POLITECNICO DI TORINO

Master of Science in Aerospace Engineering

Academic Year 2023/2024

Master's Degree Thesis

Design of an optimization algorithm for in-orbit inspection relative trajectories

Supervisors:
Prof. Elisa CAPELLO
Ing. Stefano PETRAZ
Ing. Davide FABRO

Candidate:
Luca LAGUARDIA

July 2024

Abstract

The exponential increase in space activities has highlighted a growing demand for in-orbit services, which are essential to ensure the sustainability of space operations. In-Orbit Servicing (IOS) represents a true paradigm shift, introducing unprecedented scalability and system flexibility. It provides opportunities for in-orbit maintenance, inspection, refueling, and upgrades, and will potentially change the entire approach to satellite design. However, planning efficient and safe trajectories for IOS missions poses a complex challenge due to the non-convexity of trajectory generation problems and the multiple operational constraints involved.

In this thesis, developed at the *Mission Analysis & Operations* unit of Thales Alenia Space in Turin, a direct numerical optimization method is presented to determine the optimal trajectories for IOS missions, with a specific focus on inspection and docking-type operations involving two spacecraft, a Servicer and a Target. Based on optimization algorithms for non-linear problems, such as Sequential Quadratic Programming (SQP) methods, a robust framework has been developed to generate trajectories that minimize the total ΔV and maximize fuel efficiency, while satisfying mission constraints.

By including the effect of aerodynamic drag in the Hill-based relative dynamic model, the optimizer leverages relative drag control by adjusting the Servicer's drag area to efficiently approach or retreat from the Target, thereby minimizing propellant consumption. The optimization framework includes various approaches, such as the use of control boxes and safety ellipses, to ensure the spacecraft remains within safe operational limits during its maneuvers. Furthermore, to test the optimizer's robustness and adaptability to different operational scenarios, the Target has been modeled with various shapes and geometries, including a traditional cubesat, an ellipsoid-shaped object, and a 3D model of the asteroid Didymos, with the control boxes adapted accordingly.

Finally, a refinement process has been carried out using genetic algorithms, which help explore a broader solution space, thus enhancing the likelihood of finding global optima in complex and multi-modal environments typical of orbital trajectory optimization problems.

Altogether, the methodologies and results demonstrate the effectiveness of the proposed optimization tool in providing optimal trajectories for in-orbit inspection, addressing the increasing demand in the space industry. At the same time, this thesis lays a solid foundation for future and more in-depth studies that could further enhance the safety, efficiency, and longevity of space operations, ultimately contributing to the progress and sustainability of both commercial and scientific in-orbit activities.

Acknowledgements

Vorrei esprimere la mia più sincera gratitudine alla Prof.ssa Elisa Capello del Politecnico di Torino e all'Ing. Davide Fabro di Thales Alenia Space Torino per il loro inestimabile aiuto durante lo svolgimento della tesi. Sono inoltre grato all'Ing. Stefano Petraz per avermi dato la possibilità di compiere questa esperienza e per la fiducia che ha riposto in me, e a tutto il team di Mission Analysis & Operations per il supporto e l'amicizia mostratami in questi mesi.

Sono profondamente in debito verso la mia famiglia per il loro incrollabile sostegno nel corso degli anni. Il loro amore e incoraggiamento hanno significato il mondo per me.

Infine, la mia più profonda gratitudine va ai miei amici storici di Bari e Torino. Siete stati una fonte di forza per me, soprattutto nei momenti più difficili.

"All right, all right, all right."
-Matthew McConaughey

Table of Contents

List of Tables vii

List of Figures ix

1 Introduction 1

2 Trajectory Optimization 9

2.1 Introduction to optimization . . . 9

2.2 Optimal Control Problems . . . 11

2.2.1 Common Solution Approaches . . . 13

2.2.2 Nonlinear Programming and Transcription Methods . . . 13

2.2.3 Black-box Optimization . . . 14

2.3 Adopted Optimization Methods . . . 16

2.3.1 Multiple Shooting Method . . . 16

2.3.2 Multistart and SQP . . . 18

2.3.3 Genetic Algorithm . . . 19

2.3.4 Global Optimization . . . 19

3 Mathematical Model 20

3.1 Reference Frame . . . 20

3.2 Orbit Dynamics . . . 21

3.2.1 Orbital Motion around a Central Body . . . 21

3.2.2 Equations of Relative Motion . . . 22

3.3 Trajectory Types . . . 23

3.3.1 Impulsive Maneuvers . . . 24

3.3.2 Tangential Thrust Maneuvers . . . 24

3.3.3 Radial Thrust Maneuvers . . . 26

3.4 Environmental Disturbances . . . 28

3.4.1 Atmospheric Drag . . . 28

3.4.2 Solar Radiation Pressure . . . 30

3.4.3 Magnetic Force . . . 30

4 Simulation Results 32
4.1 Optimization Problem Setup 32
4.1.1 Objective Function 32
4.1.2 Control Variables 32
4.1.3 Constraints 33
4.1.4 Implementation in MATLAB® 35
4.2 Inspection Phase 36
4.2.1 Control Boxes 36
4.2.2 Field-Of-View and Occlusion 46
4.2.3 Safety Ellipse 57
4.2.4 Return to Starting Position 66
4.3 Docking Phase 73
4.4 Inspection of an Asteroid 81
4.5 Genetic Algorithm Refinement 88
4.6 Pareto Front 89

5 Conclusions and Future Developments 90

Bibliography 92

List of Tables

4.1 Initial orbital parameters and spacecraft characteristics 34
4.2 Initial state of the spacecraft (Case 1.1) 37
4.3 Shape and dimensions of the control boxes (Case 1.1) 37
4.4 Optimization results (Case 1.1) 38
4.5 Initial state of the spacecraft (Case 1.2) 39
4.6 Shape and dimensions of the control boxes (Case 1.2) 39
4.7 Optimization results (Case 1.2) 40
4.8 Initial state of the spacecraft (Case 1.3) 41
4.9 Shape and dimensions of the control boxes (Case 1.3) 41
4.10 Initial state of the spacecraft (Case 1.4) 42
4.11 Shape and dimensions of the control boxes (case 1.4) 42
4.12 Initial state of the spacecraft (Case 1.5) 43
4.13 Shape and dimensions of the control boxes (Case 1.5) 43
4.14 Initial state of the spacecraft (Case 1.6) 44
4.15 Shape and dimensions of the control boxes (Case 1.6) 44
4.16 Optimization results (Case 1.6) 45
4.17 Initial state of the spacecraft (Cases 2.1, 2.2, 2.3, 2.4) 48
4.18 Shape and dimensions of the control boxes (Case 2.1) 49
4.19 Optimization results (Case 2.1) 50
4.20 Shape and dimensions of the control boxes (Case 2.2) 51
4.21 Optimization results (Case 2.2) 52
4.22 Shape and dimensions of the control boxes (Case 2.3) 53
4.23 Optimization results (Case 2.3) 54
4.24 Shape and dimensions of the control boxes (Case 2.4) 55
4.25 Optimization results (Case 2.4) 56
4.26 Initial state of the spacecraft (Cases 3.1, 3.2, 3.3) 59
4.27 Shape and dimensions of the control boxes (Cases 3.1, 3.2, 3.3) . . . 60
4.28 Optimization results (Case 3.1) 60
4.29 Optimization results (Case 3.2) 63
4.30 Optimization results (Case 3.3) 65

4.31 Initial state of the spacecraft (Cases 4.1, 4.2, 4.3) 66
4.32 Shape and dimensions of the control boxes (Cases 4.1, 4.2, 4.3) . . . 66
4.33 Optimization results (Case 4.1) 68
4.34 Optimization results (Case 4.2) 70
4.35 Optimization results (Case 4.3) 72
4.36 Initial state of the spacecraft (Cases 5.1, 5.2, 5.3) 74
4.37 Shape and dimensions of the control boxes (Cases 5.1, 5.2, 5.3) . . . 75
4.38 Conic approach corridor (Case 5.1) 75
4.39 Optimization results (Case 5.1) 76
4.40 Conic approach corridor (Case 5.2) 77
4.41 Optimization results (Case 5.2) 78
4.42 Conic approach corridor (Case 5.3) 79
4.43 Optimization results (Case 5.3) 80
4.44 Initial state of the spacecraft (Case 6.1) 81
4.45 Dimensions of the Target (Case 6.1) 81
4.46 Shape and dimensions of the control boxes (Case 6.1) 81
4.47 Optimization results (Case 6.1) - Part 1 83
4.48 Optimization results (Case 6.1) - Part 2 84
4.49 Initial state of the spacecraft 85
4.50 Dimensions of the Target (Case 6.2) 85
4.51 Shape and dimensions of the control boxes (Case 6.2) 86
4.52 Optimization results (Case 6.2) 86
4.53 Optimization results (Case 7) 88

List of Figures

1.1 Illustration of an In-Orbit Servicing mission [1] 1
1.2 Hubble Space Telescope Servicing Missions 4
1.3 ESA's Space Safety Programme missions 6
1.4 View of Intelsat 901 from MEV-1 during approach [7] 7

2.1 Representation of convex and non-convex functions [12] 10
2.2 Schematic representation of the black-box optimization framework [13] 15
2.3 Single Shooting vs Multiple Shooting methods for a four-section trajectory: the optimization algorithm aims to minimize the error (or defect h) to zero [14] 17
2.4 Principle of the Multistart algorithm [15] 18

3.1 LVLH frame definition [17] 21
3.2 Transfer along V-bar by tangential impulses [16] 24
3.3 Transfer to orbit of different height (Hohmann transfer) [16] 25
3.4 Fly-around by tangential impulse [16] 25
3.5 Continuous x-thrust transfer to different altitude [16] 26
3.6 Transfer along V-bar by radial impulses [16] 26
3.7 Fly-around by radial impulse [16] 27
3.8 Transfer along V-bar by continuous z-thrust [16] 27
3.9 Out-of-plane impulse resulting in sinusoidal motion [16] 28
3.10 Spacecraft in LEO encounter drag from the residual atmosphere, causing their orbits to decay [18] 29
3.11 Conceptual explanation of differential drag control: if the Chaser deploys its drag plates, it experiences a relative deceleration compared to the Target (left); if the Target deploys its drag plates, it causes a relative acceleration of the Chaser with respect to the Target (right) [19] 30
3.12 Solar pressure force on the satellite surface [20] 31

