loop control due to the non-modeled dynamics, indeed in the guidance algorithm the actuators are ideal, with infinite frequency and null error. Moreover, as expected from theory, unpredicted disturbances and the finite time controller induce an uncontrolled drift after the end of the nominal slew that in few tens of seconds causes the satellite to completely lose attitude.

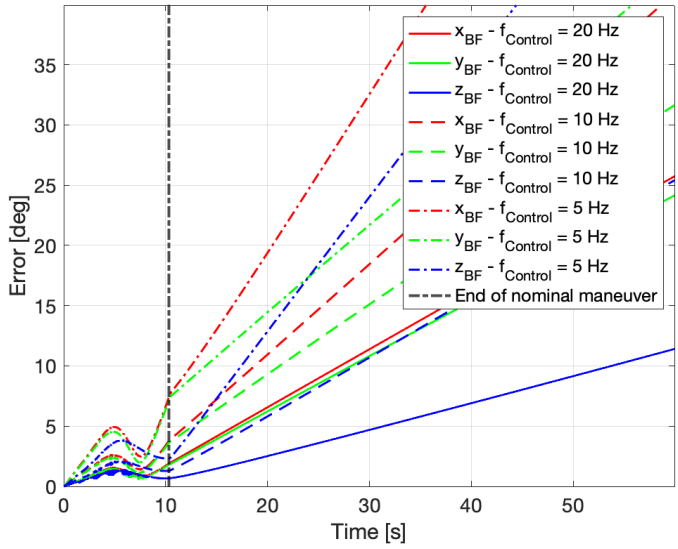


Figure 5.6: Angular error during Feed-Forward Open Loop controlled slew considering different control frequencies

5.2.2 Classical PID controller

The second control scheme developed is a classical closed loop control scheme that implements a Proportional Integral Derivative (PID) controller: the standard in almost every CubeSat flown. The success of this control scheme in the CubeSat world is due to many causes, among which the most important are surely the reliability, the reduced computational cost, the effectiveness, the stability and the TRL.

The high-level block scheme of the implemented control scheme is show in Fig. 5.7. Practically speaking, the control action is computed on-board considering the error between the *measured* attitude, determined on-board and affected by navigation errors, and the *desired* attitude, uploaded from ground in the form quaternion profiles, as shown in Sec. 5.2.

In particular, given these two informations, the matrix of the attitude errors $\mathbf{A}_e$ can be computed as in Eq. 5.3

$$\mathbf{A}_e = \mathbf{A}_m \mathbf{A}_d^T \tag{5.3}$$

in which $\mathbf{A}_m$ and $\mathbf{A}_d$ are the *measured* and *desired* attitude matrix respectively. Moreover, the derivative portion of the controller can be seen as a proportional control with respect to the angular velocity error $\boldsymbol{\omega}_e$ computed as in Eq. 5.4,

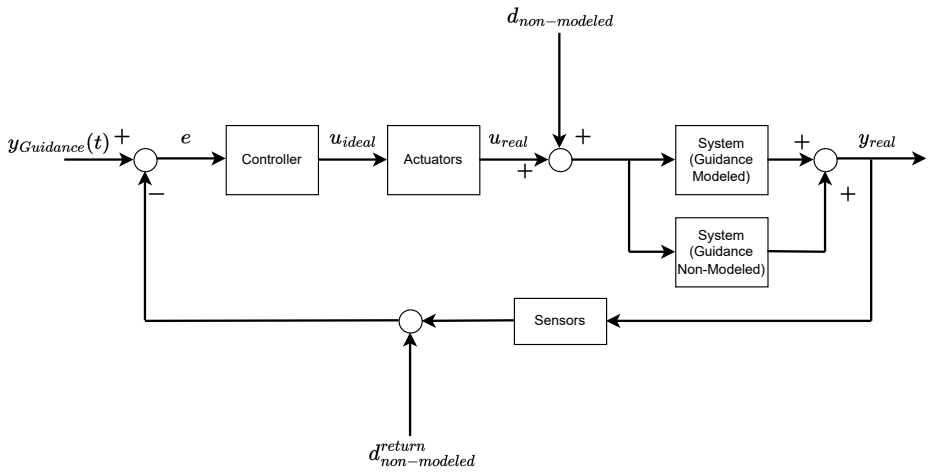


Figure 5.7: PID control scheme

in which $\boldsymbol{\omega}_m$ and $\boldsymbol{\omega}_d$ are the *measured* and *desired* angular velocities expressed in the body-fixed reference frame.

$$\boldsymbol{\omega}_e = \boldsymbol{\omega}_m - \boldsymbol{\omega}_d \tag{5.4}$$

The aim of the control is to make the attitude error matrix $\mathbf{A}_e$ as close as possible to the *identity* $\mathbf{I}$, while drive to zero the error on the angular velocity $\boldsymbol{\omega}_e$, to minimize the pointing error. It is possible to demonstrate that, for linearized motion, the torque to apply to the CubeSat can be computed as reported in Eq. 5.5

$$\begin{cases} \mathbf{M}_{c,x} = \frac{K_{p,x}}{2}\left(\mathbf{A}_e(2,3) - \mathbf{A}_e(3,2)\right) + K_{d,x}\boldsymbol{\omega}_e(1) \\ \mathbf{M}_{c,y} = \frac{K_{p,y}}{2}\left(\mathbf{A}_e(3,1) - \mathbf{A}_e(1,3)\right) + K_{d,y}\boldsymbol{\omega}_e(2) \\ \mathbf{M}_{c,z} = \frac{K_{p,z}}{2}\left(\mathbf{A}_e(1,2) - \mathbf{A}_e(2,1)\right) + K_{d,z}\boldsymbol{\omega}_e(3) \end{cases} \tag{5.5}$$

In which $K_{p,i}$ and $K_{d,i}$ are the *proportional* and *derivative* gains of the controller along the i^{th} axis. In the implementation of the controller, to make the dynamical response along each axis as similar as possible, independently from the related inertia, the controller gains were tuned accordingly to Eq. 5.6

$$\begin{bmatrix} K_{p,x} \\ K_{p,y} \\ K_{p,z} \end{bmatrix} = \alpha_p \begin{bmatrix} I_{x,x} \\ I_{y,y} \\ I_{z,z} \end{bmatrix}; \qquad \begin{bmatrix} K_{d,x} \\ K_{d,y} \\ K_{d,z} \end{bmatrix} = \alpha_d \begin{bmatrix} I_{x,x} \\ I_{y,y} \\ I_{z,z} \end{bmatrix} \tag{5.6}$$

In this way, the angular acceleration around one axis produced by the controller consequently to a small angular error around the same axis is independent from the selection of the axis itself. The integral term of the controller was removed for at least three reason [87]; first of all, the need to numerically integrate the error requires computational resources that, as mentioned before, for CubeSat hardwares are very limited. The integral term is subjected to wind-up and control saturation, that can affect the performances. Even if anti wind-up and anti-saturation techniques are available, they requires computational resources that for CubeSat hardware can be non negligible. Moreover, the beneficial effect of the integral term on the steady-state error reduction is almost completely vanished by the poor performances of CubeSat navigation algorithms and sensors, that lead to a relevant Attitude Knowledge Error (AKE). To obtain a trajectory as close as possible to the reference, the tuning parameters of the gains, α_p and α_d are optimized using a PSO algorithm with the objective of minimizing the mean quadratic angular error with respect to the reference. The resulting controller, as can be appreciate in Fig. 5.10 is extremely aggressive, even if stable, leading to a strong control action. The value of the obtained gains tuning parameters are $\alpha_p = 0.1257$ and $\alpha_d = 0.6850$. Although the designed control is extremely aggressive, the spacecraft accumulates a large error during the slew, due to the behavior of the PD controller itself, as can be appreciated looking at the 3D trajectory of the real slew maneuver presented in Fig. 5.8, in comparison with the shape-based reference.

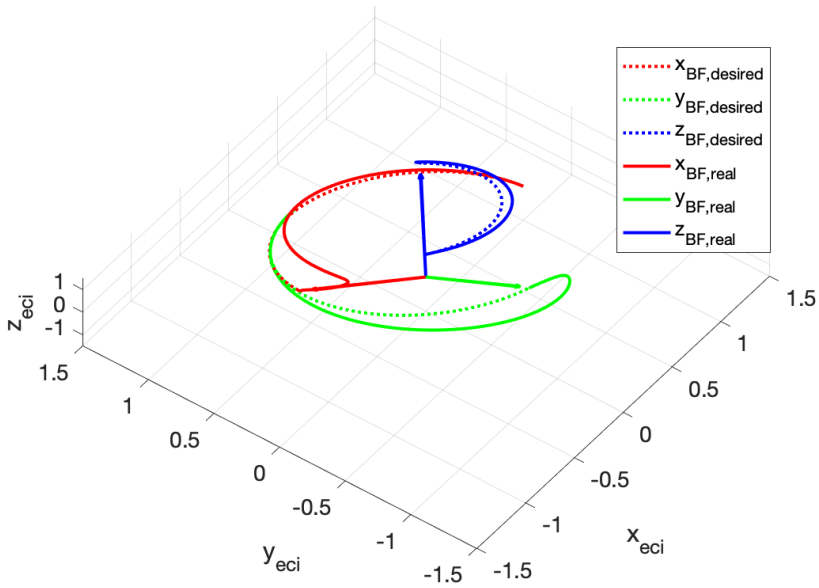


Figure 5.8: Trajectory of the PD controlled slew - control frequency = 10 Hz

In particular, Fig. 5.9 shows the angular error on each axis with respect to the reference trajectory.

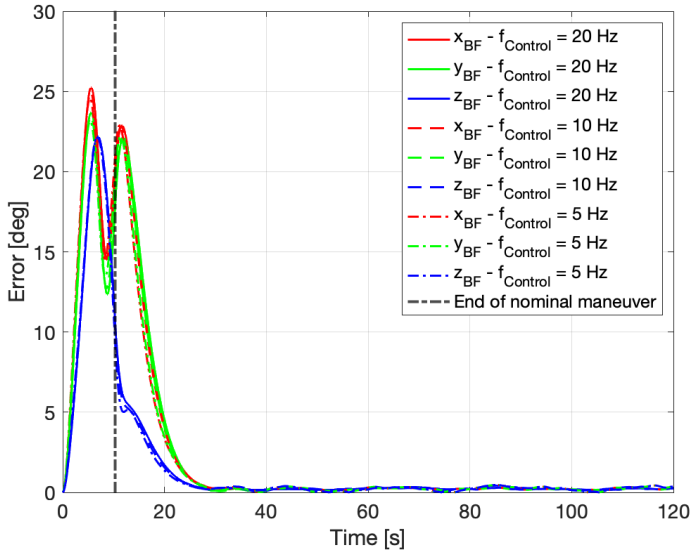


Figure 5.9: Angular error during PD controlled slew considering different control frequencies

From the graph it is possible to quantify the error that the PD controller needs to accumulate during the slew, and this quantity is a function of the control frequency. In particular, the lower is the control frequency, the higher is the error during the slew maneuver, as predicted from theory [88]. Moreover, the peak of the error is located nearby the end of the nominal maneuver, leading to a total effective maneuvering time that is three time higher with respect to the expected value. Another interesting element that can be pointed out from Fig.5.9 is the similar behavior the steady-state pointing error along the three axis, direct consequence of the selection of the inertia-scaled gains.

The analysis of the RWs status during the maneuver, depicted in Fig. 5.10, reveals the other weaknesses of this control strategy: the need to have an aggressive control action leads to the saturation of the actuator, in term of torque, and the persistence of oscillations, caused mainly by the delay induced bu the navigation frequency. From the same graph, it is possible to appreciate also the strong difference between the control action derived with the guidance algorithm, the one imposed by the on-board PD controller, and the actuated one, leading to a consistent difference in the optimality of the maneuver with respect to the shape-based one. This last effect, together with the large errors occurring during the maneuver, it is considered a very great disadvantage, which in some cases of practical interest, can be decisive [37].

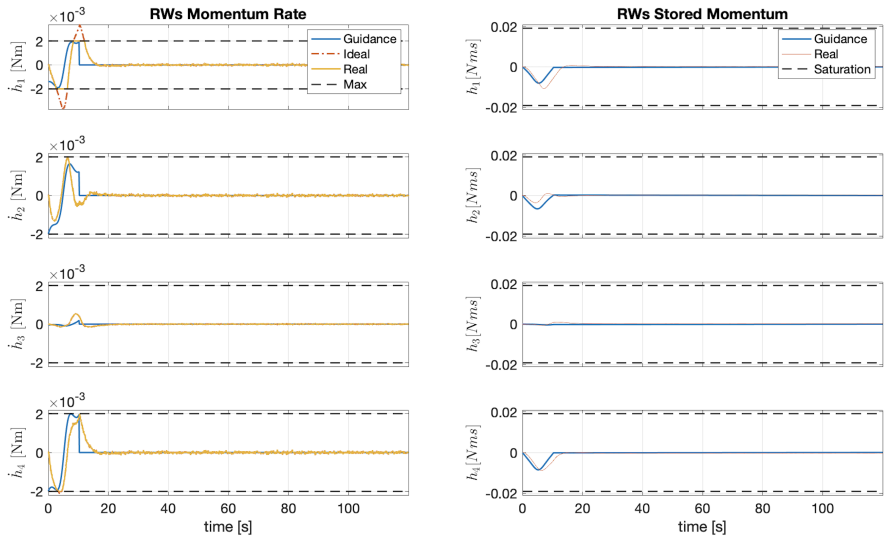


Figure 5.10: Comparison of nominal, ideal and real control actions during the slew maneuver

5.2.3 Feed-forward PID controller with Gain-Scheduling

The improvement of the pointing precision during the slew maneuver can be of primary importance for some application, such as the case in which the payload or some navigation instruments have forbidden pointing directions (e.g. payload Sun avoidance) [73] [38]. In such cases, as shown in Chapter 4, with the quaternion shaping technique it is possible to generate near-optimal guidance trajectories with keep-out cones. The problem now turns into the definition of a control scheme that is able to avoid the necessity of storing a non-negligible pointing error during the slew itself, as happens in the simple closed loop PD control scheme presented in Sec. 5.2.2, while maintaining the steady-state disturbance rejection and the moderation of the control that are typical of a well tuned PD controller. The working principle of the proposed control scheme is to take advantage of the control action calculated during the on-ground guidance definition to improve the performances of the PD controller shown in Sec. 5.2.2.