4.1 Impulsive discretization scheme [11] 33

4.2 Illustration of the Servicer in different configurations: minimum ballistic coefficient or maximum relative drag (left), maximum ballistic coefficient or minimum relative drag (center), optimal ballistic coefficient or optimal relative drag (right) 34
4.3 Optimized trajectory of the Servicer around the Target with constraints on the control boxes (Case 1.1) 37
4.4 Optimized trajectory of the Servicer around the Target with constraints on the control boxes (Case 1.2) 39
4.5 Optimized trajectory of the Servicer around the Target with constraints on the control boxes (Case 1.3) 41
4.6 Optimized trajectory of the Servicer around the Target with constraints on the control boxes (Case 1.4) 42
4.7 Optimized trajectory of the Servicer around the Target with constraints on the control boxes (Case 1.5) 43
4.8 Trajectory of the Servicer around the Target with constraints on the control boxes (Case 1.6) 44
4.9 Representation of the FOV and occlusion constraints 46
4.10 Inspection points of the target surface 47
4.11 Optimized trajectory of the Servicer around the Target with constraints on the control boxes, FOV, and LOS occlusion (Case 2.1) . 49
4.12 Optimized trajectory of the Servicer around the Target with constraints on the control boxes, FOV, and LOS occlusion (Case 2.2) . 51
4.13 Optimized trajectory of the Servicer around the Target with constraints on the control boxes, FOV, and LOS occlusion (Case 2.3) . 53
4.14 Trajectory of the Servicer around the Target with constraints on the control boxes, FOV, and LOS occlusion (Case 2.4) 55
4.15 Relative motion in the Walking Safety Ellipse [25] 57
4.16 Safety Ellipse acquisition attempt 58
4.17 Safety Ellipse implementation [16] 59
4.18 Optimized trajectory of the Servicer around the Target with constraints on the control boxes, FOV, LOS occlusion and SE (Case 3.1) 61
4.19 Optimized trajectory of the Servicer around the Target with constraints on the control boxes, FOV, LOS occlusion and SE (Case 3.2) 62
4.20 Optimized trajectory of the Servicer around the Target with constraints on the control boxes, FOV, LOS occlusion and SE (Case 3.3) 64
4.21 Optimized trajectory of the Servicer around the Target with constraints on the control boxes, FOV, LOS occlusion, SE, and return (Case 4.1) 67

4.22 Optimized trajectory of the Servicer around the Target with constraints on the control boxes, FOV, LOS occlusion, SE, and return (Case 4.2) 69

4.23 Optimized trajectory of the Servicer around the Target with constraints on the control boxes, FOV, LOS occlusion, SE, and return (Case 4.3) 71

4.24 Conic approach corridor aligned with the docking axis and with an apex half-angle α [26] 73

4.25 Optimized trajectory of the Servicer around the Target for docking within a conic approach corridor (Case 5.1) 75

4.26 Optimized trajectory of the Servicer around the Target for docking within a conic approach corridor (Case 5.2) 77

4.27 Optimized trajectory of the Servicer around the Target for docking within a conic approach corridor (Case 5.3) 79

4.28 Optimized trajectory of the Servicer around the Target with constraints on the control boxes, FOV, LOS occlusion, SE, and return (Case 6.1) 82

4.29 Optimized trajectory of the Servicer around the Target with constraints on the control boxes, FOV and LOS occlusion (Case 6.2) . 87

4.30 Pareto front showing the trade-off between total ΔV and number of impulses N 89

5.1 Continuous thrust modeling for more realistic simulation: the colored segments represent the propulsive arcs of the trajectory 91

Acronyms

SQP

Sequential Quadratic Programming.

IOS

In-Orbit Servicing.

LEO

Low Earth Orbit.

NLP

Non-Linear Programming.

OCP

Optimal Control Problem.

GA

Genetic Algorithm.

BVP

Boundary Value Problem.

CoM

Centre of Mass.

LVLH

Local-Vertical/Local-Horizontal.

CW

Clohessy–Wiltshire.

RVD

Rendezvous and Docking.

FOV

Field-of-View.

LOS

Line-of-Sight.

SE

Safety Ellipse.

Chapter 1

Introduction

Figure 1.1: Illustration of an In-Orbit Servicing mission [1]

The term *In-Orbit Servicing* (IOS) refers to the activities aimed at extending the lifespan or enhancing the functionality of spacecraft already in orbit. This can involve various operations such as performing maintenance, adjusting orbital paths, reorienting the satellite, refueling, or upgrading onboard instruments (Figure 1.1) [2]. Rather than de-orbiting or replacing the spacecraft, conducting these tasks in space provides substantial economic and logistical advantages, paving the way for a new era of space utilization.

Given the complexity of IOS operations, minimizing propellant consumption plays a critical role. Consequently, this thesis focuses on optimizing in-orbit trajectories for an active spacecraft, referred to as the Servicer, Chaser, or Inspector, tasked with inspecting and docking with a passive satellite, referred to as the Target. The optimization process considers constraints on the Inspector's position and the duration of maneuvers to ensure both safety and mission success within a reasonable timeframe.

In light of this, the primary aim of this thesis is to develop an optimization tool for the preliminary analysis of typical IOS missions by incorporating a relatively simple dynamic model to facilitate the optimization process. The optimization strategy employs a multiple-shooting method for time discretization and state propagation, combined with black-box optimization techniques to manage the problem's complexity and seek global optimal solutions.

Additionally, this thesis introduces various approaches that are worthy of further investigation through more detailed analyses. Future studies should incorporate higher-fidelity dynamics, accounting for the effects of several perturbations on both the Target and the Servicer (such as aerodynamic drag using an atmospheric database, J2 effect, solar radiation pressure, etc.), to simulate their relative motion with greater accuracy.

To achieve these goals, the following steps were undertaken:

- Definition of the optimization problem in terms of objective function, control variables, and constraints, which ultimately dictate the Servicer's trajectory around the Target.

- Formulation of the physical model that adequately represents the dynamics of the problem.

- Development of a viable approach to solve the optimization problem effectively, and identification of a suitable optimization algorithm to achieve a global optimum.

- Implementation of the chosen solution method into a tool that guides the user through the definition of the mission scenario.

- Application of the developed method to different mission scenarios to test its ability to identify the best solution.

This thesis consists of five chapters covering all aspects of the problem under investigation as well as potential future developments. Chapter 1 presents the state of the art of In-Orbit Servicing, from the earliest missions to the most recent ones. Chapter 2 focuses on the optimization problem and discusses in detail the strategies employed. Chapter 3 provides an overview of the physical and mathematical models used. Chapter 4 presents the main results achieved, including comparative showcases with different scenarios and models of the Target. Finally, Chapter 5 outlines the steps taken to extend the study to more detailed and physically accurate analyses, and presents the conclusions.

The concept of IOS dates back to the early days of space exploration. These early missions demonstrated the feasibility and potential advantages of In-Orbit Servicing, laying the foundation for more advanced and sustained operations.

The servicing of the **Hubble Space Telescope (HST)** by NASA's Space Shuttle in the 1990s successfully showcased the capability to perform complex repairs and upgrades in space, exploiting Hubble's serviceable design and modular components. These missions significantly extended the telescope's operational lifespan and increased its scientific productivity, benefiting from technological advancements over the years.

In total, five servicing missions were performed on the HST [3]:

1. **Servicing Mission 1 (SM1) - December 1993**: This was the first scheduled maintenance on the telescope. Astronauts installed new instruments, including equipment that addressed the issue with Hubble's primary mirror.

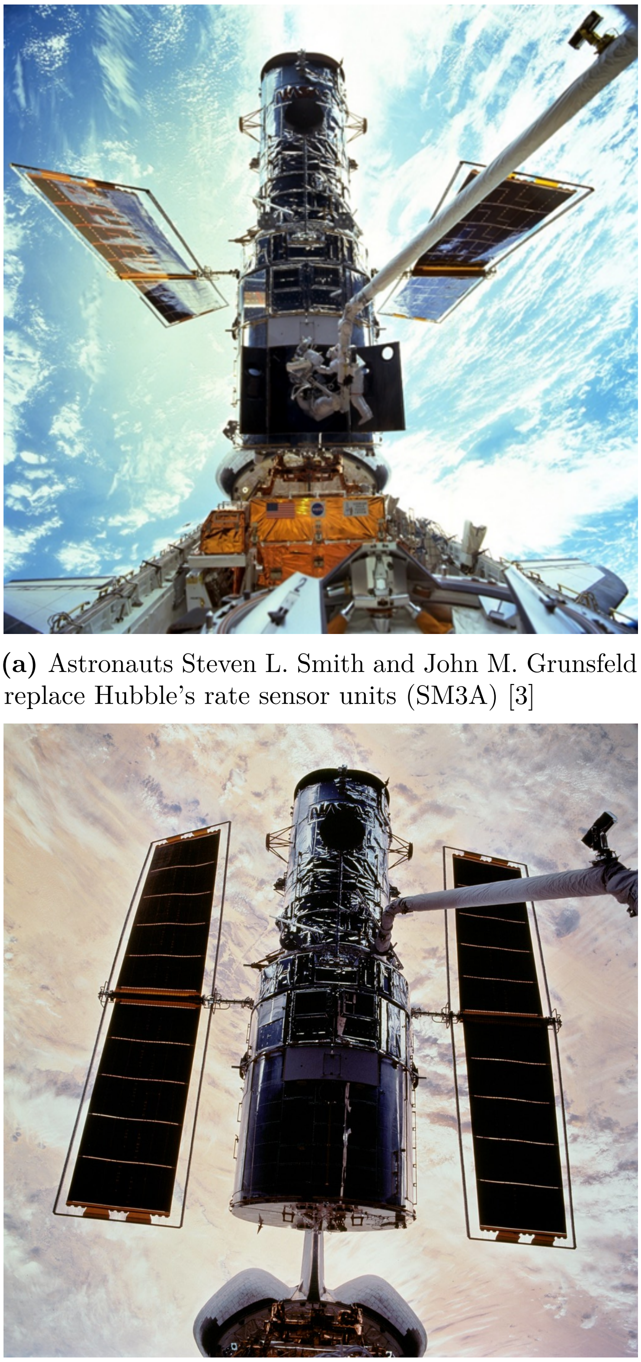
2. **Servicing Mission 2 (SM2) - February 1997**: The second servicing mission expanded the range of wavelengths visible to Hubble by installing two new instruments, ultimately enhancing the observatory's efficiency and performance.

3. **Servicing Mission 3A (SM3A) - December 1999**: Initially intended as a preventive maintenance mission, it became more urgent when the fourth of Hubble's six gyros failed. Hubble needed at least three gyros to conduct scientific work at the time, leading to the telescope entering a state of dormancy called "safe mode" while awaiting repairs. NASA divided the third servicing mission into two parts to expedite Hubble's return to operation (Figure 1.2a).

4. **Servicing Mission 3B (SM3B) - March 2002**: During SM3B, astronauts replaced Hubble's solar panels and installed the Advanced Camera for Surveys, replacing Hubble's Faint Object Camera, the telescope's last original instrument (Figure 1.2b).

5. **Servicing Mission 4 (SM4) - May 2009**: The fifth and final servicing mission left the telescope at its peak scientific capability, ensuring several more years of groundbreaking discoveries.

The European Space Agency (ESA) has been heavily involved in research on IOS as part of its **Clean Space** initiative for space debris removal and prevention, with a focus on establishing standardized servicing protocols. There are currently several



(a) Astronauts Steven L. Smith and John M. Grunsfeld replace Hubble's rate sensor units (SM3A) [3]

(b) Columbia's robotic arm dock Hubble in the shuttle's cargo bay (SM3B) [3]

Figure 1.2: Hubble Space Telescope Servicing Missions

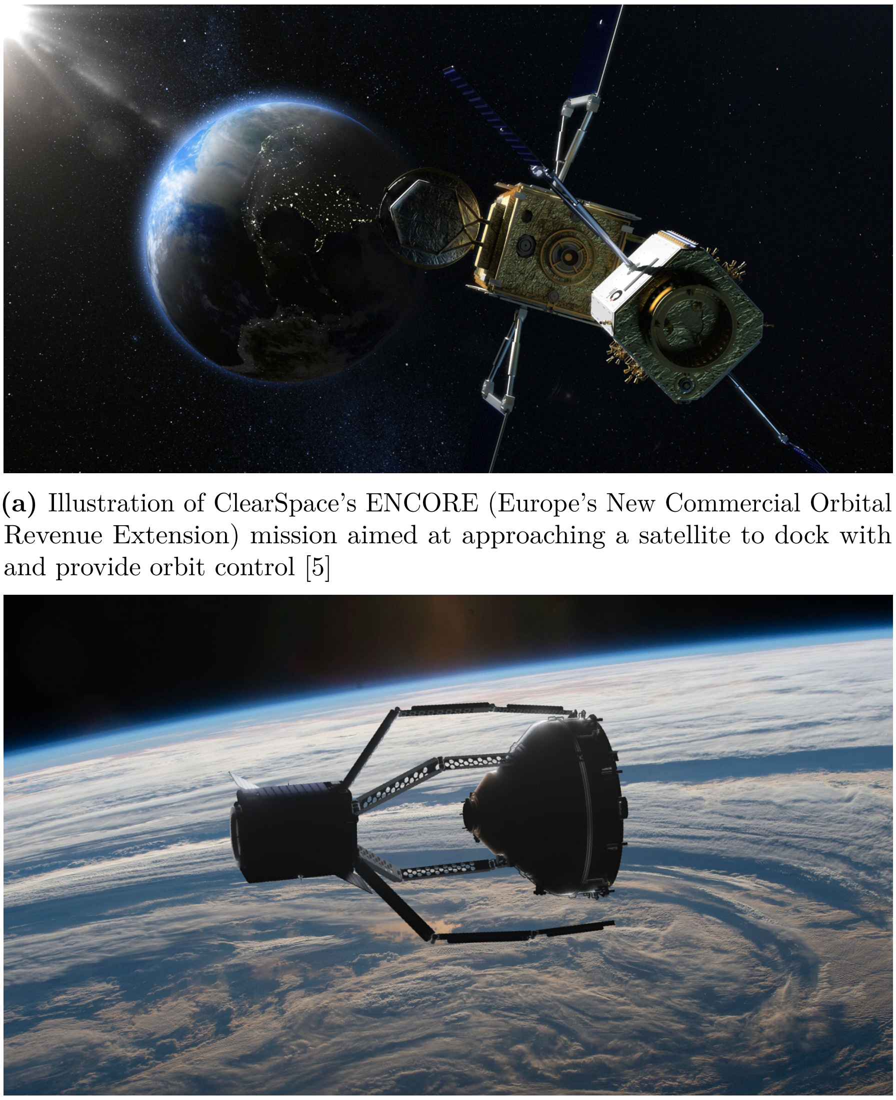
ongoing ESA-funded studies on IOS operations for satellites in low Earth orbit (LEO) and geostationary orbit (GEO). LEO is home to important satellites such as the Hubble Space Telescope, Copernicus Sentinel Earth observation satellites, and the International Space Station, while GEO hosts the Meteosat series of European weather satellites and many of the satellites used for telecommunications [2].

One clear finding from these studies is the strong interest from the telecommunications industry in having life extension services available as soon as possible. Orbital maintenance is particularly crucial, as operators need to ensure that the spacecraft remains precisely in its designated position and adjust the orbit or rotation if it has deviated over time. This process implies fuel costs, so a servicing spacecraft should be able to dock with satellites that have run out of fuel and carry out necessary orbit control. The servicing spacecraft can remain attached for as long as necessary before moving the satellite to a "graveyard orbit" and then proceeding to the next satellite in need of servicing [2].

ClearSpace-1 is the first mission of the European Space Agency (ESA) aimed at demonstrating the removal of space debris from Earth's orbit. Set for launch in 2026, the mission will rendezvous with, capture, and safely bring down for reentry the upper section of a Vespa (Vega Secondary Payload Adapter) from Europe's Vega launcher, a process commonly known as Active Debris Removal (ADR). As part of a service contract with an industrial team led by the Swiss start-up ClearSpace SA (Figure 1.3a), the mission will use ESA-developed robotic arm technology to capture the defunct rocket part, which was left in a gradual disposal orbit at an altitude of approximately 800 km by 660 km, in compliance with space debris mitigation regulations (Figure 1.3b). The mission's goal is to test the technologies necessary for debris removal as an initial step towards establishing a new and sustainable commercial space sector [4].

Not only international space agencies, but also private companies are actively working on developing servicing satellites. Northrop Grumman's **Mission Extension Vehicle-1 (MEV-1)** and **Mission Extension Vehicle-2 (MEV-2)**, launched in October 2019 and August 2020 respectively, were designed to service fully functional but ageing satellites. MEV-1 offers life-extending services by taking over the propulsion and attitude control functions to steer, point, and relocate the client spacecraft (Figure 1.4), while MEV-2 acts as a fuel source for functioning satellites without significant propulsion capabilities. Each vehicle has the capacity to perform multiple dockings and undockings during its operational lifespan. Intelsat became the first customer of the satellite life extension service, successfully docking MEV-1 to Intelsat 901 in February 2020 and MEV-2 to Intelsat 10-02 in April 2021 [7].

The Italian Space Agency (ASI) is currently working on the **In-Orbit Servicing (IOS)** demonstration mission, set to be launched in 2026. The mission will take place in LEO and is intended to test advanced technologies for future in-orbit



(a) Illustration of ClearSpace's ENCORE (Europe's New Commercial Orbital Revenue Extension) mission aimed at approaching a satellite to dock with and provide orbit control [5]

(b) Illustration of ClearSpace-1 capturing the Vespa for a controlled atmospheric reentry [6]

Figure 1.3: ESA's Space Safety Programme missions

servicing missions, including autonomous robotic operations such as refueling, component repair or replacement, orbital transfer, and atmospheric reentry.

The mission involves several important contributors [8]:

- **Thales Alenia Space** will take the lead in the mission, handling overall system design, development, and spacecraft qualification.

- **Leonardo** will be responsible for developing the robotic arm to perform

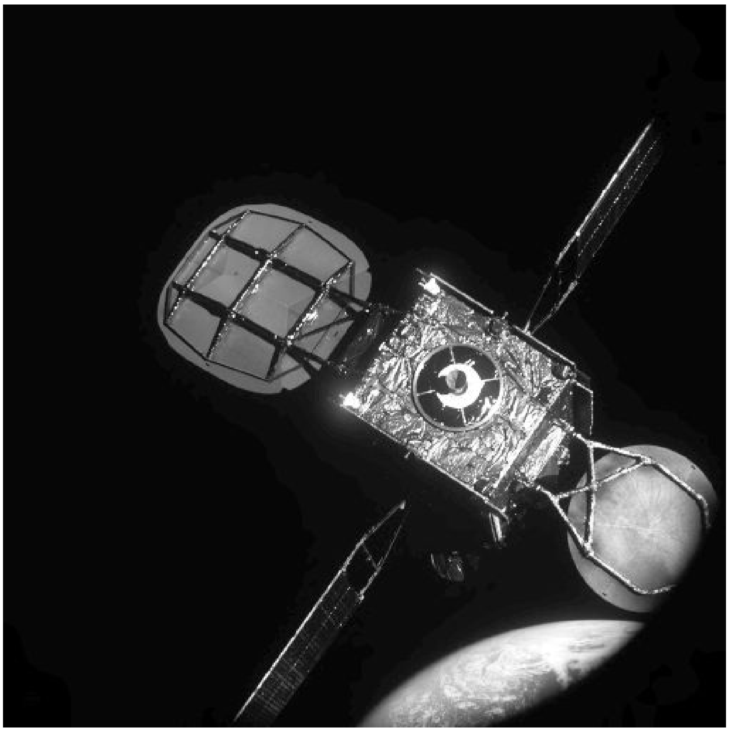


Figure 1.4: View of Intelsat 901 from MEV-1 during approach [7]

critical servicing operations, in collaboration with SAB Aerospace, the Italian National Institute for Nuclear Physics (INFN), and the Italian Institute of Technology (IIT).