In particular, the control action is computed on-board starting from the error measurement and from the control action uploaded from ground, in a FFPD configuration, as shown in Eq. 5.7

$$\mathbf{u}_{ideal}(e,t) = \mathbf{u}_{PID}(e) + \mathbf{u}_{Guidance}(t) \tag{5.7}$$

The PD action is calculated in the same way as in Sec. 5.2.2, but in this case, to maximize the benefits from the fed-forward help, a gain scheduling is proposed.

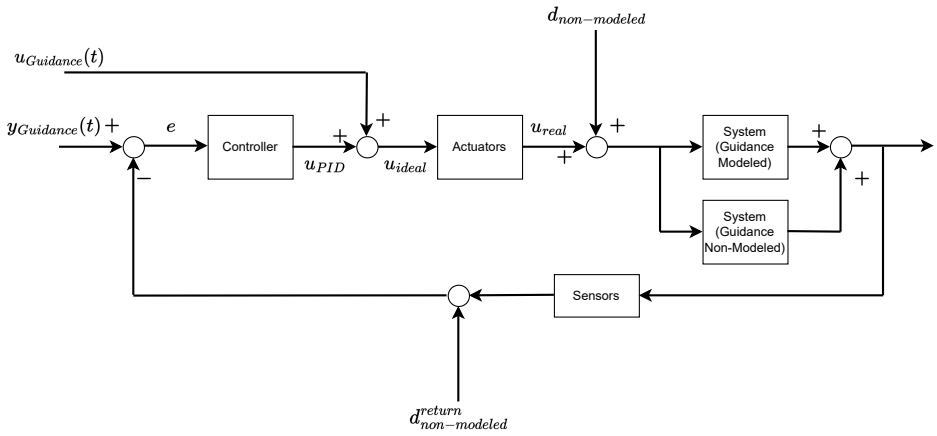


Figure 5.11: Feed-forward PID control scheme

In particular, the value of the parameters α_P and α_D that affect the gains are different for the windows in which the Feed-Forward (FF) contribution is non-null (namely, for the nominal maneuver time) and the rest of the time. This gain scheduling prevent the undesired effect of the partial neutralization of the FF caused by a too-strong control action coming from the derivative part of the PD, that is instead required in the steady-state fast disturbance rejection. The gains are optimized using a PSO algorithm, with the objective to minimize the mean quadratic error during the maneuver and in the steady-state inertial pointing. The optimal value of the parameters α_P and α_D are reported in Table 5.4.

	Nominal Slew	**Steady State**
$\boldsymbol{\alpha_P}$	0.8087	0.0781
$\boldsymbol{\alpha_D}$	0.0483	0.4419

Table 5.4: Feed-Forward PD optimized gains

From the 3D representation of the simulated trajectory, compared with the guidance in Fig. 5.12, and from the analysis of the pointing error depicted in Fig. 5.13 it is possible to find out the ability of the designed control strategy to track the reference trajectory and to counteract the unpredicted disturbances and non-modeled dynamics.

More in detail, Fig. 5.13 shows the effect of the ADCS frequency on the pointing error: it is evident that this parameter is fundamental to contains the error especially in the feed-forward aided slew. The control action, reported in Fig. 5.14, shows a good fidelity of the actuated RWs status with respect to the one predicted by the guidance.

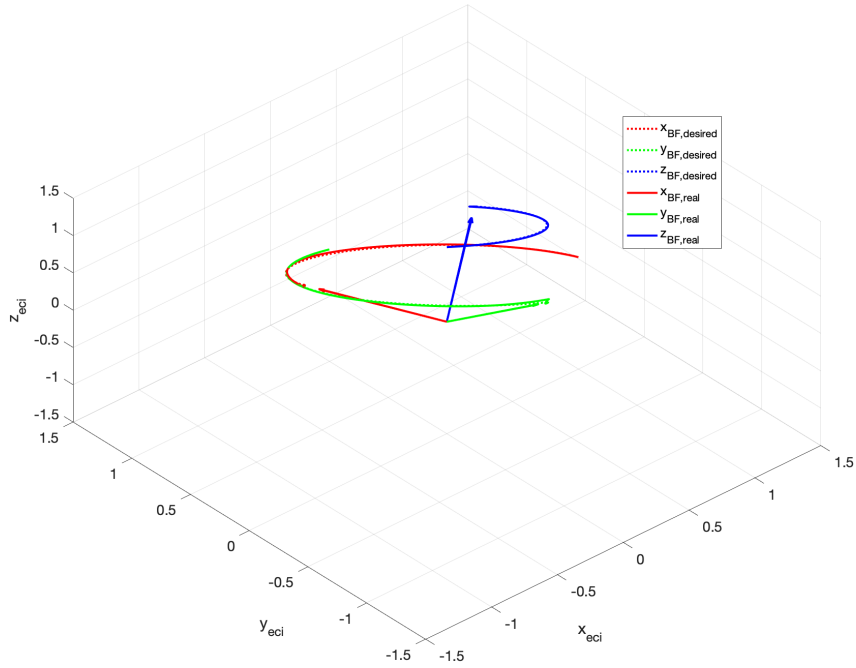


Figure 5.12: Trajectory of the FFPD controlled slew - control frequency = 10 Hz

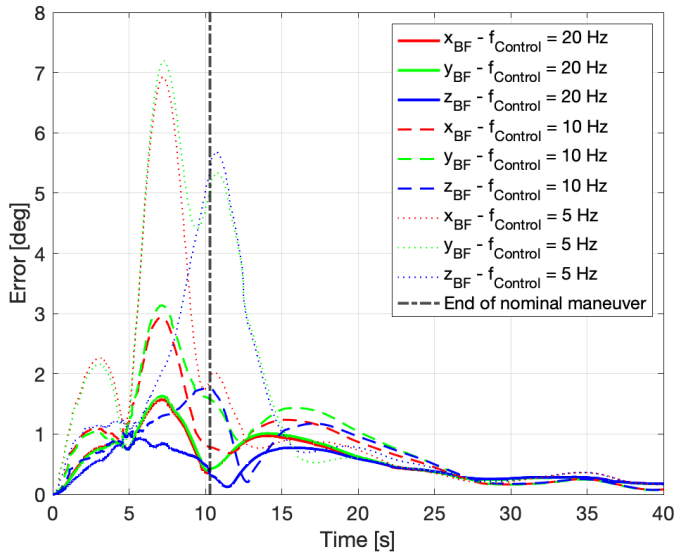


Figure 5.13: Angular error during Feed-Forward PD controlled slew considering different control frequencies

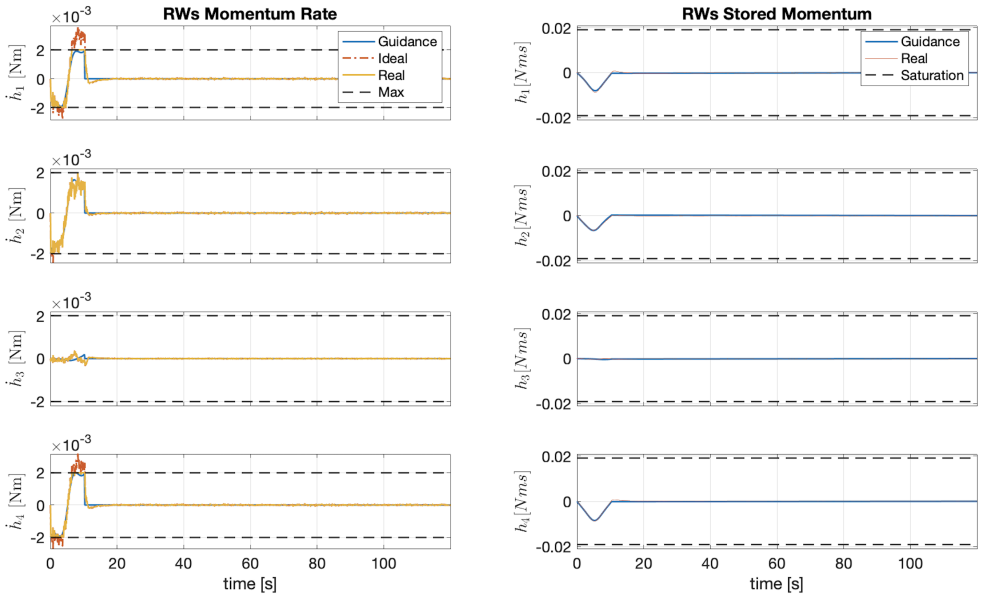


Figure 5.14: Comparison of nominal, ideal and real control actions during the slew maneuver - Feed Forward PD

5.2.4 Control Techniques Comparison

To compare the performances of the three developed control strategies, the time responses have been analyzed looking at four different parameters:

- Mean Quadratic Error during the *Nominal Maneuver*. This phase, highlighted in red in Fig. 5.15, lasts exactly as predicted from the guidance.
- Mean Quadratic Error during the *Transient Phase*. This phase, highlighted in purple in Fig. 5.15, starts at the end of the nominal phase, and lasts until the error decrease below the Mean Quadratic Error of the steady-state phase.
- Mean Quadratic Error during the *Steady-State Phase*.This phase, highlighted in green in Fig. 5.15, starts at the end of the complete (nominal and transient) maneuver.
- *Transient phase duration*

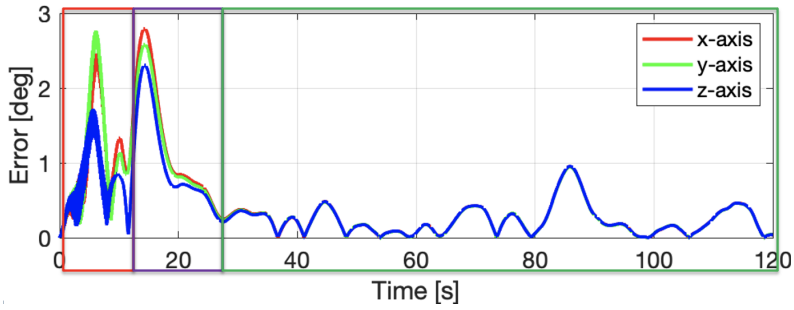


Figure 5.15: Identification of the phases during the time response: Nominal maneuver (red), Transient Phase (purple), Steady-State (green)

From the analysis of the results it is evident that the usage of a simple PD controller is sufficient to guarantee a small steady-state error, with respect to the determination uncertainties reported in Table 5.2. On the other hand, it is not possible to ensure an effective reference tracking, with a small pointing error during both the nominal and transient maneuver phases with the PD controller only, an this could be a critical element in some missions. The inclusion of a FF action is able to dramatically decrease the pointing error in both phases, with a small increase of the on-board ADCS workload. In this last case, the higher is the ratio between the scale frequency of the maneuver and the ADCS frequency, the lower is the error accumulated during the nominal maneuver and transient phase, due to the smaller amount of delay in the FF action.

	Pointing Error				
	ADCS Freq.	**Nominal Maneuver**	**Transient Phase**	**Steady-State**	**Transient Duration**
OL	20 Hz	1.67°	N/A	up to 180°	N/A
	10 Hz	2.94°	N/A	up to 180°	N/A
	5 Hz	5.54°	N/A	up to 180°	N/A
PD	20 Hz	29.66°	11.78°	0.39°	19.2 s
	10 Hz	27.88°	12.72°	0.38°	20.3 s
	5 Hz	28.40°	13.36°	0.38°	19.7 s
FFPD	20 Hz	1.45°	1.04°	0.41°	17.1 s
	10 Hz	2.35°	1.47°	0.39°	17.3 s
	5 Hz	4.97°	3.04°	0.40°	18.1 s

Table 5.5: Control technique comparison: Mean Quadratic Errors and transient phases duration

5.2.5 Effect of model uncertainties on the control techniques

In this section the effect of uncertainties in the spacecraft inertia properties on the pointing error is investigated. In particular, each component of the inertia matrix $\boldsymbol{I}_{unc}$ is affected by a normally distributed uncertainty with a given variance that is scaled with the maximum component of the nominal inertia matrix, as in Eq. 5.8

$$\boldsymbol{I}_{unc} = \boldsymbol{I}_{rigid}^{global} + \delta\boldsymbol{I} = \boldsymbol{I}_{rigid}^{global} + \sigma \max\left(\boldsymbol{I}_{rigid}^{global}\right)\boldsymbol{R} \tag{5.8}$$

in which $\boldsymbol{R}$ is a three by three matrix of normally distributed random number with a unitary variance. In the example here presented, three cases have been considered:

- *Nominal Scenario.* In this scenario the inertia of the spacecraft is exactly known.
- *Precise Inertia model.* In this scenario the Inertia of the spacecraft considered for the guidance is the ideal one, while in the control loop an uncertainty of 1% is considered, according to the error model described in Eq. 5.8.
- *Rough Inertia model.* In this scenario the Inertia of the spacecraft considered for the guidance is the ideal one, while in the control loop an uncertainty of 5% is included, adopting the error model described in Eq. 5.8.

For the nominal scenario, only one run is considered, while a reduced size Montecarlo analysis with 50 runs per scenario is performed for the other two cases. For all the scenarios the ADCS frequency is 10 Hz.

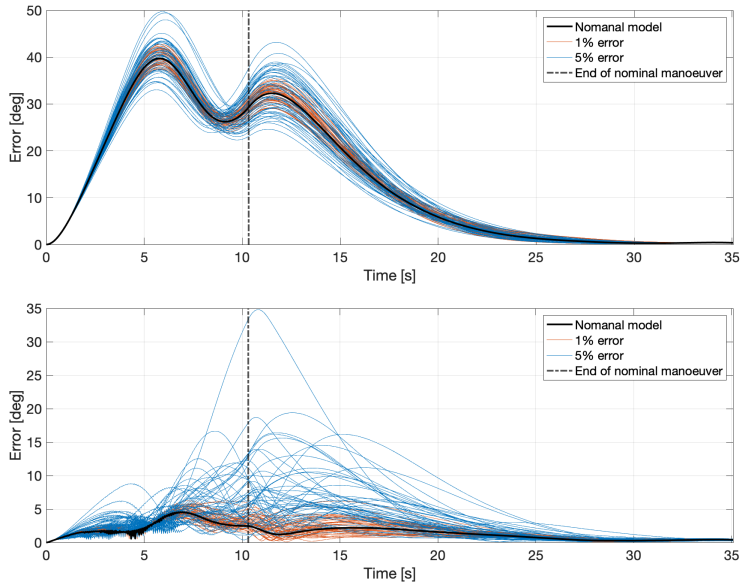


Figure 5.16: Pointing error evaluation via Montecarlo analysis for the PD controller (*upper picture*) and the FFPD controller (*lower picture*)

		Pointing Error		
	Error (3 σ)	**Nominal Maneuver**	**Transient Phase**	**Steady-State**
PD	0	27.88°	12.72°	0.38°
	1%	28.01°	12.26°	0.38°
	5%	27.89°	11.92°	0.39°
FFPD	0	2.35°	1.47°	0.39°
	1%	2.54°	1.59°	0.39°
	5%	3.74°	3.70°	0.39°

Table 5.6: Mean quadratic pointing error variation adopting an uncertainty in the inertia model

The results of the Montecarlo analysis are summarized in Table 5.6, while the pointing error behavior is depicted in Fig. 5.16; from the analysis of these results it is possible to point out some important conclusion, in particular:

- As predicted from theory the PD controller is less influenced by the model uncertainties, indeed the only remarkable effect on the pointing error is that the higher is the uncertainty on the inertia tensor, the higher is the dispersion of the curves, as can be seen in Fig. 5.16. This effect is due to the fact that the gains of the controller are tuned using the nominal inertia, and therefore once they are applied to the real system, with its uncertainties, the time response is different, even if the steady-state error is almost unchanged.
- The effect of the model uncertainty on the FFPD controller is higher and more complex; it can be conceptually divided in two phases:

  1. During the nominal maneuver phase (namely for $t \leq 10.32s$), the FF part of the controller, being based on an ideal model of the spacecraft with no uncertainties, induces an higher error with respect to the reference, that the PD part of the controller is only partially able to counteract. this fact is evident both in Fig. 5.16 and looking at the mean quadratic pointing error during the nominal maneuver reported in Table. 5.6, which grows by about 8% for an uncertainty of 1% (at 3σ) and by about 59% for an uncertainty of 5% (at 3σ)
  2. As consequence, the pointing error, as well as the difference between the actual angular velocity and the reference one, at the end of the nominal maneuver are higher with respect to the case with the ideal model . This fact leads to a longer transient phase, characterized by a considerably higher mean quadratic pointing error.

  As for the simple PD controller, the steady state pointing error is almost unchanged with respect to the ideal case.

Even if, looking at the mean quadratic values, the developed FFPD controller shows an acceptable robustness for practical purposes, it is important to underline that there are few exceptions with high errors at the end of the nominal maneuver that can lead to relevant violations of the pointing constraints.

CHAPTER 6

Flexible Dynamics

The aim of this section is to describe the developed multi-body model [42] suitable for CubeSat. In particular, the modeling aims to simulate the dynamics of a CubeSat including the motion of the most relevant flexible element embarked on such class of satellites.

6.1 multi-body modeling

Looking at typical CubeSat configurations [1], it is possible to identify three different main elements:

- **Central Body.** The central body of a CubeSat is a compact element with a regular cuboid shape. The size of the cuboid define the class of the CubeSat, and in general it is a multiple of the so-called $1U$ element: a 10cm by 10cm cube. The most common form factors range from $1U$ to $12U$, as shown in Fig.6.1. The structure of the central body is typically very rigid, with natural frequency larger than 100 Hz [1], extremely higher with respect to the ones exited by attitude and orbital motion, as well as natural perturbations, and therefore it will be considered as a rigid body in the here presented modeling.

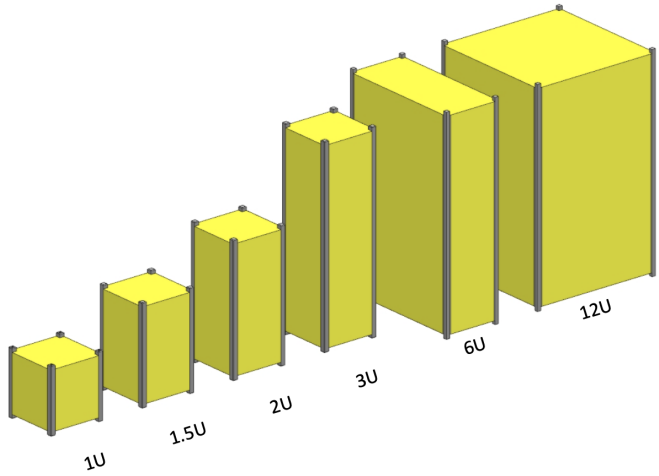


Figure 6.1: Typical CubeSat form factors - Credits of Cal Poly – San Luis Obispo, CA .

Among all the subsystems and components contained in the central body, only two are of interest for the multi-body modeling of the whole CubeSat, namely the the thrusters and momentum exchange devices. The thrusters embarked by CubeSats use solid of gaseous propellant, therefore the motion of this class of satellites is not affected by the sloshing effects, that are of primary importance in larger satellites equipped with liquid propellant thrusters [89]. Concerning the momentum exchange devices, CubeSats are typically equipped with RWs, and the dynamics of these elements shall be included in the modeling of the central body.

- **Solar Panels.** Even if CubeSat can be equipped with body mounted solar panels, in most of the cases the power budget of the platforms exceeds the capability of these units, indeed the available external surface, already limited by the small size of the satellites, it is further reduced by the necessary presence of payloads and sensors that need to be mounted on these surfaces. Furthermore, the positioning of the panels on the external surfaces is not optimal, indeed it is impossible to orient all the panels at the same time towards the Sun. For these reasons the simpler CubeSats are equipped with body mounted solar panels only: the most of the platforms are equipped with deployable solar arrays. The size of these element is highly variable, and can reach remarkably large values, if compared to the size of the central body, especially in the presence of electric thrusters or payloads that are particularly demanding in terms of power, leading to designs with multiple folded panels. In this last case, the most common technical solution consists of a sequence of panels (typically two or three on each side) connected by hinges and

rotational springs. This design, that is necessary to contains the size of the CubeSat in the folded configuration within the stringent requirements of the commercial CubeSat deployers, can be source of vibration in the low frequency spectrum, that can affect the pointing performances of the CubeSats.

- **Antennas.** To ensure the communication with ground and between satellites, three possible solutions are adopted:
  - *Patch antennas*: typically used for directional S-band or X-band communications. These elements, that are patched on the external surfaces of the spacecraft, are not source of vibrations in the low spectrum, and therefore can be considered as a part of the central rigid hub.
  - *Deployable 'wire' antennas*: typically used for UHF/VHF omni-directional communication. These elements are folded in the central hub during the launch, and are deployed at the commissioning of the satellite. These elements are extremely flexible, and even if their masses are small with respect to the total mass of the spacecraft, it is interesting to investigate the flexible dynamics of such element since their natural frequencies, that are very low, fall in the band exited by the spacecraft dynamics and control.
  - *Deployable High Gain Antennas*: this is the solution used for the first interplanetary CubeSat, MarCO from NASA [90]. It works the X-band and it is characterized by a considerably higher gain (29 dB vs 6-7 dB of typical S-band pathed antennas) with respect to the other solution, enabling interplanetary communications. Future CubeSats that will fly outside the LEO environment will surely implement this solution for the high gain communications. From a structural point of view, it is extremely similar to the solar panels, being composed by rigid plates interconnected by hinges and springs.

In the following sections, the mathematical model of each element is presented and the EoM for the multi-body model of the CubeSat are derived using a Lagrangian approach [45, 91].

6.1.1 Central body

The central body of a CubeSat, as introduced before, is considered as a rigid body, subjected by the external perturbation torques $\sum \boldsymbol{M}_{external}(\mathbf{q}, \boldsymbol{\omega}, t)$, that depend on the state of the spacecraft and on the epoch, and to the control torque $\boldsymbol{M}_{control}$, that for a RWs assembly is given by Eq. 6.1 , as reported in Sec. 4.2.

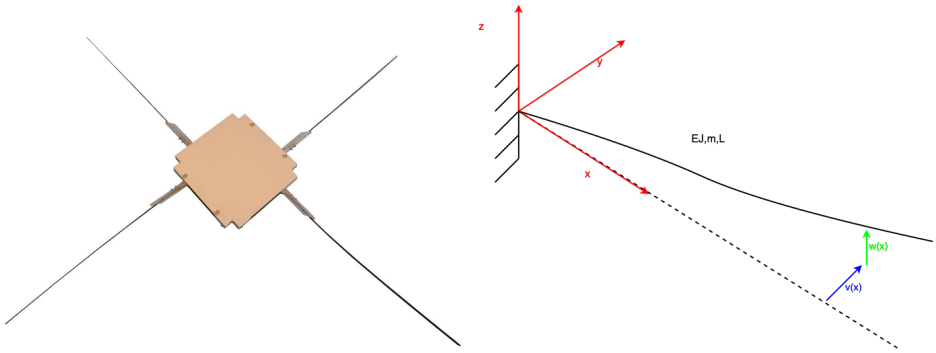
$$\boldsymbol{M}_{control} = -\left(\boldsymbol{A}_{RWs}^{assembly}\dot{\boldsymbol{h}}_r + \boldsymbol{\omega} \times A_{RWs}^{assembly}\boldsymbol{h}_r\right) \tag{6.1}$$

The kinetic energy for the rigid body, considering the attitude of motion only, body reads as in Eq. 6.2, and no source of potential energy is present.

$$T = \frac{1}{2}\boldsymbol{\omega}^{\boldsymbol{T}}\boldsymbol{I}_{\boldsymbol{b}}\boldsymbol{\omega} = \frac{1}{2}\begin{bmatrix} p & q & r \end{bmatrix}\begin{bmatrix} I_{x,x} & I_{x,y} & I_{x,z} \\ I_{y,x} & I_{y,y} & I_{y,z} \\ I_{z,x} & I_{z,y} & I_{z,z} \end{bmatrix}\begin{bmatrix} p \\ q \\ r \end{bmatrix} \tag{6.2}$$

6.1.2 Booms

The antenna booms are modeled with the Eulero-Bermoulli beam theory [44], and the peculiar structure of the antenna booms requires a DPM in order to develop a more faithful model, that is able to replicate the real dynamics of these elements. Figure 6.2 shows the modeling of each boom, with its local RF, and in particular $u(x,t)$, $v(x,t)$ and $w(x,t)$ are the elastic displacement of each point of the boom with respect to the **x**, **y** and **z** axis. The developed flexible model of the booms considers only small decoupled elastic displacements but large overall rigid motion.



(a) Typical UHF/VHF for CubeSat. Credits: ISISpace.

(b) displacements in Local Reference Frame.

Figure 6.2: Antenna booms modeling

The kinetic energy of the system composed by n_{ant} antennas is the sum of the kinetic energy of each boom, as in Eq. 6.3 and Eq. 6.4, in which $\boldsymbol{v_p}(x,t)$ is the velocity of each point of the boom.

$$T_{ant} = \sum_{i=1}^{n_{ant}} T_{ant,i} \tag{6.3}$$

$$T_{ant,i} = \frac{1}{2}\int_0^l m\boldsymbol{v_p}^2 dx \tag{6.4}$$

By definition, the velocity of each point can be computed taking the first time derivative of the position of the point, as in Eq. 6.5, in which $\boldsymbol{x}_{rigid}$ is the position pf the generic point P in the BF RF, and $\boldsymbol{s}_{flex}$ is the flexible displacement with respect to the undeformed configuration.

$$\boldsymbol{v}_p = \frac{d}{dt}\boldsymbol{x}_p = \frac{d}{dt}\left(\boldsymbol{x}_{rigid} + \boldsymbol{s}_{flex}\right) = \dot{\boldsymbol{s}}_{flex} + \boldsymbol{\omega} \times \left(\boldsymbol{x}_{rigid} + \boldsymbol{s}_{flex}\right) \tag{6.5}$$

following the hypothesis of small elastic displacement, namely $\boldsymbol{s}_{flex} \ll \boldsymbol{x}_{rigid}$, and neglecting the velocity of the origin of the BF RF, Eq. 6.5 can be simplified as shown in Eq . 6.6

$$\boldsymbol{v}_p = \dot{\boldsymbol{s}}_{flex} + \boldsymbol{\omega} \times \left(\boldsymbol{x}_{rigid}\right) = \boldsymbol{\omega} \times (P - G) + \dot{\boldsymbol{s}}_{flex} \tag{6.6}$$

Since the elastic displacement $\boldsymbol{s}_{flex}(x,t)$ is a continuous function of space and time, a spatial discretization is needed to reduce the infinite degrees of freedom of the resulting EoM to a finite, and possibly small, number. To do that, a RG approach is adopted in the discretization of the elastic behavior of the structure, as shown in eq. 6.7, in which the matrix of the trials function $\mathbf{N}(\mathbf{x})$ is used to discretize in space the motion of the beams with respect to the temporal coordinates $\boldsymbol{u}(t)$. It is important to underline that, since the elastic displacements and the consequent elastic deformations are small, the flexible motion in the $\mathbf{x}$ and $\mathbf{y}$ directions are fully decoupled, leading to a diagonal matrix of the trials function $\mathbf{N}(\mathbf{x})$.

$$\boldsymbol{s}(\boldsymbol{x},t)_{flex}^{LocalRF} = \begin{bmatrix} u(x,t) \\ v(x,t) \\ w(x,t) \end{bmatrix} = \boldsymbol{N}(\boldsymbol{x})\boldsymbol{u}(t) = \begin{bmatrix} 0 & 0 & 0 \\ 0 & N_y(x) & 0 \\ 0 & 0 & N_z(x) \end{bmatrix} = \begin{bmatrix} u_x(t) \\ u_y(t) \\ u_z(t) \end{bmatrix} \tag{6.7}$$

Given the RG discretization, it is possible to write the velocity and the deformation of the Eulero-Bermoully beam as function of the temporal coordinates $\boldsymbol{u}(t)$ as shown in Eq. 6.8.