- **Telespazio** and **Altec** will be in charge of designing, developing, and validating the ground segment.
- **Avio** will manage the design and development of the Orbital Support and Propulsion Module for the orbital stages.
- **D-Orbit** will oversee all activities related to the target satellite platform, which is based on the company-owned InOrbit NOW (ION) platform, and the refueling system, involving the transfer of a fluid from the operator satellite to the target satellite.

To summarize, the current state of In-Orbit Servicing is characterized by a convergence of technological innovation, commercial interest, and international collaboration. According to Ross Findlay, IOS system engineer at ESA [2]:

> *"In-Orbit Servicing could fundamentally change the way that future satellites are designed and operated. Towards the 2030s, satellites will likely*

> *need to be designed with interfaces and other features that allow service and disposal spacecraft to do their work."*

Moreover, the integration of IOS with upcoming space exploration initiatives, such as lunar and Mars missions, will further expand its potential applications. In-Orbit Servicing will not only support the maintenance and extension of satellite operations, but will also play a crucial role in establishing a sustainable human presence beyond Earth.

Chapter 2

Trajectory Optimization

This thesis addresses non-convex optimization problems which require global optimization strategies. The following chapter begins with a brief introduction to optimization and its relationship with trajectory generation problems. It then introduces some concepts about common solution approaches and finally details the methodology adopted to achieve the main optimization goal of this thesis. For more detailed information on optimization and its applications, the reader is referred to the bibliography [9, 10, 11].

2.1 Introduction to optimization

For any given system subject to specific conditions, optimization involves finding a feasible configuration among all possible ones that cannot be further improved with respect to a particular criterion. More specifically, the objective of optimization is to discover the best possible solution, often represented as a maximum or minimum, for a mathematical model within the bounds of specified constraints.

Different classes of optimization problems can be identified based on their main characteristics:

1. **Dimensionality**: Optimization problems can be classified as either finite-dimensional, if they involve a finite number of real variables, or infinite-dimensional, if they involve functions defined on functional spaces as problem variables.

2. **Convexity**: The main distinction in optimization lies between convex and non-convex problems. The convex structure makes solving the problem much easier than solving non-convex problems, as the local solution found is also the global solution. Therefore, local optimization methods are sufficient, and global strategies are not required (Figure 2.1).

3. **Number of objectives**: Optimization problems can be single-objective, involving the minimization or maximization of one objective function, or multi-objective, where multiple conflicting objectives are optimized simultaneously. Multi-objective optimization aims to find a set of solutions known as the Pareto front, where a solution is Pareto optimal if no other solution can improve one objective without worsening another.

4. **Cost function and constraints**: Another distinction can be made based on the structure of the cost function and constraints. Linear Programming (LP) problems feature linear objective functions and constraints, whereas Nonlinear Programming (NLP) problems involve either a nonlinear objective function, nonlinear constraints, or both. Quadratic Programming (QP) problems involve a quadratic objective function with linear or quadratic constraints.

5. **Type of variables**: Optimization problems can also be differentiated based on the type of decision variables. While many problems involve real-valued variables, Integer Programming (IP) problems require all variables to be integers, whereas Mixed-Integer Programming (MIP) problems require a mix of integer and real variables.

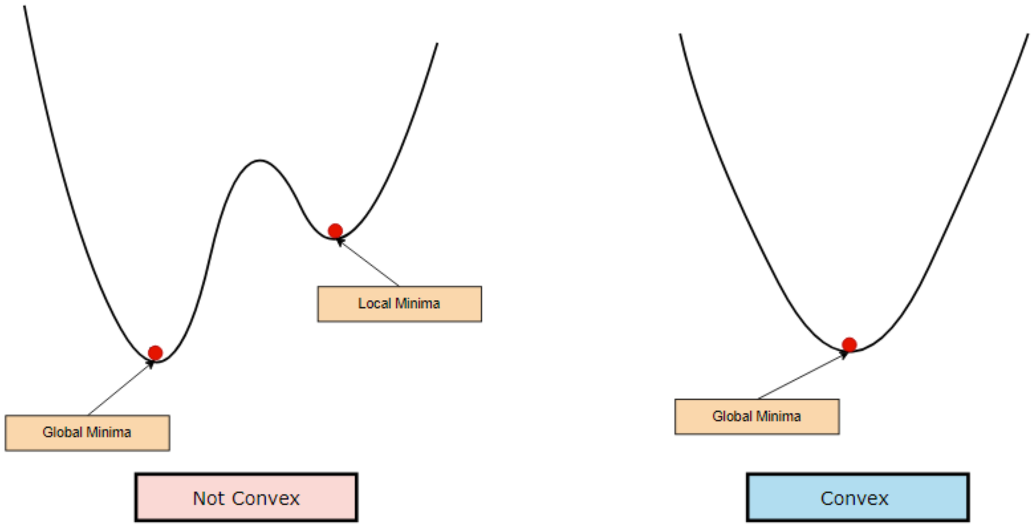


Figure 2.1: Representation of convex and non-convex functions [12]

2.2 Optimal Control Problems

Solving an *Optimal Control Problem* (OCP) involves finding a control function for a dynamical system over a specified period of time to optimize a given criterion. The state of the system and its control are time-dependent functions that need to be determined. This system is described by a set of differential equations that relate the state derivative, the state itself, and the control function. The problem includes specified initial and final state conditions, as well as possible control and path constraints. To exemplify this concept, the optimal control problem tackled in this thesis focuses on determining a trajectory that minimizes propellant consumption during a visual inspection mission. In this context, the control and trajectory functions are the unknowns to be solved through optimization.

The performance criterion is defined by an objective function that evaluates the system's behavior. This may involve maximizing or minimizing specific outcomes while considering control inputs, system dynamics, and constraints on system states and relevant variables.

Trajectory optimization involves computing a dynamically feasible state and control signal that satisfy a set of constraints while optimizing key mission objectives. The trajectory generation problem is almost always non-convex, which typically means that global optimization strategies are required to find a global optimum.

From a mathematical perspective, formulating an optimal control problem generally includes:

- a mathematical model of the system,
- a performance index specification,
- identification of free variables,
- boundary conditions for control and state variables.

The system model is often represented by a set of ordinary differential equations (ODEs):

$$\dot{\mathbf{x}}(t) = f[\mathbf{x}(t), \mathbf{u}(t), \mathbf{p}, t], \quad t \in [t_0, t_f] \tag{2.1}$$

where:

- $\mathbf{x} : [t_0, t_f] \to \mathbb{R}^{n_x}$ is the **state variable vector**,
- $\mathbf{u} : [t_0, t_f] \to \mathbb{R}^{n_u}$ is the **control variable vector**,
- $\mathbf{p} \in \mathbb{R}^{n_p}$ is the **static parameter vector**, which includes elements independent of t,

- $t \in [t_0, t_f] \in \mathbb{R}$ is the **independent variable**, usually representing time.

The starting and ending conditions can be represented by a series of inequality constraints, commonly referred to as *event constraints*:

$$\mathbf{e_L} \leq \mathbf{e}[\mathbf{x}(t_0), \mathbf{u}(t_0), \mathbf{x}(t_f), \mathbf{u}(t_f), \mathbf{p}, t_0, t_f] \leq \mathbf{e_U} \tag{2.2}$$

The problem may include time-dependent constraints on the states and/or control variables. These constraints can be formulated as time-dependent inequalities and are typically known as *path constraints*:

$$\mathbf{h_L} \leq \mathbf{h}[\mathbf{x}(t), \mathbf{u}(t), \mathbf{p}, t] \leq \mathbf{h_U}, \quad t \in [t_0, t_f] \tag{2.3}$$

Bound constraints are imposed on controls, states, and static parameters:

$$\mathbf{u_L} \leq \mathbf{u}(t) \leq \mathbf{u_U}, \quad t \in [t_0, t_f] \tag{2.4}$$

$$\mathbf{x_L} \leq \mathbf{x}(t) \leq \mathbf{x_U}, \quad t \in [t_0, t_f] \tag{2.5}$$

$$\mathbf{p_L} \leq \mathbf{p} \leq \mathbf{p_U} \tag{2.6}$$

Additionally, the initial and final times may not necessarily be fixed:

$$t_{0L} \leq t_0 \leq t_{0U} \tag{2.7}$$

$$t_{fL} \leq t_f \leq t_{fU} \tag{2.8}$$

$$t_f - t_0 \geq 0 \tag{2.9}$$

The performance index that needs to be minimized (or maximized) is known as the *objective function* J. While the specific goals may differ depending on the problem, the typical formulation, often referred to as the *Bolza cost function*, is defined as follows:

$$J_B = \varphi(\mathbf{x}_{j-1}^+, \mathbf{x}_j^-, t_{j-1}^+, t_j^-) + \sum_j \int_{t_{j-1}^+}^{t_j^-} \phi[\mathbf{x}(t), \mathbf{u}(t), t]\, dt \tag{2.10}$$

The parameters t_0 and t_f correspond to the initial and final times, respectively. The term φ stands for the *Mayer term*, which represents the cost associated with the final states. Meanwhile, ϕ is known as the *Lagrange term* or the running cost, which monitors the state and control costs across their entire time spans. The general structure of the Bolza formulation (Equation 2.10) illustrates a comprehensive cost function in optimal control theory. Depending on the particular optimization problem, the objective function might comprise solely the Mayer term, solely the Lagrange term, or both.

2.2.1 Common Solution Approaches

Different approaches are used to solve optimal control problems:

- **Direct Methods**: These methods use discretization (transcription) to transform the optimal control problem into a finite-dimensional optimization (NLP) problem and solve this directly.
- **Indirect Methods**: These methods use necessary (and sufficient) conditions for optimality (e.g., Pontryagin's maximum principle) to solve the optimal control problem in terms of a Boundary Value Problem (BVP).
- **Black-box Approaches**: These methods handle functions whose analytical expressions are not known in advance (e.g., a function that is the solution of a differential equation obtained numerically by propagation).
- **Evolutionary Algorithms**: These algorithms exploit large populations of solutions that evolve according to specific rules towards the global optimum, mimicking natural selection processes.
- **Simulation** (e.g., Single/Multiple Shooting): This approach proves to be beneficial for simple problems or when good guess solutions are available.

2.2.2 Nonlinear Programming and Transcription Methods

Nonlinear Programming (NLP) involves finding solutions to optimization problems where the objective function or some constraints are nonlinear. The optimization process involves maximizing or minimizing an objective function while satisfying a set of constraints. An NLP problem can be represented mathematically as follows:

$$\begin{aligned} \text{Minimize} \quad & J(\mathbf{y}) \\ \text{Subject to} \quad & \mathbf{h_l} \leq \mathbf{h}(\mathbf{y}) \leq \mathbf{h_u} \\ & \mathbf{y_l} \leq \mathbf{y} \leq \mathbf{y_u} \end{aligned} \tag{2.11}$$

where the **decision vector y** $\in \mathbb{R}^{n_y}$ is finite-dimensional, consisting of numerical variables and algebraic equations. By contrast, in an optimal control problem, which deals with trajectory optimization, the system is described by differential equations, and the decision variables are continuous functions defined over a time interval $[t_0, t_f]$.

Transforming a continuous optimal control problem into an NLP problem can be achieved by approximating the infinite-dimensional problem into a finite-dimensional one through a so-called *transcription method*, which involves three fundamental steps:

- Convert the dynamic system into a problem with a finite number of variables.
- Use a suitable algorithm, such as Sequential Quadratic Programming (SQP), to solve the resulting finite-dimensional NLP problem.
- Assess the accuracy of the approximation and, if necessary, repeat the aforementioned steps.

In trajectory optimization, a very popular set of transcription methods are the *shooting methods*. These are simulation-based techniques used for calculating state histories given the control histories of the system. They can be categorized into two main types [11]:

- **Direct Single Shooting Methods**: These methods transform the optimal control problem into a parameter optimization problem by parameterizing the control inputs and integrating the system's differential equations forward in time from an initial guess. The resulting optimization problem minimizes the cost function while satisfying the boundary conditions and any path constraints. However, this method can be sensitive to initial guesses and may suffer from numerical instability over long time intervals.
- **Multiple Shooting Methods**: These methods address some of the limitations of direct shooting by dividing the overall time horizon into smaller subintervals. In each subinterval, the state values at the beginning of the subinterval and the control variables are unknowns to be determined in the optimization. The dynamics are then satisfied by integrating the differential equations of motion with a time-marching algorithm, propagating the solution from one time instant to the next.

Methods for solving NLP problems are well established. Often these methods involve solving a sequence of quadratic programming sub-problems, in which the constraints are linear and the objective is a quadratic function. Popular implementations of NLP methods include SNOPT and IPOPT.

In this thesis, the Multiple Shooting Method, combined with black-box optimization techniques, is implemented in the MATLAB environment. Specifically, the `fmincon` function with the SQP algorithm, or other alternatives such as the `ga` function, which implements a genetic algorithm, were utilized to solve the associated NLP problem, as detailed in the following sections.

2.2.3 Black-box Optimization

Black-box optimization refers to the process in which there is a complete separation between the evaluation of the objective function, and possibly other functions used to impose constraints (*optimization part*), and the solution procedure (*simulation part*), as illustrated in Figure 2.2. This approach is typically used for numerical optimization problems in which the form of the objective function and its derivatives are unknown. This normally occurs when the objective function is determined through a complex simulation about which the optimization algorithm has no prior knowledge [13].

During the simulation phase, the cost function and constraints are evaluated for a given set of data, treating the problem as a black-box. The output generated from this black-box is then used as input for the optimization phase, where an algorithm iteratively adjusts the value of the control variables vector **u**. This iterative process continues until certain criteria are satisfied, ideally resulting in an optimal solution.

The black-box approach simplifies the problem by reducing the number of input variables. For example, in the case of a high-thrust trajectory, this can be divided into a series of ballistic arcs, leading to easier parameterization.

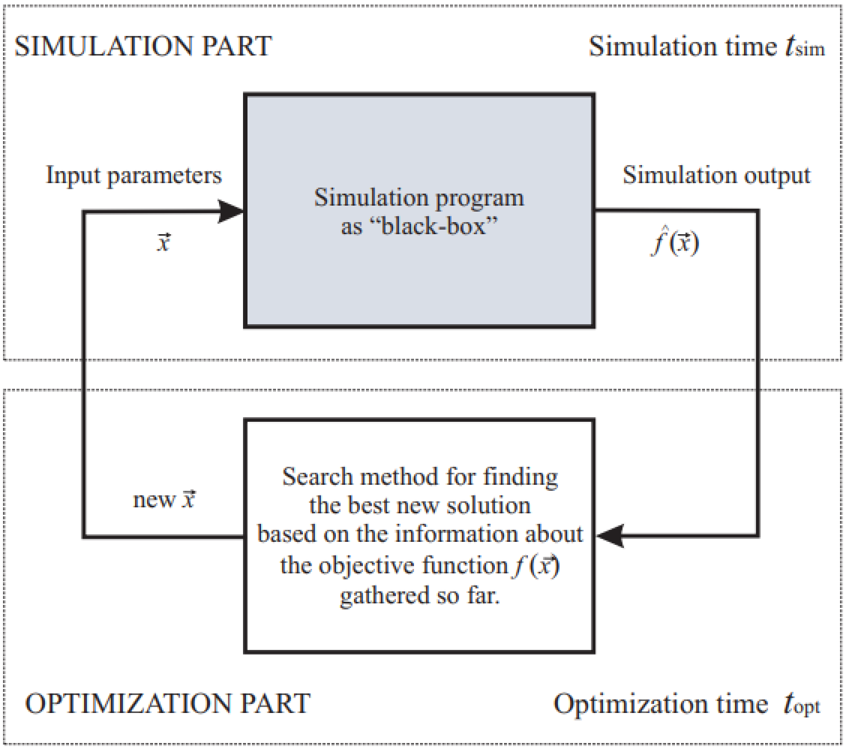

SIMULATION PART
Simulation time t_{sim}
Input parameters
$\vec{x}$
Simulation program as "black-box"
Simulation output
$\hat{f}(\vec{x})$
new $\vec{x}$
Search method for finding the best new solution based on the information about the objective function $f(\vec{x})$ gathered so far.
OPTIMIZATION PART
Optimization time t_{opt}

Figure 2.2: Schematic representation of the black-box optimization framework [13]

2.3 Adopted Optimization Methods

This section outlines the solution approaches and optimization algorithms utilized in this thesis. The aim is to develop a flexible tool capable of finding the global optimum or, at a minimum, a highly effective solution across diverse scenarios with different operational sequences and constraints. While strict adherence to some constraints was required, others allowed for greater flexibility.

To achieve this goal, a hybrid approach was adopted, integrating black-box optimization with the Multiple Shooting Method. The solution algorithm combines a traditional Multistart method with Sequential Quadratic Programming (SQP). Additionally, an evolutionary algorithm (Genetic Algorithm) is employed for the refinement process.

2.3.1 Multiple Shooting Method

As mentioned earlier, Direct Shooting Methods treat the trajectory as a single, continuous entity. Starting from the initial conditions and optimization variables, the trajectory is integrated, and the final conditions are fed back to the optimizer to adjust the variables. This approach can lead to issues with optimization algorithms, as small variations in initial variables can be amplified throughout the trajectory, potentially resulting in significant deviations due to the highly nonlinear nature of dynamic functions [14].

To mitigate these issues, Multiple Shooting Methods were developed. These methods divide the trajectory into several segments, each integrated separately. By breaking the trajectory into smaller segments, the amplification of small variations is reduced. The initial states at the beginning of each segment are included in the set of initial optimization variables. Multiple Shooting Methods transform the optimal control problem with boundary conditions (BVP) into several initial conditions problems. The goal is to estimate the initial conditions for each segment, ensuring they meet constraints throughout the propagation interval $[t_0, t_f]$ and satisfy final state constraints, while optimizing a specified performance index. The term "multiple shooting" originates from the division of the trajectory into smaller segments and solving for the initial conditions of each segment [14].

The implementation of the Multiple Shooting Algorithm involves the following steps [14]:

- Divide the trajectory into multiple segments, splitting the initial time interval $[t_0, t_f]$ into $N-1$ smaller intervals:

$$t_0 < t_1 < \cdots < t_i < \cdots < t_N = t_f \tag{2.12}$$

The state vector values at the beginning of each segment, $\tilde{x}_2, \ldots, \tilde{x}_N$, along with the control variables $u_1, \ldots, u_N$, are unknowns to be determined in the optimization process.

- Guess the initial conditions for each segment $z_i = x_i(t_0)$.
- Propagate the system state trajectory for each segment using a numerical propagator, if an analytical solution is unavailable.
- Match the end of one segment with the beginning of the next to ensure continuity.
- Verify that the cost function $f(z)$ is minimized and that the boundary conditions are satisfied, adjusting the initial conditions as needed by solving an NLP problem.

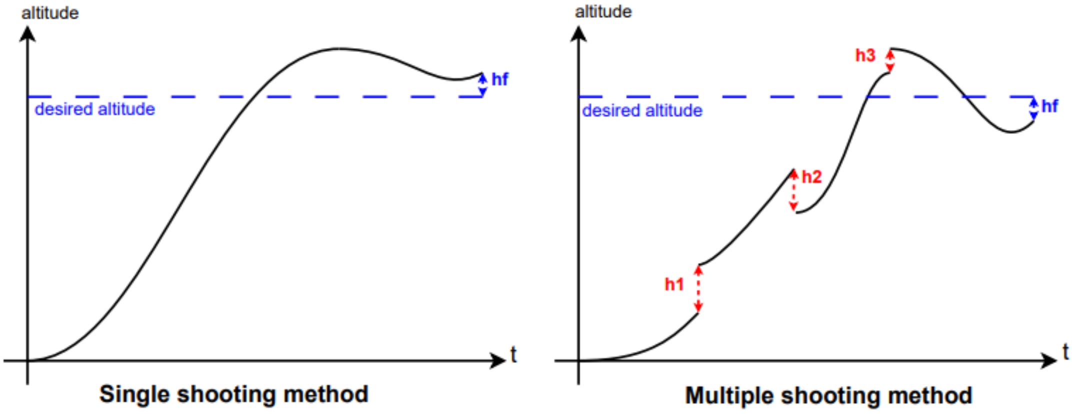


Figure 2.3: Single Shooting vs Multiple Shooting methods for a four-section trajectory: the optimization algorithm aims to minimize the error (or defect h) to zero [14]

While Single Shooting is effective for simple problems, it often fails in more complex scenarios. Multiple Shooting is more robust for such cases, as it divides the problem into smaller sub-problems, solves them individually, and ensures that each segment connects smoothly to the previous one.