$$\begin{cases} \dot{\boldsymbol{s}}(\boldsymbol{x},t)_{flex} = \boldsymbol{A_{b,i}}\boldsymbol{N}(\boldsymbol{x})\dot{\boldsymbol{u}}(t) \\ \frac{\partial}{\partial x}(\boldsymbol{s}(\boldsymbol{x},t)_{flex}) = \boldsymbol{A_{b,i}}\frac{\partial \boldsymbol{N}(\boldsymbol{x})}{\partial x}\boldsymbol{u}(t) = \boldsymbol{A_{b,i}}\boldsymbol{N}(\boldsymbol{x})_{/x}\boldsymbol{u}(t) \end{cases} \tag{6.8}$$

The RG discretization shall be applied to the kinetic energy of the antennas, as in Eq. 6.9.

$$\begin{aligned} T_{ant,i} &= \frac{1}{2}\int_0^l m(\boldsymbol{\omega}\times(P-G)+\dot{\boldsymbol{s}}_{flex})^2 dx \\ &= \frac{1}{2}\int_0^l m(\boldsymbol{\omega}\times(P-G)+\boldsymbol{A_{b,i}}\boldsymbol{N}(\boldsymbol{x})\dot{\boldsymbol{u}}(t))^2 dx \end{aligned} \tag{6.9}$$

The kinetic energy can be reorganized in order to highlight the dependence of the function on the free coordinates, as reported in 6.10.

$$\begin{aligned} T_{ant,i} = \frac{1}{2}\int_0^l m\left(\boldsymbol{\omega}\times(P-G)\right)^2 dx + \frac{1}{2}\dot{\boldsymbol{u}}(t)^T\int_0^l m\boldsymbol{N}^T(\boldsymbol{x})\boldsymbol{N}(\boldsymbol{x})dx\dot{\boldsymbol{u}}(t)+ \\ + \int_0^l m\boldsymbol{\omega}\times(P-G)\cdot\boldsymbol{A_{b,i}}\boldsymbol{N}(\boldsymbol{x})dx\dot{\boldsymbol{u}}(t) \end{aligned} \tag{6.10}$$

In which it is possible to recognize and isolate the rigid contribution to the motion, as reported in 6.11 and the contribution due to the free coordinates introduced with the discretization, that includes two different therms, the first depending only on $\boldsymbol{u}(t)$ and the second that couples the flexible small motion with the rigid overall one.

$$\boldsymbol{T}_{rigid} = \frac{1}{2}\int_0^l m\left(\boldsymbol{\omega}\times(P-G)\right)^2 dx = \frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{I}_{rigid}^{boom}\boldsymbol{\omega} \tag{6.11}$$

The two terms that depends on the flexible free coordinates $\boldsymbol{u}(t)$ can be conveniently rewritten in a more compact matrix formulation, as shown in Eq. 6.12

$$\begin{cases} \boldsymbol{M}_{\boldsymbol{u},\boldsymbol{u}}^{boom} = \int_0^l m\boldsymbol{N}^T(\boldsymbol{x})\boldsymbol{N}(\boldsymbol{x})dx \\ \boldsymbol{\omega}^T\boldsymbol{M}_{\boldsymbol{w},\boldsymbol{u}}^{boom} = \int_0^l m\boldsymbol{\omega}\times(P-G)\cdot\boldsymbol{A_{b,i}}\boldsymbol{N}(\boldsymbol{x})dx \end{cases} \tag{6.12}$$

Equation 6.13 summarize the matrix formulation of the kinetic energy for the general RG discretization of the booms.

$$T_{ant,i} = \frac{1}{2}\boldsymbol{\omega}^T\boldsymbol{I}_{rigid}^{boom}\boldsymbol{\omega} + \frac{1}{2}\dot{\boldsymbol{u}}^T(t)\boldsymbol{M}_{\boldsymbol{u},\boldsymbol{u}}^{boom}\dot{\boldsymbol{u}}(t) + \boldsymbol{\omega}^T\boldsymbol{M}_{\boldsymbol{w},\boldsymbol{u}}^{boom}\dot{\boldsymbol{u}}(t). \tag{6.13}$$

To compute the elastic potential energy of the antennas, it is necessary to rely on a proper structural model. As mentioned before, due to the assumption of

small decoupled elastic motion the Eulero-Bermoulli model is adopted, therefore the potential energy of the beam can be expressed as in 6.14.

$$V_{elastic}^{booms} = \frac{1}{2}\int_0^l \boldsymbol{s}_{/xx}^T EJ\boldsymbol{s}_{/xx}dx = \frac{1}{2}\boldsymbol{u}^T\int_0^l \boldsymbol{N}_{/xx}^T EJ\boldsymbol{N}_{/xx}dx\boldsymbol{u} \tag{6.14}$$

In order to build up the mass, coupling and stiffness matrices it is necessary to specify the trial functions $\mathbf{N}(\mathbf{x})$ to be used for the spatial RG discretization. The selection of the functions will be driven by the essential boundary conditions, namely null displacements and slopes along $\mathbf{x}$ and $\mathbf{y}$ directions in the joint, as reported in Eq. 6.15.

$$\begin{cases} v(0,t) = w(0,t) = 0 \\ v_{/x}(0,t) = w_{/x}(0,t) = 0 \end{cases} \tag{6.15}$$

while the natural boundary conditions are null bending moment and null shear force at the tip, as reported in Eq. 6.16.

$$\begin{cases} v_{/xx}(l,t) = w_{/xx}(l,t) = 0 \\ v_{/xxx}(l,t) = w_{/xxx}(l,t) = 0 \end{cases} \tag{6.16}$$

The essential boundary conditions shall be applied to the trials functions, $\mathbf{N}(\mathbf{x})$, otherwise the constraints shall not be respected at any instant t, while the natural boundary conditions are optionals, and can be imposed in order to have a more precise representation of the dynamics. Within this work, a fourth-order polynomial comparison trial function able to satisfy both essential and natural boundary conditions is adopted, as reported in 6.17 and Fig. 6.3

$$\begin{cases} N_y(x) = N_z(x) = -\frac{1}{24}\left(\frac{x}{l}\right)^4 + \frac{1}{6}\left(\frac{x}{l}\right)^3 - \frac{1}{4}\left(\frac{x}{l}\right)^2 \\ N_{y/x}(x) = N_{z/x}(x) = -\frac{1}{6}\frac{x^3}{l^4} + \frac{1}{2}\frac{x^2}{l^3} - \frac{1}{2}\frac{x}{l^2} \\ N_{y/xx}(x) = N_{z/xx}(x) = -\frac{1}{2}\frac{x^2}{l^4} + \frac{x}{l^3} - \frac{1}{2}\frac{1}{l^2} \end{cases} \tag{6.17}$$

6.1.3 solar panels

The developed multi-body model for the solar panels, that could be used also for the deployable High Gain Antennas, is depicted in Fig. 6.4. In particular, the solar panels are considered as rigid plates (two for each array in the here presented example); the first plate is connected to the central rigid body with an ideal hinge and a rotational spring, and the second panel is linked to the first one with the same type of constraint. The hinges are characterized by

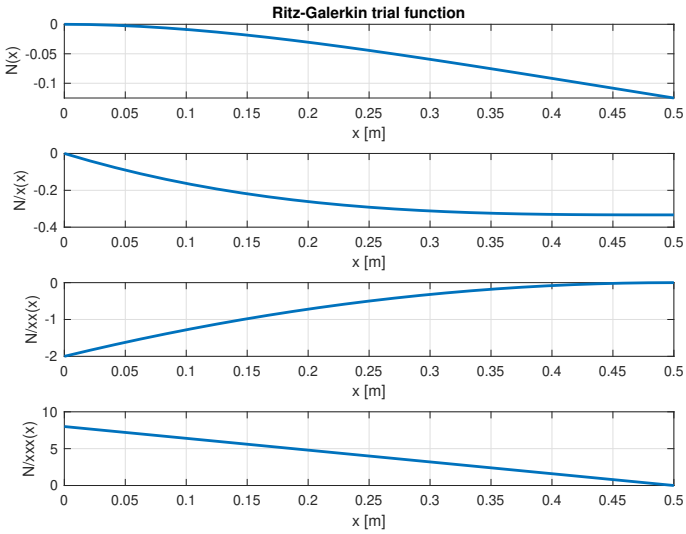
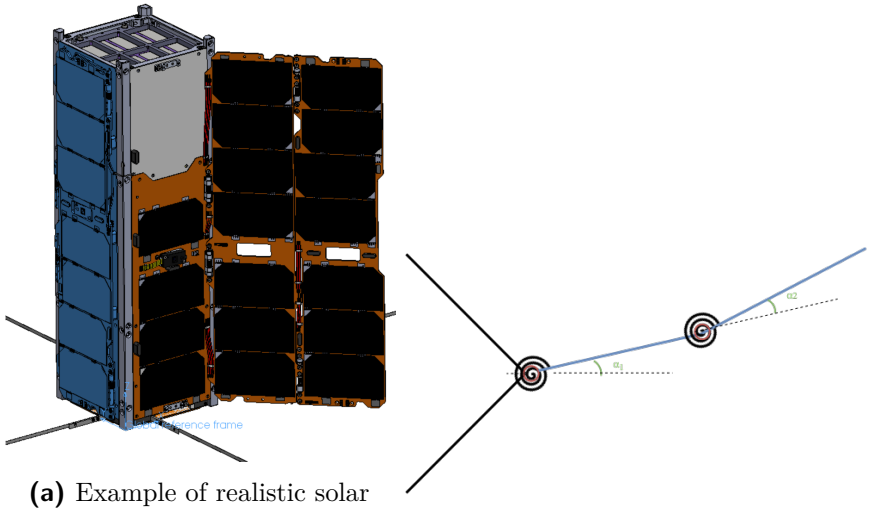


Figure 6.3: Selected RG trial function.

a rotation axis $\mathbf{i}_{hinge}$ and the rotational springs are characterized by a linear behavior and elastic constant k_{spring}. The relative angular displacement of each panel is α_i.



(a) Example of realistic solar panels. Credits: ISISpace.

(b) Multi-body model - Local RF.

Figure 6.4: Solar panels modeling

The kinetic energy of the solar arrays is the sum of the kinetic energy of each solar array, as in Eq. 6.18

$$T_{SA} = \sum_{i=1}^{n_{SA}} T_{SA,i} \tag{6.18}$$

in which the kinetic energy is the sum of the kinetic energy of each single panel, considered as a rigid body with barycentric velocity $\boldsymbol{v}_{C,j}$ and angular velocity relative to the rigid central hub $\dot{\alpha}_m \boldsymbol{i}_{hinge}$, in which $\boldsymbol{i}_{hinge}$ is the direction of the hinge in the BF RF.

$$T_{SA,i} = \sum_{j=1}^{n_{panels}} \left[\frac{1}{2} m \boldsymbol{v}_{C,j}^2 + \frac{1}{2} \left(\boldsymbol{\omega} + \sum_{m=1}^{j} \dot{\alpha}_m \boldsymbol{i}_{hinge} \right)^T \boldsymbol{I}_{SP,j} \left(\boldsymbol{\omega} + \sum_{m=1}^{j} \dot{\alpha}_m \boldsymbol{i}_{hinge} \right) \right] \tag{6.19}$$

The barycentric velocity of each panels can be computed by means of consecutive rigid motion act, as in Eq. 6.20.

$$\begin{aligned} \boldsymbol{v}_{C,j} = \boldsymbol{\omega} \times (H_1 - G) + \left(\boldsymbol{\omega} + \sum_{m=1}^{j-1} \dot{\alpha}_m \boldsymbol{i}_{hinge} \right) \times (H_j - H_{j-1}) + \\ + \left(\boldsymbol{\omega} + \sum_{m=1}^{j} \dot{\alpha}_m \boldsymbol{i}_{hinge} \right) \times (C_j - H_j) \end{aligned} \tag{6.20}$$

If only small elastic displacement are considered for the rotational springs, the kinetic energy of the solar arrays can be rewritten in a more compact matrix formulation, as in Eq. 6.21, in which $\boldsymbol{I}_{\boldsymbol{\alpha\alpha}}$ is the inertia term related with the small elastic angular displacement, $\boldsymbol{I}_{rigid}^{SA}$ is the inertia term due to the contribution of the rigid motion of the hub and $\boldsymbol{I}_{\boldsymbol{\omega\alpha}}$ is the coupling term between the rigid motion and the elastic angular displacement.

$$T_{SA,i} = \frac{1}{2} \boldsymbol{\omega}^T \boldsymbol{I}_{rigid}^{SA} \boldsymbol{\omega} + \frac{1}{2} \dot{\boldsymbol{\alpha}}^T \boldsymbol{I}_{\boldsymbol{\alpha\alpha}} \dot{\boldsymbol{\alpha}} + \frac{1}{2} \boldsymbol{\omega}^T \boldsymbol{I}_{\boldsymbol{\omega\alpha}} \dot{\boldsymbol{\alpha}} \tag{6.21}$$

The potential energy stored in the rotational springs can be written as in Eq. 6.22.

$$V_{elastic}^{SA} = \sum_{j=1}^{n_{panels}} \left(\frac{1}{2} k_{spring} \alpha_i^2 \right) \tag{6.22}$$

6.1.4 EoM

The EoM can be computed by means of the equation of Lagrange. In particular, due to the conservative behavior of the elastic elements, the Lagrangian function reads as in Eq. 6.23.

$$L = T - V \tag{6.23}$$

In which the kinetic energy is the sum of the kinetic energy of each component modeled: the central rigid hub with the contribution of the RWs, the solar arrays, and the antennas, as in Eq. 6.24

$$T = T_{centralbody} + \sum_{i=1}^{n_{RWs}} T_{RWs,i} + \sum_{i=1}^{n_{SA}} T_{SA,i} + \sum_{i=1}^{n_{ant}} T_{ant,i} \tag{6.24}$$

and the potential energy of the system includes only the elastic contribution of the antenna booms and the rotational springs of the hinges of the solar panels, a per Eq. 6.25.

$$V = \sum_{i=1}^{n_{SA}} V_{elastic,SA,i} + \sum_{i=1}^{n_{ant}} V_{elastic,ant,i} \tag{6.25}$$

The Lagrange equation in the pseudo-coordinates ω, reported in Eq. 6.26, leads to the EoM that describes the rigid motion of the spacecraft, as in Eq. 6.27.