An alternative approach is *Direct Collocation*, which approximates the trajectory using a piecewise polynomial. The physical laws are satisfied by ensuring that the dynamics (state derivatives) match the polynomial derivatives at each collocation point, which are the points that define the polynomial. Essentially, shooting methods use explicit integration to satisfy dynamics, whereas collocation methods use implicit integration [14].

There are various collocation schemes. Orthogonal collocation, often used for computing satellite trajectories, employs a few high-order polynomials and is highly accurate for problems with well-behaved solutions. On the other hand, direct collocation uses many low-order polynomials, which, while slightly less accurate, can be more robust. Advanced methods like GPOPS employ adaptive meshing to iteratively adjust the number of segments and polynomial order.

2.3.2 Multistart and SQP

Multistart combined with Sequential Quadratic Programming (SQP) is an optimization strategy used to find the global minimum of a nonlinear and nonconvex objective function by leveraging the strengths of a local optimization method. The nonconvex nature of the function necessitates a global approach, as relying solely on a local search algorithm is insufficient for identifying the global optimum.

Multistart is a global optimization technique that involves running a local optimization algorithm multiple times from various initial points in the search space. This approach aims to explore a broader region of the solution space, thereby increasing the likelihood of finding a global minimum.

SQP is a local optimization method designed for solving nonlinear-constrained optimization problems. The algorithm utilizes a local quadratic approximation to model the nonlinear objective function and constraints.

After executing SQP from multiple starting points, the best solution obtained from all runs is considered the potential global minimum. The combination of Multistart and SQP is advantageous due to its simplicity; however, the random selection of starting points may be insufficient for solving more challenging problems.

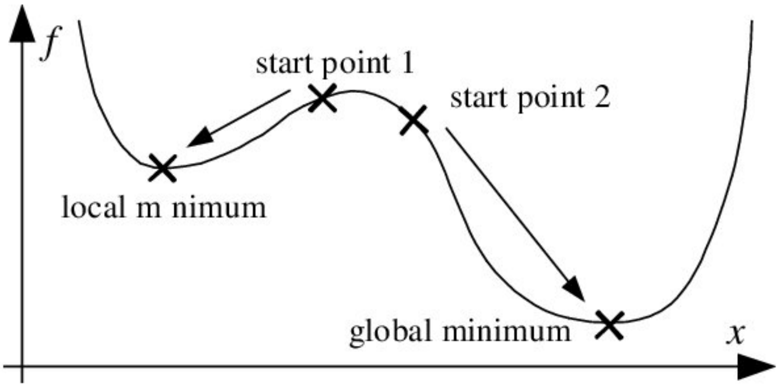


Figure 2.4: Principle of the Multistart algorithm [15]

2.3.3 Genetic Algorithm

Genetic Algorithms (GAs) are optimization techniques inspired by natural selection, frequently employed to tackle complex optimization and search problems by emulating biological evolution.

In this thesis, GAs were utilized to refine solutions by iteratively enhancing the population of potential solutions, guiding its evolution towards an optimal or near-optimal state. The main steps involved in this process are as follows:

1. **Initialization**: A diverse population of potential solutions is generated, either randomly or based on informed guesses.
2. **Evaluation**: Each solution's fitness is assessed by computing its cost function. If constraints are violated, a penalty is added to the cost.
3. **Selection**: Solutions are chosen for reproduction based on their fitness, with higher fitness solutions having a greater chance of being selected.
4. **Crossover**: Selected solutions exchange genetic information to create new solutions, mimicking biological recombination.
5. **Mutation**: Random changes are introduced to some solutions to maintain diversity and explore the search space.
6. **Replacement**: The new generation consists of offspring and some existing solutions, replacing the least fit solutions to maintain population size.
7. **Termination**: The process repeats until a predefined number of generations is reached or a satisfactory solution is found.

GAs are particularly advantageous for their ability to handle complex, multimodal landscapes where traditional optimization methods might struggle. Their stochastic nature allows them to avoid local minima, increasing the likelihood of finding a global optimum. However, the effectiveness of GAs can depend on several factors, including the size of the population, the rates of crossover and mutation, and the selection method. Careful tuning of these parameters is crucial for achieving optimal performance.

2.3.4 Global Optimization

The discussed algorithms, Multistart + SQP and GAs, are part of two overall optimization strategies. An important consideration is that these are not true Global Optimization Algorithms. True Global Optimization Algorithms, which guarantee finding the absolute best solution, do not exist. Instead, we use globalization strategies, such as the Multistart function from MATLAB's Global Optimization Toolbox, to handle these specific problems effectively.

Chapter 3

Mathematical Model

In this chapter, the physical model of the system is examined in detail, with the derivation of the general equations of relative motion for circular Target orbits. Then, the problem of in-orbit Target inspection is explained. Since spacecraft are considered as point masses in this thesis, only forces are examined, while moments that may affect spacecraft orientation are excluded from this work. For more detailed information, the reader is referred to the bibliography [16].

3.1 Reference Frame

For the analysis of inspection and docking trajectories, it is best to use a reference frame that originates at the center of mass (CoM) of the target vehicle, i.e., to observe the movement of the tracker as an astronaut sitting in the target vehicle would. This frame is the Spacecraft Local Orbital frame (F_{lo}), often referred to as the Local-Vertical/Local-Horizontal (LVLH) frame, and is employed in this thesis to describe the motion of the Servicer with respect to the Target during all phases of the mission. Operations in the F_{lo} frame require a precise understanding of relative dynamics and orbital perturbations that can affect the spacecraft.

In Figure 3.1, the LVLH frame is represented in terms of:

- $\boldsymbol{O}_{\mathbf{LVLH}}$: origin coinciding with the CoM of the Target;
- $\boldsymbol{X}_{\mathbf{LVLH}}$ (**V-bar**): in-track axis, tangential to the Target's orbit, pointing in the direction of the orbital velocity vector;
- $\boldsymbol{Y}_{\mathbf{LVLH}}$ (**H-bar**): cross-track axis, pointing outward in the opposite direction of the orbit's angular momentum vector;
- $\boldsymbol{Z}_{\mathbf{LVLH}}$ (**R-bar**): radial axis, pointing from the Target CoM to the center of the Earth.

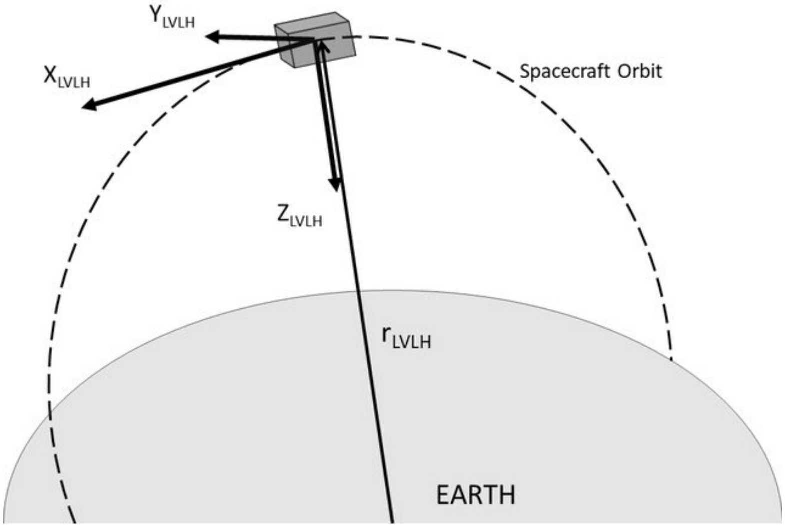


Figure 3.1: LVLH frame definition [17]

3.2 Orbit Dynamics

3.2.1 Orbital Motion around a Central Body

The fundamental force acting on the spacecraft is the gravitational attraction of the central body, which can be expressed by Newton's law of universal gravitation. For the case of an undisturbed spherical gravitational field, Newton's law of gravitation:

$$\mathbf{F} = -\frac{Gm_c m_s}{r^2} \tag{3.1}$$

can be combined with his second law, relating force and acceleration:

$$\mathbf{F} = m_s \ddot{\mathbf{r}} \tag{3.2}$$

The resulting equation describes the orbital motion of a satellite around a central body:

$$\ddot{\mathbf{r}} = -\frac{\mu}{r^2} \tag{3.3}$$

where:

- $G = 6.674 \times 10^{-11}\,\text{N m}^2/\text{kg}^2$ is the universal gravitational constant;
- m_c is the mass of the central body;
- m_s is the mass of the spacecraft;

- $\boldsymbol{r}$ is the distance between their centers;
- $\mu = G \cdot m_c$ is the gravitational constant of the central body.

The resulting motion leads to elliptical orbits, with special cases of circular, parabolic, and hyperbolic trajectories depending on the spacecraft's energy [16].

3.2.2 Equations of Relative Motion

The equations discussed in Section 3.2.1 are given in the orbital plane frame. Equations of motion in this frame can be conveniently used for trajectory analysis until the chaser vehicle is in the close vicinity of the Target. For relative navigation, it becomes more convenient to keep one of the spacecraft as a fixed point. The resulting frame is the Spacecraft Local Orbital frame of the Target, F_{lo}, defined in Section 3.1.

Hill's equations describe the relative motion of two nearby orbiting objects in a circular orbit around a central body in the LVLH frame:

$$\ddot{x} - 2\omega\dot{z} = \gamma_x \tag{3.4}$$

$$\ddot{y} + \omega^2 y = \gamma_y \tag{3.5}$$

$$\ddot{z} + 2\omega\dot{x} - 3\omega^2 z = \gamma_z \tag{3.6}$$

where:

- x, y, z are the relative position coordinates of the Servicer with respect to the Target;
- $\dot{x}, \dot{y}, \dot{z}$ are the relative velocity components;
- ω is the mean motion of the Target orbit;
- m_c is the mass of the Servicer;
- $\gamma_{x,y,z} = \frac{F_{x,y,z}}{m_c}$ are the imposed accelerations acting on the Servicer, whether resulting from thruster activities or from external disturbances.

A linearized solution to the relative motion problem has been derived from Hill's equations by W. H. Clohessy and R. S. Wiltshire. The main assumption of the Clohessy–Wiltshire (CW) equations is that the relative distances between the Servicer and Target vehicles are very small compared with the distance to the center of the Earth. For impulsive thrust maneuvers and constant accelerations over the time period considered, the resulting equations of motion are:

$$x(t) = \left(\frac{4\dot{x}_0}{\omega} - 6z_0\right)\sin(\omega t) - \frac{2\dot{z}_0}{\omega}\cos(\omega t) + (6\omega z_0 - 3\dot{x}_0)t + \left(x_0 + \frac{2\dot{z}_0}{\omega}\right) + \dots$$
$$+ \frac{2}{\omega^2}\gamma_z(\omega t - \sin(\omega t)) + \gamma_x\left(\frac{4}{\omega^2}(1-\cos(\omega t)) - \frac{3}{2}t^2\right) \tag{3.7}$$
$$y(t) = y_0\cos(\omega t) + \frac{\dot{y}_0}{\omega}\sin(\omega t) + \frac{\gamma_y}{\omega^2}(1-\cos(\omega t)) \tag{3.8}$$
$$z(t) = \left(\frac{2\dot{x}_0}{\omega} - 3z_0\right)\cos(\omega t) + \frac{\dot{z}_0}{\omega}\sin(\omega t) + \left(4z_0 - \frac{2\dot{x}_0}{\omega}\right) + \dots$$
$$+ \frac{2}{\omega^2}\gamma_x(\sin(\omega t) - \omega t) + \frac{\gamma_z}{\omega^2}(1-\cos(\omega t)) \tag{3.9}$$

However, because of the linearization, the accuracy of the CW equations decreases with the distance from the origin of the reference frame.

3.3 Trajectory Types

In order to meet mission objectives, different types of trajectories may be utilized, each with specific properties. These trajectories can be categorized into three main types [16]:

- **Free drift motions**: These occur when the spacecraft moves without active propulsion, following a natural trajectory determined by its initial conditions for position and velocity. During free drift, the spacecraft's motion is governed by the laws of orbital mechanics, and its trajectory can be precisely predicted if the initial state is known. Accurate modeling of free drift trajectories requires consideration of perturbative forces such as atmospheric drag, solar radiation pressure, and geopotential anomalies (the J_2 effect).
- **Impulsive maneuvers**: These involve rapid changes in the spacecraft's velocity, resulting in an instantaneous ΔV achieved through short bursts of thrust. Common examples include Hohmann transfer orbits and bi-elliptic transfers. These maneuvers are efficient in terms of propellant usage and are executed in a very short time relative to the orbit period. The impulsive model is the standard procedure used to simulate spacecraft maneuvers with low specific impulse and high thrust.
- **Continuous thrust maneuvers**: These are characterized by a prolonged application of thrust in order to achieve a certain ΔV, resulting in gradual changes to the spacecraft's trajectory. This approach is advantageous for missions requiring fine adjustments or long-duration orbit transfers, and is often used for low-thrust propulsion systems, such as electric propulsion

systems, which provide higher efficiency but lower thrust compared to chemical propulsion.

3.3.1 Impulsive Maneuvers

The mathematical modeling of the problem in this thesis is based on impulsive maneuvers. Hence, in this subsection, we present an overview of the most common ideal maneuvers used in rendezvous and proximity operations, along with their corresponding accelerations and ΔV requirements. This overview is essential for understanding the relative motion of the Servicer in relation to the Target and evaluating the performance of different maneuvers. The described trajectories form the basis for the mission's maneuver planning and are considered in an ideal scenario without disturbances (see, e.g., [16]).

3.3.2 Tangential Thrust Maneuvers

A tangential boost initiates the transfer along V-bar (Figure 3.2). If the ΔV is applied in the direction of the velocity (positive boost), the spacecraft accelerates, increasing its altitude and moving away from the Target. Conversely, a negative ΔV brings the spacecraft closer to the Target. After one full orbital period T, the Servicer's displacement is:

$$\Delta x = \frac{6\pi}{\omega}\Delta V_x \tag{3.10}$$

where $\omega = \frac{2\pi}{T}$ is the Target's angular velocity. The ΔV_x must be applied twice, at the beginning and end, to complete the maneuver. The total ΔV required is:

$$\Delta V = \omega\frac{3\pi\Delta x}{T} \tag{3.11}$$

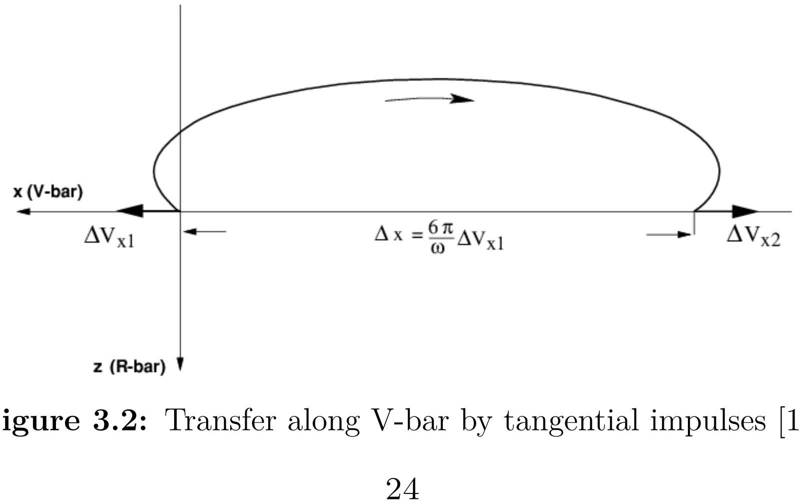


Figure 3.2: Transfer along V-bar by tangential impulses [16]

For a Hohmann transfer (Figure 3.4), which changes the spacecraft's orbit altitude, the relationship between the displacement along the x-axis and z-axis is:

$$\Delta x = \frac{3\pi}{4}\Delta z \tag{3.12}$$

This relationship helps determine the ΔV cost for each tangential boost. In CubeSat missions, continuous thrust can replace impulsive boosts (Figure 3.5), requiring double the time to complete the transfer:

$$\gamma_x = -\omega^2 \frac{4\pi\Delta z}{T} \tag{3.13}$$

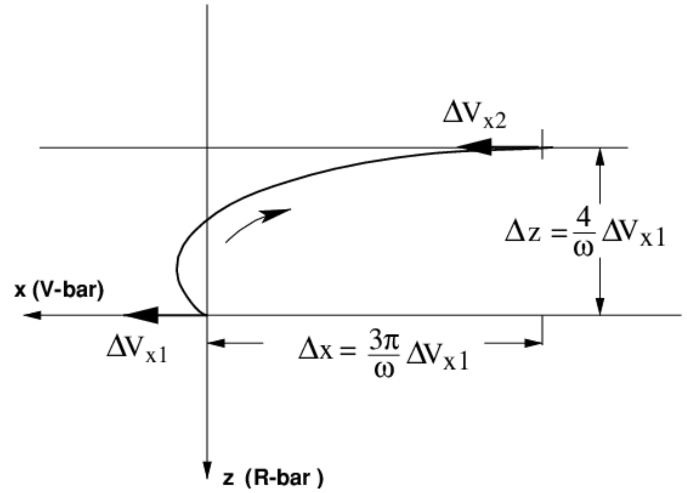

ΔV_{x2}
$\Delta z = \frac{4}{\omega}\Delta V_{x1}$
x (V-bar)
ΔV_{x1}
$\Delta x = \frac{3\pi}{\omega}\Delta V_{x1}$
z (R-bar)

Figure 3.3: Transfer to orbit of different height (Hohmann transfer) [16]

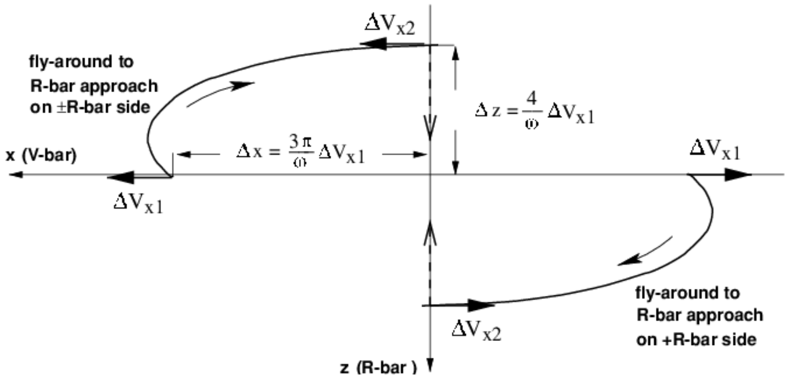


Figure 3.4: Fly-around by tangential impulse [16]

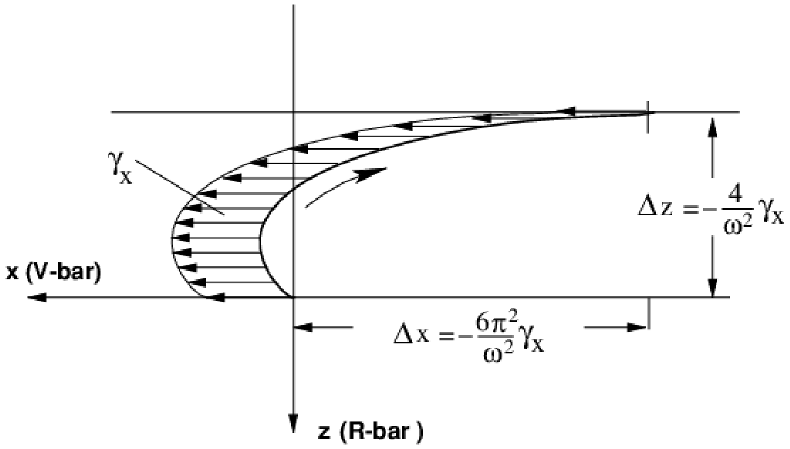

γ_x
x (V-bar)
$\Delta z = -\frac{4}{\omega^2}\gamma_x$
$\Delta x = -\frac{6\pi^2}{\omega^2}\gamma_x$
z (R-bar)