$$\frac{d}{dt}\left[\frac{\partial}{\partial \boldsymbol{\omega}} T\right] + \boldsymbol{\omega} \times \frac{\partial}{\partial \boldsymbol{\omega}} T = \sum \boldsymbol{M_{external}} + \boldsymbol{M}_{control} \tag{6.26}$$

$$\boldsymbol{I}_{rigid}^{global} \dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times \boldsymbol{I}_{rigid}^{global} \boldsymbol{\omega} = \sum \boldsymbol{M}_{external} + \boldsymbol{M}_{control} + \boldsymbol{M}_{booms} + \boldsymbol{M}_{SA} \tag{6.27}$$

In this EoM, the external torques provoked by the environmental perturbation and by any other *external* attitude control device able to exchange momentum with the surrounding environment, such as magnetic actuators or thrusters, are included in the vector of the external actions $\sum \boldsymbol{M_{external}}$. On the other hands, the control action imposed by *internal* momentum exchange device (namely the RWs) is grouped in the control vector $\boldsymbol{M}_{control}$ for the sake of convenience and clarity. The definition of the *internal* control torque, that belong directly from the Lagrange equation in the pseudo-coordinate 1ω, is reported in Eq.6.28, in which the term $\boldsymbol{A}_{RWs}^{assembly}$ is the assembly matrix of the RWs, specifying the BF mounting direction of each reaction wheel, as in Eq. 6.29.

$$\boldsymbol{M}_{control} = -\left(\boldsymbol{A}_{RWs}^{assembly}\dot{\boldsymbol{h}}_r + \boldsymbol{\omega} \times A_{RWs}^{assembly}\boldsymbol{h}_r\right) \tag{6.28}$$

$$\boldsymbol{A}_{RWs}^{assembly}\boldsymbol{h}_r = \sum_{i=1}^{n_{RWs}} \boldsymbol{h}_{RW,i} = \sum_{i=1}^{n_{RWs}} \boldsymbol{I}_{RW,i}\boldsymbol{\omega}_{RW,i} \tag{6.29}$$

The other terms present in the right-hand side of Eq. 6.27, namely $\boldsymbol{M}_{booms}$ and $\boldsymbol{M}_{SA}$, represent the influence of the flexible elements on the overall rigid motion, and are defined in Eq.6.30.

$$\begin{cases} \boldsymbol{M}_{booms} = -\left(\boldsymbol{M}_{\boldsymbol{u},\boldsymbol{u}}^{booms}\ddot{\boldsymbol{u}} + \boldsymbol{\omega} \times \boldsymbol{M}_{\boldsymbol{\omega},\boldsymbol{u}}^{booms}\dot{\boldsymbol{u}}\right) \\ \boldsymbol{M}_{SA} = -\left(\boldsymbol{I}_{\boldsymbol{\alpha},\boldsymbol{\alpha}}^{SA}\ddot{\boldsymbol{\alpha}} + \boldsymbol{\omega} \times \boldsymbol{I}_{\boldsymbol{\omega},\boldsymbol{\alpha}}^{booms}\dot{\boldsymbol{\alpha}}\right) \end{cases} \tag{6.30}$$

As can be seen from Eq..6.30, to compute the torques provoked by the flexible behavior of antennas and solar panels, it is necessary to know the dynamics of these disturbances. The most practical and effective way to write the EoM for the disturbances is to take advantage of the Lagrange equation in the coordinates $\mathbf{u}$ and α, as in Eq.6.31

$$\begin{cases} \frac{d}{dt}\left[\frac{\partial}{\partial\dot{\mathbf{u}}}L\right] - \frac{\partial}{\partial\mathbf{u}}L = 0 \\ \frac{d}{dt}\left[\frac{\partial}{\partial\dot{\boldsymbol{\alpha}}}L\right] - \frac{\partial}{\partial\boldsymbol{\alpha}}L = 0 \end{cases} \tag{6.31}$$

The solution of the Lagrangian equation leads to the dynamic EoM of the flexible disturbances up to solar panels and antennas, as in Eq. 6.32

$$\begin{cases} \boldsymbol{M}_{\boldsymbol{u},\boldsymbol{u}}^{booms}\ddot{\boldsymbol{u}} + \boldsymbol{C}_{\boldsymbol{u},\boldsymbol{u}}^{booms}\dot{\boldsymbol{u}} + \boldsymbol{K}_{\boldsymbol{u},\boldsymbol{u}}^{booms}\boldsymbol{u} = -\boldsymbol{M}_{\boldsymbol{u},\boldsymbol{\omega}}\dot{\boldsymbol{\omega}} \\ \boldsymbol{I}_{\boldsymbol{\alpha},\boldsymbol{\alpha}}^{SA}\ddot{\boldsymbol{\alpha}} + \boldsymbol{C}_{\boldsymbol{\alpha},\boldsymbol{\alpha}}^{SA}\dot{\boldsymbol{\alpha}} + \boldsymbol{K}_{\boldsymbol{\alpha},\boldsymbol{\alpha}}^{SA}\boldsymbol{u} = -\boldsymbol{I}_{\boldsymbol{\alpha},\boldsymbol{\omega}}\dot{\boldsymbol{\omega}} \end{cases} \tag{6.32}$$

In which $\mathbf{u}(t)$ are the temporal degrees of freedom related to the RG discretization of the flexible booms and α are the relative angular displacements of the solar panels. Therefore $\boldsymbol{M}_{\boldsymbol{u},\boldsymbol{u}}$ and $\boldsymbol{I}_{\boldsymbol{\alpha},\boldsymbol{\alpha}}^{SA}$ are the mass and inertia matix related to these degrees of freedom, $\boldsymbol{K}_{\boldsymbol{\alpha},\boldsymbol{\alpha}}^{SA}$ and $\boldsymbol{K}_{\boldsymbol{u},\boldsymbol{u}}^{booms}$ are the stiffness matrix and $\boldsymbol{M}_{\boldsymbol{u},\boldsymbol{\omega}}$ and $\boldsymbol{I}_{\boldsymbol{\alpha},\boldsymbol{\omega}}$ are the coupling matrices between flexible and rigid motion, as stated in Sec. 6.1.3 and Sec. 6.1.2 respectively. In order to replicate the behavior of real structures, in which a damping effect is always present, even if very often of modest entity, the damping matrices $\boldsymbol{C}_{\boldsymbol{u},\boldsymbol{u}}^{\boldsymbol{booms}}$ and $\boldsymbol{C}_{\boldsymbol{\alpha},\boldsymbol{\alpha}}^{\boldsymbol{SA}}$ have been added to the EoM of the booms and solar panels flexible disturbances. Damping in distributed parameter structures is never easy to be identified, and therefore it was decided to use a proportional damping model [44], as in Eq.6.33, calibrating the proportionality parameters with respect to the mass

and stiffness matrices appropriately, as will be explained in more detail in Sec. 6.2

$$\begin{cases} \boldsymbol{C}_{\boldsymbol{u},\boldsymbol{u}}^{\boldsymbol{booms}} = \epsilon_{1,u}\boldsymbol{M}_{\boldsymbol{u},\boldsymbol{u}}^{booms} + \epsilon_{2,u}\boldsymbol{K}_{\boldsymbol{u},\boldsymbol{u}}^{booms} \\ \boldsymbol{C}_{\boldsymbol{\alpha},\boldsymbol{\alpha}}^{\boldsymbol{SA}} = \epsilon_{1,\alpha}\boldsymbol{I}_{\boldsymbol{\alpha},\boldsymbol{\alpha}}^{SA} + \epsilon_{2,\alpha}\boldsymbol{K}_{\boldsymbol{\alpha},\boldsymbol{\alpha}}^{SA} \end{cases} \tag{6.33}$$

6.2 Test Cases

The multi-body model developed is implemented in MATLAB-Simulink 2021b, and tested in this section following the maneuver described in Sec. 5.2.

6.2.1 Rest-to-rest fast slew

In this section the fast rest-to-rest maneuver described in Sec. 5.2 consisting of a time-optimal slew of 170° around the axis $\mathbf{i}_{rot} = [1, 1, 2]$ is adopted to test the multi-body model of the HERMES CubeSat. The CubeSat, that is depicted in Fig. 5.1, has properties identical to the rigid one, summarized in Eq. 5.1, Eq. 5.2 and Table 5.2 and discussed in Sec. 5.2, with the exception of the antenna booms and the flexible solar panels. In particular, the Solar Panels are modeled with the multi-body approach presented in Sec. 6.1.3 with the mass, stiffness and damping parameters reported in Table 6.1.

Configuration	2 solar arrays composed by 2 panel (Fig. 6.4)
solar panel size	0,3m x 0,1m each panel
Mass	0,150 kg per panel (uniformly distributed)
Torsional Stiffness	0.0975 Nmm/deg
Damping Mass proportional coef.	0.07
Damping Stiffness proportional coef.	0.001

Table 6.1: Solar arrays flexible model parameters

The antenna booms are modeled with the RG approach described in Section 6.1.2, using the physical parameter reported in Tab. 6.2.

The values of the mass and stiffness proportional damping coefficient for both the solar panels and booms come from the analysis of the free time response of the systems, in particular those values are selected to ensure a settling time of 5 s, quite representative of real structures. With the above mentioned parameter, it is easy to verify that the discretized dynamical system presented in Eq. 6.32 leads to a natural frequency for the free oscillation for the booms

Configuration	four booms (Fig. 6.2)
Length	50 cm
Mass per unit length	$0.0047\,\mathrm{kg\,m^{-1}}$
Material	Ni-Ti alloy: E = 85 GPa
Cross Section	Circular: d = 1 mm
Damping Mass Proportional coefficient	0.05
Damping Stiffness Proportional coefficient	0.03

Table 6.2: Booms flexible model parameters

equal to $f_{Booms} = 2.04Hz$ and for the solar panels equal to $f_{SA,1} = 3.22Hz$, while the second frequency of the solar panels is $f_{SA,21} = 10.8Hz$

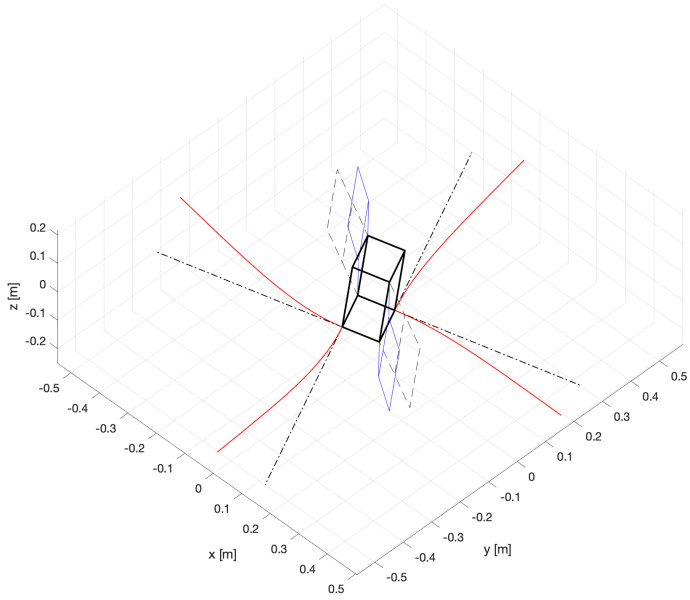


Figure 6.5: Comparison of rigid and flexible configuration during the slew ($t = 3s$). Elastic displacement enlarged for sake of visibility

The dynamical response of the system can be better appreciate in Fig. 6.6, in which the displacement of the booms tips and the angular displacement of the solar panels during the maneuver and in the steady-state inertial pointing are reported.

In particular, comparing the deflection with the angular acceleration and velocity behaviors in Fig. 6.7 it is possible to appreciate the static deflection due to the angular acceleration and the dynamic response.

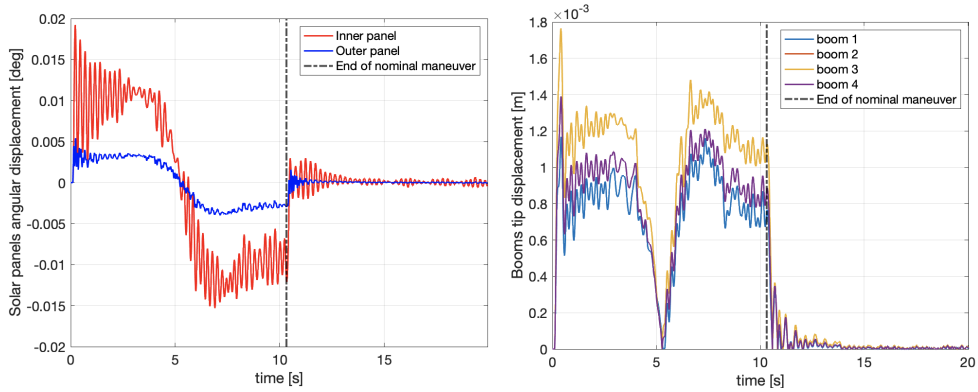


Figure 6.6: Solar panel angular displacements (*left*) and booms tip displacements (*right*).

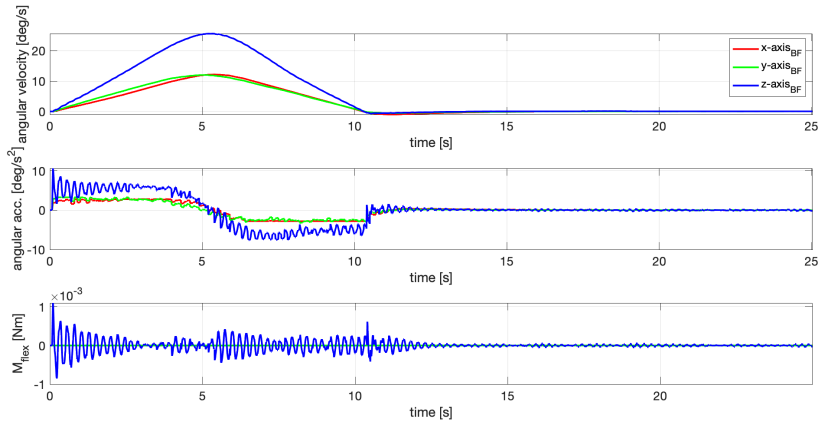


Figure 6.7: Angular velocity(*upper*), angular acceleration (*middle*) and flexible disturbance (*bottom*)

The fact that the flexible torque is almost aligned with the z_{BF} axis is due to the fact that the main contribution arises from the flexible dynamics of the solar panels, that are hinged in that direction. Moreover, it is possible to see that the flexible disturbance of the solar panels during the slew is order of magnitude higher with respect to the external disturbances, but due to its periodic behavior it has a limited effect on the pointing error, as depicted in Fig. 6.8

Looking at the frequency analysis of the flexible disturbance $\boldsymbol{M}_{flex}$ reported in Fig. 6.9 it is possible to see that the fundamental frequencies of the solar panels and of the booms predicted from theory match with a good approximation the peak of the Fast Fourier transform of the flexible disturbance torque.

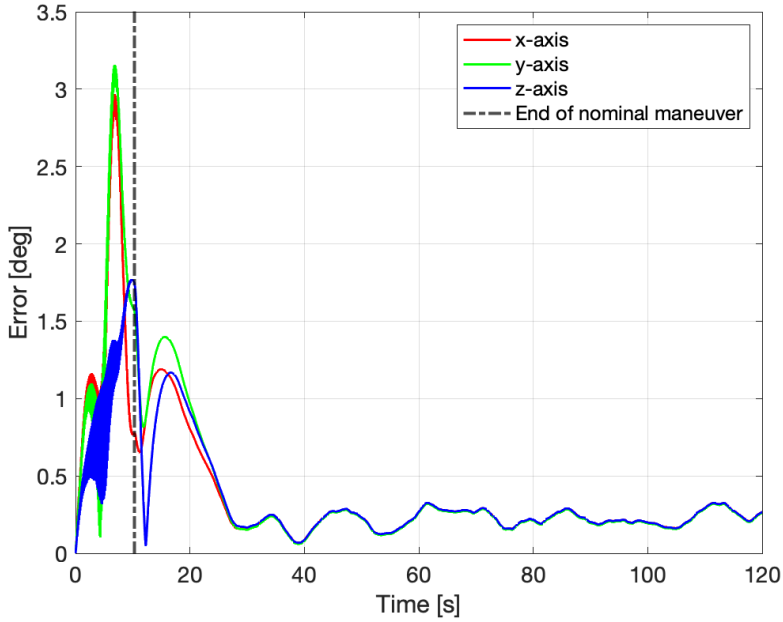


Figure 6.8: Pointing error - FFPD controller

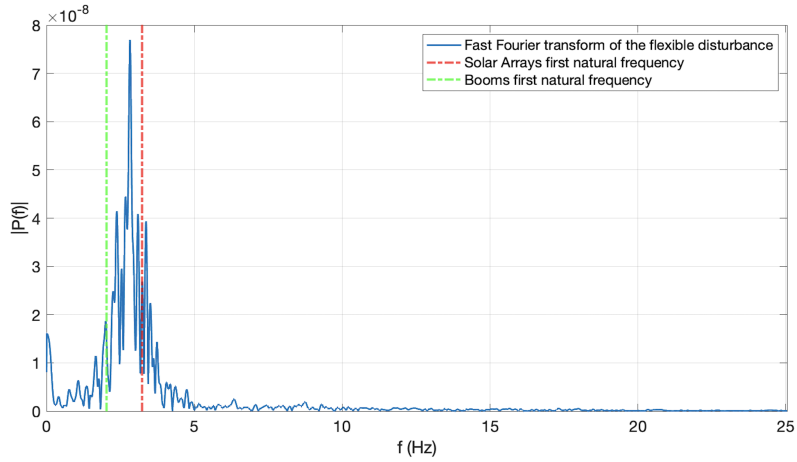


Figure 6.9: Frequency analysis of the flexible disturbance

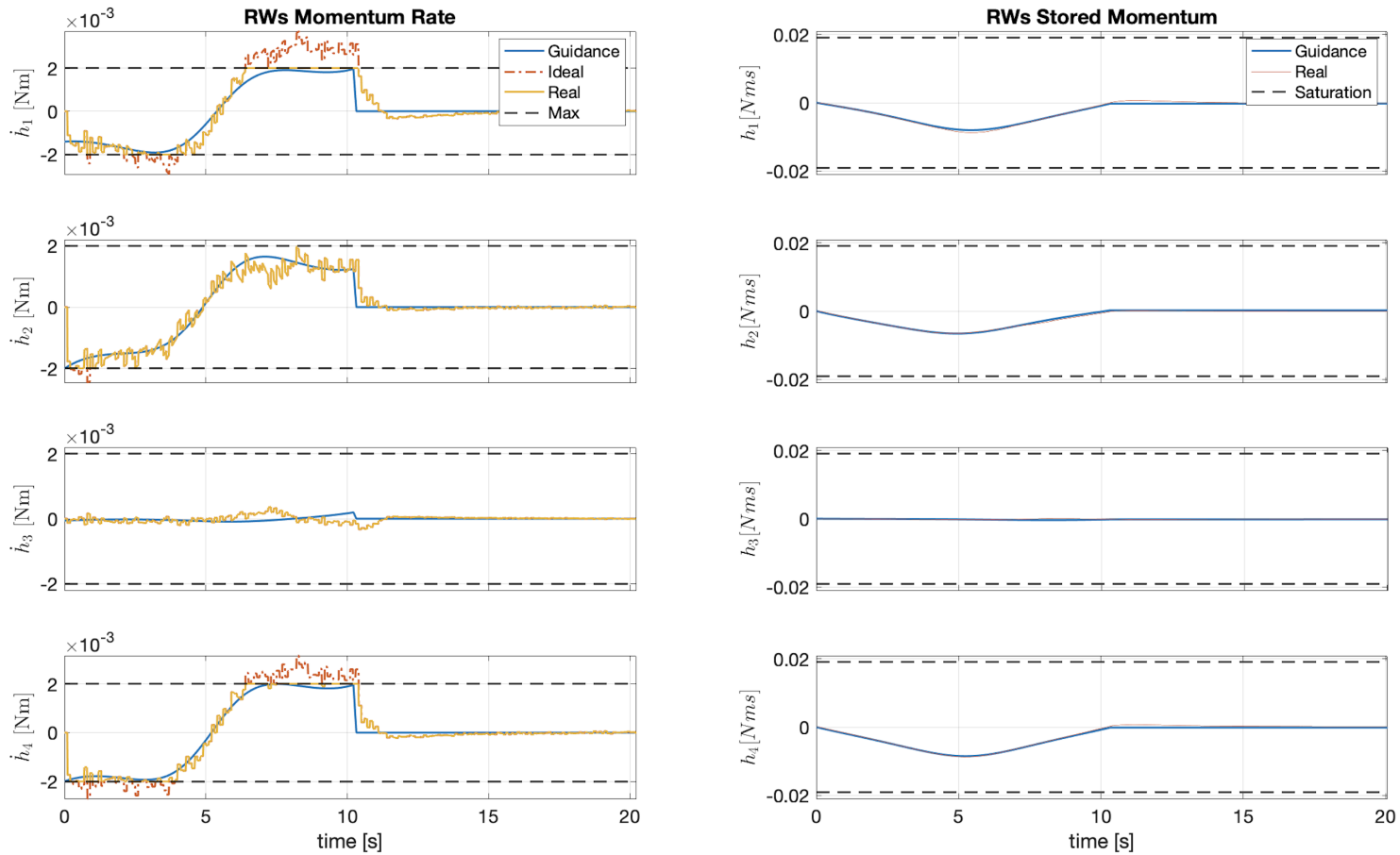


Figure 6.10: Comparison of nominal, ideal and real control actions during the slew maneuver - Feed Forward PD with flexible model

CHAPTER 7

Conclusions

This Chapter states the tentative conclusions of this work in relation to the objective of the research presented in Chapter 1. In particular, Section 7.1 shows the major results achieved, while Section 7.2 tries to pave the way for possible future works.

7.1 Major Results

The presentation of the future results follows the order of the research objective presented in Chapter 1.

Objective 1: To derive a fast semi-analytical guidance algorithm for the CoM motion that is suitable to deal with the peculiarities of the small-sat and their mission scenarios as indicated in the introduction, that is able to overcome some of the typical limitation of the in-literature available algorithms. The method should be able to quickly identify near-optimal trajectories even in presence of very complex mission scenarios.

The task was addressed in Chapter 3 with the development and testing o a shape-based algorithm for the CoM motion. The algorithm can be successfully employed, as shown in the related test cases and comparison with literature reported in sec. 3.1.3, both as ground-based guidance for small-sat equipped with low thrust, or to look for near-optimal trajectories in very complex mission scenarios, to be further optimized using, for example, a direct or indirect approach. Moreover, it shows also good performances when employed in the multiple injection planning. The algorithm shows a good convergence and it is extremely fast and sufficiently flexible to insert operational and platform sizing constraints which affect the trajectory definition. In particular, the implemented strategy to take into account no-thrust in shadow requirements is fast and efficient and works with any class of orbits without singularities. The algorithm converges for any value of the maximum thrust if purely keplerian motion is considered. The speed of the algorithm is guaranteed by the fact that, even if an large number of expressions have to be evaluated, they are all composed by the same block of repeated terms that can be computed only once. Moreover, most of the trigonometrical expressions are not directly evaluated by MATLAB functions, but exploiting trigonometrical relations. The main drawback of the developed method, that is also a common problem of almost all shape based algorithms available in literature, is that the solution can be only near-optimal due to the continuous nature of the shaping functions. The goodness of the solution, as pointed out with the test cases reported in Sec. 3.1.3, strongly depends on the specific problem to be solved. The routing algorithm developed and tested in Sec. 3.2 extends the field of applicability to scenarios that are very interesting for future small-sat and CubeSat missions [66], such as the multi-mission injection and the orbital change in Sun-Synchronous Orbit (SSO) environment taking advantage from natural perturbation to the RAAN drift, saving a considerable fraction of fuel mass and reducing the overall cost of the mission allowing the possibility to share the launch with satellites that are released in different orbits. Thanks to the introduction of the matrix containing the possible permutations as input to the problem, proved extremely effective and fast. Moreover, the branch and bound based heuristic approach increases the performances of the heuristic algorithm by allowing to focus only on the most promising releasing orders of the large search space.

Objective 2: To derive a semi-analytical guidance algorithm for the attitude motion of the spacecraft, that is able to manage practical and mission related constraints, such as actuation limitation and attitude keep-out cones. A multi-disciplinary approach should be carried on to tailor the algorithm on small-sat and CubeSat needs and capabilities.

The task was addressed in Chapter 4, in which the novel shape-based algorithm developed for the attitude guidance and planning is presented. The algorithm,

that takes advantage of a semi-analytical implementation based on a non-linear interpolation between the initial and desired propagated attitudes, has proven to be able to find near-optimal slew trajectory even in presence of keep-out cones, as shown from the test cases and literature comparison in Sec. 4.3. In particular the semi-analytical nature of the algorithm ensure a fast convergence and a reduced CPU time, if compared with literature [73, 78]. The possibility to include the actuators dynamics and the perturbation directly into the EoM turns out into a higher fidelity model with respect to the literature. Moreover, the objectives and constraints that are included, as well as the spacecraft configuration (e.g. solar panels size and orientation, antennas LOS direction) make the algorithm particularly suitable for the guidance definition and attitude maneuver planning of small-sat. In principle it could be used also in an hypothetical on-board implementation, at least in two cases. The first is if there are no *strong* attitude constraints, indeed in such case the heuristic approach could be avoided in favor of a less demanding deterministic one. The second is if there is no real interest in the optimization of the slew itself: in this case the heuristic approach can be used with a reduced population, and for a reduced number of iteration to find only a feasible , not optimal and not even necessary close to the global optimum, solution that can be then optimized with a deterministic algorithm. The drawback is that the algorithm find easily feasible solutions only if the number of keep-out cones is reduced (e.g. two or three), but if this number grows the flexibility of the shape is no more sufficient to follow the twisted trajectory. A possible mitigation, to be investigated, is the adoption of an interpolating shape $\chi(x)$ with more degrees of freedom, such as an higher order polynomial or a linear combination of period functions.

Objective 3: To identify the most suitable control schemes and algorithms that can fly on CubeSat hardwares with very limited computational performances but able to follow the developed guidance in an high fidelity simulated environment. Within this framework, the developed guidance algorithms could be also tested.

Chapter 5 highlighted the necessity to have a FF effect in order to contain the pointing error during the slew if attitude constraints are active. Indeed the only presence of a feedback controller is not able to achieve a small error in this phase, while maintaining a moderation in the control. In particular, the FFPD control scheme employed with a *Gain-Scheduling* technique ensured the best performance in the pointing error control during the slew and natural perturbation rejection during the steady-state phase. More advanced control technique could be in principle used, such as MPC or LQR, but the extremely reduced performances of the CubeSat ADCS boards are not able, at the current state, to perform on-line optimizations, and in any case it appears useless to waste on-board CPU time and memory to reach certain performances, if the

same can be reached with simpler methods, with proved stability, reliability and effectiveness.

Objective 4: To explore new multi-body modeling techniques tailored for CubeSat, with the aim of investigate the flexible effect of small-sat on the attitude performances.

Chapter 6 presents a novel approach to the CubeSat multi-body modeling. The model developed is able to takes into account the flexible effect of large deployable solar arrays and booms, such as the UHF/VHF antennas. The model uses a LPM for the discretization of the solar arrays, considering rigid plates and rotational springs, and a DPM approach for the booms, based on a single function RG spatial discretization. It proved to be able to correctly catch the natural frequencies of the CubeSat with a simple but effective approach. The analysis of the flexible motion of a representative 3U CubeSat shows that the elastic disturbances of the solar panels in very fast slew can be order of magnitude higher with respect to the natural perturbations, but due to its period behavior, the pointing error is only slightly affected. One strong limitation of the developed algorithm, that could be overcome in future, is that the bending dynamics of the booms are elastically decoupled, and no torsion is considered. This can be a good approximation, is small displacement are considered, for boom with circular or squared sections, but for more non axial symmetric sections this assumption is no more valid.

7.2 Future Works

One Ph.D thesis is surely not sufficient to cover the extremely wide research field of Guidance and Control for CubeSat, and therefore a relevant number of next steps are foreseen.

- It would be interesting to extend the field of application of the shape-based algorithm for CoM motion to other dynamics, such as the Circular Restricted Three-Body Problem (CR3BP), in which a continuously growing interest is manifested.
- A fully coupled 6 DoF semi-analytical guidance could be found starting from the methods developed in this work
- The multi-body modeling of the spacecraft could be improved by introducing the coupling therms in the RG discretization. Moreover, a DPM approach could be effective to model extremely large and flexible plates, such as the steerable High Gain Antennas and Solar Panels with higher fidelity.

- The chain of the Guidance and Control here developed and tested on a personal laptop should be tested with relevant hardware. Therefore a porting of the algorithm will be necessary to start a Processor In the Loop (PIL) test campaign. It would be interesting to close the loop with existent Navigation algorithms.

The points here reported are only a small subset implications directly related to the work performed in this thesis, but the list of possible improvement in the Guidance Navigation and Control for small-sat and CubeSat is virtually endless.

Appendices

APPENDIX A

List of derivatives

The appendix reports the equations fundamental to the geometrical interpolation of the trajectory. The subscript '1' indicates the departure orbit, while the subscript '2' indicates the arrival orbit.

A.1 Departure orbit

The inclination of the initial orbit with respect to the reference plane can be computed as in Eq. A.1.

$$\begin{cases} \cos\alpha_1 = \hat{\mathbf{h}}_1 \cdot \hat{\mathbf{h}}_{REF} \\ \sin\alpha_1 = \xi_1\sqrt{1-\cos\alpha_1{}^2} \\ \begin{cases} \xi_1 = 1 & if \quad \mathbf{v}_i \cdot \hat{\mathbf{h}}_{REF} > 0 \\ \xi_1 = -1 & if \quad \mathbf{v}_i \cdot \hat{\mathbf{h}}_{REF} < 0 \end{cases} \end{cases} \tag{A.1}$$

The declination $(\delta(x)_1)$ of the initial orbit over the reference plane can be computed using Eq. A.2, while its derivatives can be computed using Eq. A.3,

Eq. A.4 and Eq. A.5.

$$\sin\delta_1 = \sin\alpha_1 \frac{\sin(\psi x)}{\sin\beta_1} \tag{A.2}$$

$$\delta_1' = \frac{\psi \sin\alpha_1 \cos(\psi x) - \beta_1' \cos\beta_1 \sin\delta_1}{\cos\delta_1 \sin\beta_1} \tag{A.3}$$

$$\delta_1'' = \frac{-\psi^2 \sin\alpha_1 \sin(\psi x) + \sin\beta_1 \sin\delta_1 \left(\delta_1'^2 + \beta_1'^2\right)}{\cos\delta_1 \sin\beta_1} + \\ - \frac{2\delta_1'\beta_1' \cos\delta_1 \cos\beta_1 + \beta_1'' \sin\delta_1 \cos\beta_1}{\cos\delta_1 \sin\beta_1} \tag{A.4}$$

$$\delta_1''' = \frac{-\psi^3 \sin\alpha_1 \cos(\psi x) + \sin\beta_1 \sin\delta_1 \left(3\delta_1'\delta_1'' + 3\beta_1'\beta_1''\right)}{\cos\delta_1 \sin\beta_1} + \\ \cos\beta_1 \frac{-\cos\delta_1 \left(3\delta_1'\delta_1'' + 3\beta_1'\beta_1''\right) + \sin\delta_1 \left(3\delta_1'^2\beta_1' + \beta_1'^3 - \beta_1'''\right)}{\cos\delta_1 \sin\beta_1} \\ + \frac{\sin\beta_1 \cos\delta_1 \left(3\beta_1'^2\delta_1' + \delta_1'^3\right)}{\cos\delta_1 \sin\beta_1} \tag{A.5}$$

In the previous equations another spherical angle ($\beta(x)_1$ in Figure 3.2) is introduced together with its derivatives. They can be computed using Eq. A.6.

$$\begin{cases} \beta_1 = \arccos\left(\sin\alpha_1 \cos(\psi x)\right) \\ \beta_1' = \psi \sin\alpha_1 \frac{\sin(\psi x)}{\sin\beta_1} \\ \beta_1'' = \frac{\psi^2 \sin\alpha_1 \cos(\psi x) - \cos\beta_1 \beta_1'^2}{\sin\beta_1} \\ \beta_1''' = \frac{-\psi^3 \sin\alpha_1 \sin(\psi x) - 3\beta_1'\beta_1'' \cos\beta_1 + \beta_1'^3 \sin\beta_1}{\sin\beta_1} \end{cases} \tag{A.6}$$

$\Delta L_1(x)$ is fundamental to compute the Longitude ($6^{th} MEE$) on the initial orbit at each x and, as a consequence, the attractor distance from the attractor on the departure orbit as function of x; it can be computed with Eq. A.7 and Eq. A.8.

$$\begin{cases} \sin(\Delta L_1) = \frac{1}{\sin\alpha_1} \sin\delta_1 \\ \cos(\Delta L_1) = \cos(\psi x) \cos\delta_1 \end{cases} \tag{A.7}$$

$$\begin{cases} \Delta L_1' = \frac{\delta_1'}{\sin\alpha_1 \cos(\psi x)} \\ \Delta L_1'' = \frac{\delta_1'' + \psi \sin\alpha_1 \sin(\psi x) \Delta L_1'}{\sin\alpha_1 \cos(\psi x)} \\ \Delta L_1''' = \frac{\delta_1''' + 2\psi \sin\alpha_1 \sin(\psi x) \Delta L_1'' + \psi^2 \sin\alpha_1 \cos(\psi x) \Delta L_1'}{\sin\alpha_1 \cos(\psi x)} \end{cases} \tag{A.8}$$

A.2 Target orbit

The inclination of the arrival orbit with respect to the reference plane can be computed as in Eq. A.9.

$$\begin{cases} \cos\alpha_2 = \hat{\mathbf{h}}_2 \cdot \hat{\mathbf{h}}_{REF} \\ \sin\alpha_2 = \xi_2\sqrt{1-\cos{\alpha_2}^2} \\ \xi_2 = 1 \qquad if \quad \mathbf{v}_f \cdot \hat{\mathbf{h}}_{REF} < 0 \\ \xi_2 = -1 \qquad if \quad \mathbf{v}_f \cdot \hat{\mathbf{h}}_{REF} > 0 \end{cases} \tag{A.9}$$

The declination $(\delta(x)_2)$ of the arrival orbit over the reference plane can be computed using Eq. A.10, while its derivatives can be computed using Eq. A.11, Eq. A.12 and Eq. A.13

$$\sin\delta_2 = \sin\alpha_2 \frac{\sin(\psi(1-x))}{\sin\beta_2} \tag{A.10}$$

$$\delta_2' = \frac{-\psi\sin\alpha_2\cos\left(\psi(1-x)\right) - \beta_2'\cos\beta_2\sin\delta_2}{\cos\delta_2\sin\beta_2} \tag{A.11}$$

$$\begin{aligned} \delta_2'' = \frac{-\psi^2\sin\alpha_2\sin\left(\psi(1-x)\right) + \sin\beta_2\sin\delta_2\left(\delta_2'^2+\beta_2'^2\right)}{\cos\delta_2\sin\beta_2} + \\ - \frac{2\delta_2'\beta_2'\cos\delta_2\cos\beta_2 + \beta_2''\sin\delta_2\cos\beta_2}{\cos\delta_2\sin\beta_2} \end{aligned} \tag{A.12}$$

$$\begin{aligned} \delta_2''' = \frac{\psi^3\sin\alpha_2\cos\left(\psi(1-x) + \sin\beta_2\sin\delta_2\left(3\delta_2'\delta_2'' + 3\beta_2'\beta_2''\right)\right.}{\cos\delta_2\sin\beta_2} + \\ \cos\beta_2\frac{\cos\delta_2\left(3\delta_2'\delta_2'' + 3\beta_2'\beta_2''\right) + \sin\delta_2\left(3\delta_2'^2\beta_2' + \beta_2'^3 - \beta_2'''\right)}{\cos\delta_2\sin\beta_2} \\ + \frac{\sin\beta_2\cos\delta_2\left(3\beta_2'^2\delta_2' + \delta_2'^3\right)}{\cos\delta_2\sin\beta_2} \end{aligned} \tag{A.13}$$

In the previous equations another spherical angle $(\beta(x)_2$ in Figure 3.2) is introduced together with its derivatives. They can be computed using Eq. A.14.

$$\begin{cases} \beta_2 = \arccos\left(\sin\alpha_2\cos(\psi(1-x))\right) \\ \beta_2' = -\psi\sin\alpha_2\frac{\sin\left(\psi(1-x)\right)}{\sin\beta_2} \\ \beta_2'' = \frac{-\psi^2\sin\alpha_2\cos\left(\psi(1-x)\right) - \cos\beta_2\beta_2'^2}{\sin\beta_2} \\ \beta_2''' = \frac{\psi^3\sin\alpha_2\sin\left(\psi(1-x)\right) - 3\beta_2'\beta_2''\cos\beta_2 + \beta_2'^3\sin\beta_2}{\sin\beta_2} \end{cases} \tag{A.14}$$

The angle $\Delta L_2(x)$ is fundamental to compute the Longitude ($6^{th}MEE$) on the arrival orbit at each x and so the attractor distance of the arrival orbit as function of x; it can be computed with Eq. A.15 and Eq. A.16.

$$\begin{cases} \sin(\Delta L_2) = \frac{1}{\sin\alpha_2}\sin\delta_2 \\ \cos(\Delta L_2) = \cos(\psi(1-x))\cos\delta_2 \end{cases} \tag{A.15}$$

$$\begin{cases} \Delta L_2' = \frac{\delta_2'}{\sin\alpha_2\cos(\psi(1-x))} \\ \Delta L_2'' = \frac{\delta_2'' - \psi\sin\alpha_2\sin(\psi(1-x))\Delta L_2'}{\sin\alpha_2\cos(\psi(1-x))} \\ \Delta L_2''' = \frac{\delta_2''' + 2\psi\sin\alpha_2\sin(\psi(1-x))\Delta L_2'' + \psi^2\sin\alpha_2\cos(\psi(1-x))\Delta L_2'}{\sin\alpha_2\cos(\psi(1-x))} \end{cases} \tag{A.16}$$

A.3 Attractor distances

To compute the attractor distance the Longitude at each position x on the initial and final orbits using Eq. A.17 and Eq. A.18 has to be computed first.

$$\begin{cases} l_1(x) = L_1 + \Delta L_1(x) \\ l_1'(x) = \Delta L_1'(x) \end{cases} \tag{A.17}$$

$$\begin{cases} l_2(x) = L_2 - \Delta L_2(x) \\ l_2'(x) = -\Delta L_2'(x) \end{cases} \tag{A.18}$$

The attractor distance on the initial and final orbits can be computed using Eq. A.19.

$$\begin{cases} s_i(x) = \frac{p_i}{q_i(x)} \\ s_i(x)' = -\frac{p_i q_i'}{q_i^2} \\ s_i(x)'' = 2\frac{p_i q_i'^2}{q_i^3} - \frac{p_i q_i''}{q_i^2} \\ s_i(x)''' = -6\frac{p_i q_i'^3}{q_i^4} + 6\frac{p_i q_i' q_i''}{q_i^3} - \frac{p_i q_i'''}{q_i^2} \end{cases} \tag{A.19}$$

the q term appears with its derivatives; it can be computed using Eq. A.20

$$\begin{cases} q_i(x) = 1 + f_i\cos l_i(x) + g_i\sin l_i(x) \\ q_i(x)' = (-f_i\sin l_i + g_i\cos l_i)\,\Delta L_i' \\ q_i(x)'' = (1-q_i)\Delta L_i'^2 + q_i'\frac{\Delta L_i''}{\Delta L_i'^2} \\ q_i(x)''' = -q_i'\Delta L_i'^2 + 3(1-q_i)\Delta L_i'\Delta L_i'' + q_i'\frac{\Delta L_i'''}{\Delta L_i'} \end{cases} \tag{A.20}$$

Bibliography

[1] "Cubesat design specification", *Cal Poly – San Luis Obispo, CA*, 2020.
[2] J. Bouwmeester and J. Guo, "Survey of worldwide pico-and nanosatellite missions, distributions and subsystem technology", *Acta Astronautica*, vol. 67, no. 7-8, pp. 854–862, 2010.
[3] H. C. Polat, J. Virgili-Llop, and M. Romano, "Survey, statistical analysis and classification of launched cubesat missions with emphasis on the attitude control method", *Journal of small satellites*, vol. 5, no. 3, pp. 513–530, 2016.
[4] J. Prinetto and M. Lavagna, "Main belt active asteroids samples collection and return mission design", in *Italian Association of Aeronautics and Astronautics XXIV International Conference, AIDAA, Palermo-Enna, Italy*, 2017.
[5] B. A. Conway, *Spacecraft Trajectory Optimization*. Cambridge University Press, 2010.
[6] V. A. Chobotov, *Orbital Mechanics*. AIAA Education Series, 2002.
[7] B. Wall and B. Conway, "Shape-based approach to low-thrust rendezvous trajectory design", *Journal of Guidance, Control, and Dynamics*, vol. 32, pp. 95–101, 2009.
[8] P. J. Cefola, "Equinoctial orbital elements - application to artificial satellite orbits", in *AIAA/AAS Astrodynamics Specialist Conference and Exhibit, Palo Alto, California*, 1972.
[9] H. Tsien, "Take-off from satellite orbit", *Journal of the American Rocket Society*, vol. 23, pp. 233–236, 1953.
[10] R. Battin, *An Introduction to the Mathematics and Methods of Astrodynamics*. AIAA education Series, 1999.

[11] R. Kluever C. Oleson, "Direct approach for computing near-optimal low-thrust earth-orbit transfers", *Journal of Spacecraft and Rockets*, vol. 35, pp. 509–515, 1998.

[12] F. Topputo and C. Zhang, "Survey of direct transcription for low-thrust space trajectory optimization with applications", *Abstract and Applied Analysis*, 2014.

[13] B. J. Wall, "Shape-based approximation method for low-thrust trajectory optimization", in *AIAA/AAS Astrodynamics Specialist Conference and Exhibit, AIAA, Honolulu, Hawaii*, 2008.

[14] A. E. Petropoulos and J. a. Sims, "A Review of Some Exact Solutions to the Planar Equations of Motion of a Thrusting Spacecraft Anastassios", *International Symposium Low Thrust Trajectories*, vol. 2, no. 1, pp. 1–14, 2002.

[15] M. V. De Pascale P., "Preliminary design of low-thrust multiple gravity-assist trajectories", *Journal of Spacecraft and Rockets*, vol. 43, pp. 1065–1076, 2006.

[16] A. E. Petropoulos and J. M. Longuski, "Shape-based algorithm for automated design of low-thrust. gravity-assist trajectories", *Journal of Spacecraft and Rockets*, vol. 41, pp. 787–796, 2004.

[17] C. Xie, G. Zhang, and Y. Zhang, "Shaping approximation for low-thrust trajectories with large out-of-plane motion", *Journal of Guidance, Control, and Dynamics*, vol. 39, pp. 2776–2785, 2016.

[18] K. Zeng, Y. Geng, and B. Wu, "Shape-based analytic safe trajectory design for spacecraft equipped with low-thrust engines", *Journal of Spacecraft and Rockets*, vol. 62, pp. 87–97, 2017.

[19] M. V. Novak D. M., "Improved shaping approach to the preliminary design of low-thrust trajectories", *Journal of Guidance, Control, and Dynamics*, vol. 34, pp. 128–147, 2011.

[20] E. A. Ehsan Taheri Ilya Kolmanovsky, "Shaping low-thrust trajectories with thrust-handling feature", *Advance in Space Research*, pp. 879–890, 2017.

[21] O. Abdelkhalik and E. Taheri, "Approximate on-off low-thrust space trajectories using fourier series", *Journal of Spacecraft and Rockets*, vol. 49, pp. 962–965, 2012.

[22] E. Taheri and O. Abdelkhalik, "Initial three-dimensional low-thrust trajectory design", *Advances in Space Research*, vol. 57, pp. 889–903, 2016.

[23] D. J. Gondelach and R. Noomen, "Hodographic-shaping method for low-thrust interplanetary trajectory design", *Journal of Spacecraft and Rockets*, vol. 52, pp. 728–738, 2015.

[24] O. Abdelkhalik and E. Taheri, "Shape based approximation of constrained low-thrust space trajectories using fourier series", *Journal of Spacecraft and Rockets*, vol. 49, pp. 535–546, 2012.

[25] K. T. Alfriend, D. J. Lee, and N. Glenn Creamer, "Optimal servicing of geosynchronous satellites", *AIAA/AAS Astrodynamics Specialist Conference and Exhibit*, no. August, pp. 1–9, 2002. DOI: 10.2514/6.2002-4905.

[26] D. Izzo, D. Hennes, I. Getzner, and L. F. Simões, "Evolving solutions to TSP variants for active space debris removal", *GECCO 2015 - Proceedings of the 2015 Genetic and Evolutionary Computation Conference*, pp. 1207–1214, 2015. DOI: 10.1145/2739480.2754727.

[27] N. Bérend and X. Olive, "Bi-objective optimization of a multiple-target active debris removal mission", *Acta Astronautica*, vol. 122, pp. 324–335, 2016. DOI: 10.1016/j.actaastro.2016.02.005.

[28] K. Alemany and R. D. Braun, "Survey of global optimization methods for low-thrust, multiple asteroid tour missions", 2007.

[29] B. J. Wall and B. A. Conway, "Genetic algorithms applied to the solution of hybrid optimal control problems in astrodynamics", *Journal of Global Optimization*, vol. 44, no. 4, pp. 493–508, 2009. DOI: 10.1007/s10898-008-9352-4.

[30] J. Zhang, Y. Luo, H. Li, and G. Tang, "Analysis of multiple asteroids rendezvous optimization using genetic algorithms", *2015 IEEE Congress on Evolutionary Computation, CEC 2015 - Proceedings*, pp. 596–602, 2015. DOI: 10.1109/CEC.2015.7256945.

[31] J. Prinetto and M. Lavagna, "Elliptical shape-based model for multi-revolution planeto-centric mission scenarios", *Celestial Mechanics and Dynamical Astronomy*, vol. 133, no. 1, pp. 1–24, 2021. DOI: 10.1007/s10569-020-10001-9.

[32] M. Diaz Ramos and H. Schaub, "Kinematic steering law for conically constrained torque-limited spacecraft attitude control", *Journal of Guidance, Control, and Dynamics*, vol. 41, no. 9, pp. 1990–2001, 2018.

[33] G Avanzini, G Radice, and I Ali, "Potential approach for constrained autonomous manoeuvres of a spacecraft equipped with a cluster of control moment gyroscopes", *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering*, vol. 223, no. 3, pp. 285–296, 2009.

[34] H. C. Kjellberg and E. G. Lightsey, "Discretized constrained attitude pathfinding and control for satellites", *Journal of Guidance, Control, and Dynamics*, vol. 36, no. 5, pp. 1301–1309, 2013.

[35] E Frazzoli, M. Dahleh, E Feron, and R Kornfeld, *A randomized attitude slew planning algorithm for autonomous spacecraft*. AIAA Reston, VA, 2001, vol. 4155.

[36] X. Tan, S. Berkane, and D. V. Dimarogonas, "Constrained attitude maneuvers on so (3): Rotation space sampling, planning and low-level control", *Automatica*, vol. 112, p. 108 659, 2020.

[37] H. B. Hablani, "Attitude commands avoiding bright objects and maintaining communication with ground station", *Journal of Guidance, Control, and Dynamics*, vol. 22, no. 6, pp. 759–767, 1999.

[38] J. D. Biggs and L. Colley, "Geometric attitude motion planning for spacecraft with pointing and actuator constraints", *Journal of Guidance, Control, and Dynamics*, vol. 39, no. 7, pp. 1672–1677, 2016.

[39] G. A. Boyarko, M. Romano, and O. A. Yakimenko, "Time-optimal reorientation of a spacecraft using an inverse dynamics optimization method", *Journal of Guidance, Control, and Dynamics*, vol. 34, no. 4, pp. 1197–1208, 2011.

[40] R. G. Melton, "Differential evolution/particle swarm optimizer for constrained slew maneuvers", *Acta Astronautica*, vol. 148, pp. 246–259, 2018.

[41] M. J. Sidi, *Spacecraft dynamics and control: a practical engineering approach*. Cambridge university press, 1997, vol. 7, pp. 88–111.

[42] A. A. Shabana, *Dynamics of multibody systems*. Cambridge university press, 2020.

[43] J. L. Junkins, *Introduction to dynamics and control of flexible structures*. Aiaa, 1993.

[44] L.Meirovitch, *Fundamentals of vibrations*. McGraw-Hill, 2001.

[45] A. W. Leissa and M. S. Qatu, *Vibrations of continuous systems*. McGraw-Hill Education, 2011.

[46] M. Petyt, *Introduction to finite element vibration analysis*. Cambridge university press, 2010.

[47] J. Pearl, *Heuristics: intelligent search strategies for computer problem solving*. Addison-Wesley Longman Publishing Co., Inc., 1984.

[48] C. Blum and A. Roli, "Metaheuristics in combinatorial optimization: Overview and conceptual comparison", *ACM computing surveys (CSUR)*, vol. 35, no. 3, pp. 268–308, 2003.

[49] J. H. Holland *et al.*, *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.

[50] R. Marte *et al.*, *Handbook of heuristics*. Springer, 2018.

[51] J. Kennedy and R. Eberhart, "Particle swarm optimization", *Proceedings of ICNN'95 - International Conference on Neural Networks*, vol. 4, 1942–1948 vol.4, 1995. DOI: 10.1109/ICNN.1995.488968.

[52] H. Curtis, *Orbital Mechanics for Engineering Students*. Elsevier, 2005.

[53] O. Sutton G. P., *Rocket Propulsion Elements*. Wiley, 2010.

[54] W. J. Larson and J. R. Wertz, *Space Mission Analysis and Design*. Kluver Academy Publisher, 1999.

[55] S. Sreesawet and A. Dutta, "Fast and robust computation of low-thrust orbit-raising trajectories", *Journal of Guidance, Control, and Dynamics*, vol. 41, pp. 1888–1905, 2018.

[56] Arianespace, *VEGA user's Manual*. 2014.

[57] E. European Space Agency, *pace Debris Mitigation Compliance Verification Guidelines*. 2015.

[58] S. J. W. Nocedal J., *Numerical Optimization*. Springer, 2006.

[59] A. V. Graham K. F.and Rao, "Minimum-time trajectory optimization of multiple revolution low-thrust earth-orbit transfers", *Journal of Spacecraft and Rockets*, vol. 52, 2015.

[60] NASA. "What are smallsats and cubesats?" (), [Online]. Available: `https://www.nasa.gov/content/what-are-smallsats-and-cubesats`. //accessed: 30.03.2021.

[61] Euroconsult, *Satellites To Be Built and Launched by 2028. A complete analysis & forecast of satellite manufacturing & launch services*. 2019, pp. 5–10.

[62] SpaceWorks. "2020 nano/microsatellite market forecast, 10th edition". (2020).

[63] J. Bang and J. Ahn, "Two-phase framework for near-optimal multi-target Lambert rendezvous", *Advances in Space Research*, vol. 61, no. 5, pp. 1273–1285, 2018. DOI: `10.1016/j.asr.2017.12.025`.

[64] "MATLAB optimization toolbox". The MathWorks, Natick, MA, USA. (2020b), [Online]. Available: `https://it.mathworks.com/products/global-optimization.html`.

[65] P. Fortescue, G. Swinerd, and J. Stark, *Spacecraft Systems Engineering*. Wiley, 2011.

[66] S. Silvestrini, J. Prinetto, G. Zanotti, and M. Lavagna, "Design of robust passively safe relative trajectories for uncooperative debris imaging in preparation to removal", in *2020 AAS/AIAA Astrodynamics Specialist Conference*, 2020, pp. 1–18.

[67] J. Pearl, *Heuristics Intelligent Search Strategies for Computer Problem Solving*. Addison-Wesley Publishing Company, 1984, pp. 3–4.

[68] C. Blum and A. Roli, "Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison", *ACM Computing Surveys*, vol. 35, no. 3, pp. 268–308, 2003. DOI: `10.1145/937503.937505`.

[69] V. Martínez-Cagigal. "Multi-objective particle swarm optimization (mopso)". MATLAB Central File Exchange. (2020), [Online]. Available: `https://www.mathworks.com/matlabcentral/fileexchange/62074-multi-objective-particle-swarm-optimization-mopso`.

[70] ESA. "Starlink satellite constellation of spacex". (), [Online]. Available: `https://directory.eoportal.org/web/eoportal/satellite-missions/s/starlink`. //accessed: 30.03.2021.

[71] W. J. Larson and J. R. Wertz, "Space mission analysis and design", Torrance, CA (United States); Microcosm, Inc., Tech. Rep., 1992.

[72] ArieneGroup. "Electric ion space propulsion systems and thrusters". (), [Online]. Available: `https://www.space-propulsion.com/spacecraft-`

propulsion/propulsion-systems/electric-propulsion/index.html. //accessed: 30.03.2021.

[73] F. Celani and R. Bruni, "Minimum-time spacecraft attitude motion planning using objective alternation in derivative-free optimization", *Journal of Optimization Theory and Applications*, vol. 191, no. 2, pp. 776–793, 2021.

[74] R. H. Byrd, J. C. Gilbert, and J. Nocedal, "A trust region method based on interior point techniques for nonlinear programming", *Mathematical programming*, vol. 89, no. 1, pp. 149–185, 2000.

[75] R. H. Byrd, M. E. Hribar, and J. Nocedal, "An interior point algorithm for large-scale nonlinear programming", *SIAM Journal on Optimization*, vol. 9, no. 4, pp. 877–900, 1999.

[76] J. Kennedy and R. Eberhart, "Particle swarm optimization", in *Proceedings of ICNN'95-international conference on neural networks*, IEEE, vol. 4, 1995, pp. 1942–1948.

[77] J. Lagarias, J. Reeds, M. Wright, and P. Wright, "Convergence properties of the nelder-mead simplex method in low dimensions", *SIAM Journal on Optimization*, vol. 9, no. 1, pp. 112–147, 1998. DOI: 10.1137/S1052623496303470.

[78] D. Spiller, L. Ansalone, and F. Curti, "Particle swarm optimization for time-optimal spacecraft reorientation with keep-out cones", *Journal of Guidance, Control, and Dynamics*, vol. 39, no. 2, pp. 312–325, 2016.

[79] B. Wie, *Space vehicle dynamics and control*. Aiaa, 1998.

[80] M. H. Kaplan, *Modern spacecraft dynamics and control*. Courier Dover Publications, 2020.

[81] F. L. Markley and J. L. Crassidis, *Fundamentals of spacecraft attitude determination and control*. Springer, 2014, vol. 1286.

[82] F. Fiore *et al.*, "The hermes-technologic and scientific pathfinder", in *Space Telescopes and Instrumentation 2020: Ultraviolet to Gamma Ray*, SPIE, vol. 11444, 2020, pp. 214–228.

[83] A Colagrossi, S Silvestrini, J Prinetto, M Lavagna, *et al.*, "Hermes: A cubesat based constellation for the new generation of multi-messenger astrophysics", *ADVANCES IN THE ASTRONAUTICAL SCIENCES*, vol. 175, pp. 4243–4262, 2021.

[84] A. Colagrossi, J. Prinetto, S. Silvestrini, and M. R. Lavagna, "Sky visibility analysis for astrophysical data return maximization in hermes constellation", *Journal of Astronomical Telescopes, Instruments, and Systems*, vol. 6, no. 4, p. 048 001, 2020.

[85] X. Bai and J. L. Junkins, "New results for time-optimal three-axis reorientation of a rigid spacecraft", *Journal of guidance, control, and dynamics*, vol. 32, no. 4, pp. 1071–1076, 2009.

[86] Y.-L. Kuo and T.-L. Wu, "Open-loop and closed-loop attitude dynamics and controls of miniature spacecraft using pseudowheels", *Computers & Mathematics with Applications*, vol. 64, no. 5, pp. 1282–1290, 2012.

[87] A. Datta, M.-T. Ho, and S. P. Bhattacharyya, *Structure and synthesis of PID controllers*. Springer Science & Business Media, 1999.

[88] K. J. Åström and T. Hägglund, "Pid control", *IEEE Control Systems Magazine*, vol. 1066, 2006.

[89] J. R. Wertz, *Spacecraft Attitude Determination and Control*. Kluwer Academic Publishers, 1990.

[90] A. Klesh and J. Krajewski, "Marco: Cubesats to mars in 2016", 2015.

[91] M. Géradin and D. J. Rixen, *Mechanical vibrations: theory and application to structural dynamics*. John Wiley & Sons, 2014.

Colophon