Figure 3.5: Continuous x-thrust transfer to different altitude [16]

3.3.3 Radial Thrust Maneuvers

A common maneuver to approach a Target in the same orbit is to apply thrust in the radial direction (Figure 3.6). This maneuver requires an impulsive boost at the beginning and end to stop the motion. The cost for each boost is given by:

$$\Delta V = \omega \frac{4\Delta x}{T} \tag{3.14}$$

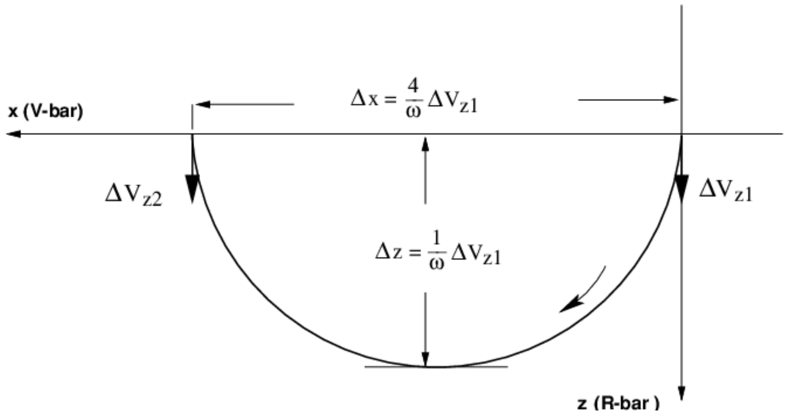

x (V-bar)
$\Delta x = \frac{4}{\omega}\Delta V_{z1}$
ΔV_{z2}
ΔV_{z1}
$\Delta z = \frac{1}{\omega}\Delta V_{z1}$
z (R-bar)

Figure 3.6: Transfer along V-bar by radial impulses [16]

This maneuver can also be executed continuously if thrust limitations are present, lasting an entire orbital period (Figure 3.8). The required thrust per unit mass is:

$$\gamma_z = \omega^2 \frac{4\pi\Delta x}{T} \tag{3.15}$$

where Δx is the displacement along the x-axis. The ΔV is then:

$$\Delta V = \gamma_z T = \omega^2 \Delta x \tag{3.16}$$

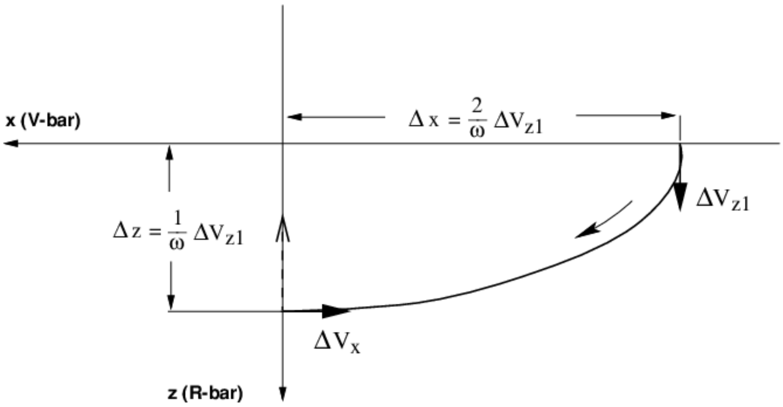

x (V-bar)
$\Delta x = \frac{2}{\omega} \Delta V_{z1}$
ΔV_{z1}
$\Delta z = \frac{1}{\omega} \Delta V_{z1}$
ΔV_x
z (R-bar)

Figure 3.7: Fly-around by radial impulse [16]

This maneuver is advantageous for safety compared to tangential maneuvers. If a malfunction occurs, the Servicer remains at a constant distance from the Target, avoiding collision risks. However, it is more expensive, requiring $\frac{3\pi}{2}$ times the ΔV of a tangential transfer. Real missions, such as ATV, HTV, and SSO, often prefer the safer but more costly radial maneuver.

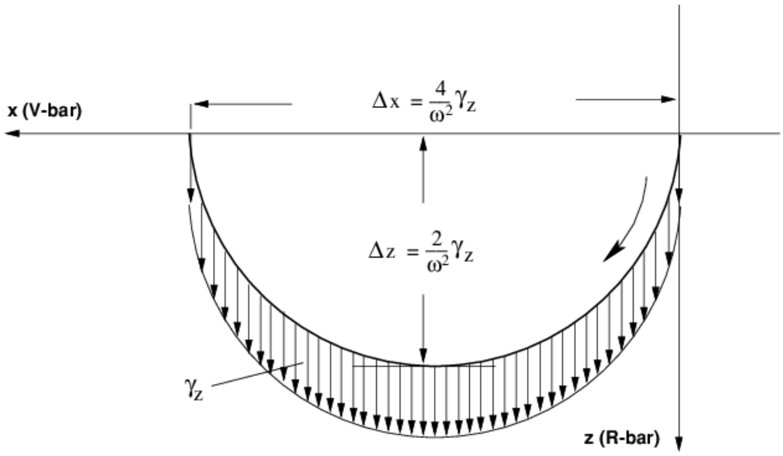

x (V-bar)
$\Delta x = \frac{4}{\omega^2} \gamma_z$
$\Delta z = \frac{2}{\omega^2} \gamma_z$
γ_z
z (R-bar)

Figure 3.8: Transfer along V-bar by continuous z-thrust [16]

The result of an out-of-plane impulse is a pure sinusoidal motion starting with $y_0 = 0$ (Figure 3.9). The displacement in the y-direction, for the same impulse, is equivalent to the z-displacement resulting from a radial impulse. The out-of-plane impulse does not cause displacements in other directions, as this motion is decoupled from the in-plane motions.

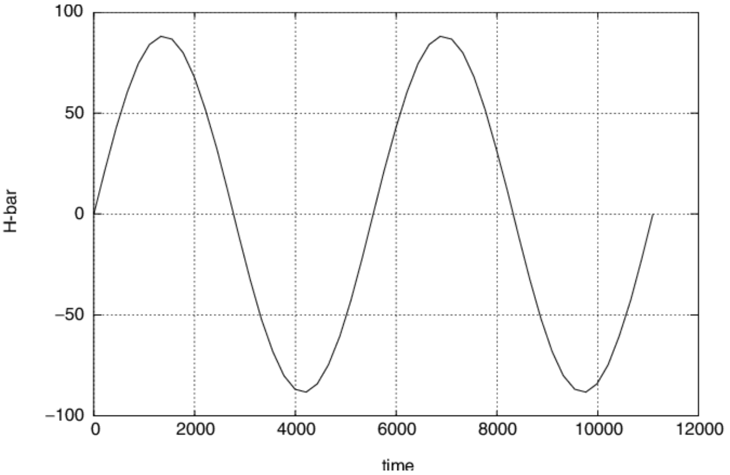


Figure 3.9: Out-of-plane impulse resulting in sinusoidal motion [16]

This decoupling simplifies the trajectory planning process, as out-of-plane maneuvers can be independently optimized and added to in-plane maneuvers (such as tangential and radial thrust maneuvers) discussed earlier. This independent adjustment enhances the flexibility and precision of orbital corrections, enabling comprehensive three-dimensional control over the spacecraft's path. Additionally, out-of-plane maneuvers are crucial for maintaining or adjusting the spacecraft's inclination without affecting its in-plane trajectory, which is particularly beneficial for certain mission objectives.

3.4 Environmental Disturbances

This section outlines the main trajectory disturbances acting on a spacecraft orbiting Earth. Of all the disturbances, only atmospheric drag is implemented in the developed optimization tool. The other disturbances are mentioned for the sake of completeness and for further development but are not included in the current implementation (see, e.g., [16]).

3.4.1 Atmospheric Drag

Drag forces from the residual atmosphere act opposite to the velocity vector, causing deceleration and orbital decay of the spacecraft, and are relevant for low Earth orbits (Figure 3.10):

$$\mathbf{F_D} = -\frac{\rho}{2}\mathbf{V_x}^2 C_D A \tag{3.17}$$

where:

- $\mathbf{V_x} = \omega \mathbf{r}$ is the orbital velocity,
- C_D is the drag coefficient,
- A is the cross-sectional area of the body.

As both vehicles are affected by drag, and since their absolute velocity difference is negligible, the differential drag force per unit mass $\gamma_D = \frac{F_d}{m}$ acting on the Servicer relative to the Target in a circular orbit is given by:

$$\Delta\gamma_D = \gamma_{Dc} - \gamma_{Dt} = -\frac{\rho}{2}\omega^2 r^2 \left(\frac{C_{Dc}A_c}{m_c} - \frac{C_{Dt}A_t}{m_t}\right) \tag{3.18}$$

where m is the mass of the vehicles and the indices c and t denote Chaser and Target, respectively. The ballistic coefficient of the vehicle, denoted as $C_B = \frac{m}{C_D A}$, simplifies the equation to:

$$\Delta\gamma_D = -\frac{\rho}{2}\omega^2 r^2 \frac{1}{C_{Bc}}\left(1 - \frac{C_{Bc}}{C_{Bt}}\right) \tag{3.19}$$

The impact of this disturbance on the trajectory can be calculated by substituting $\Delta\gamma_D$ as γ_x into Equation 3.4. For a more detailed model, not only the vehicle's cross-section but also the individual surfaces and their orientation relative to the orbital velocity vector must be considered (Figure 3.11).

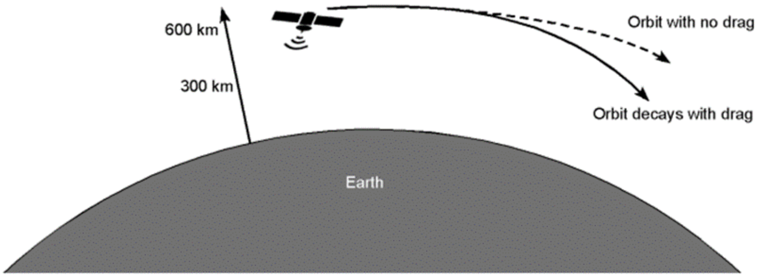


Figure 3.10: Spacecraft in LEO encounter drag from the residual atmosphere, causing their orbits to decay [18]

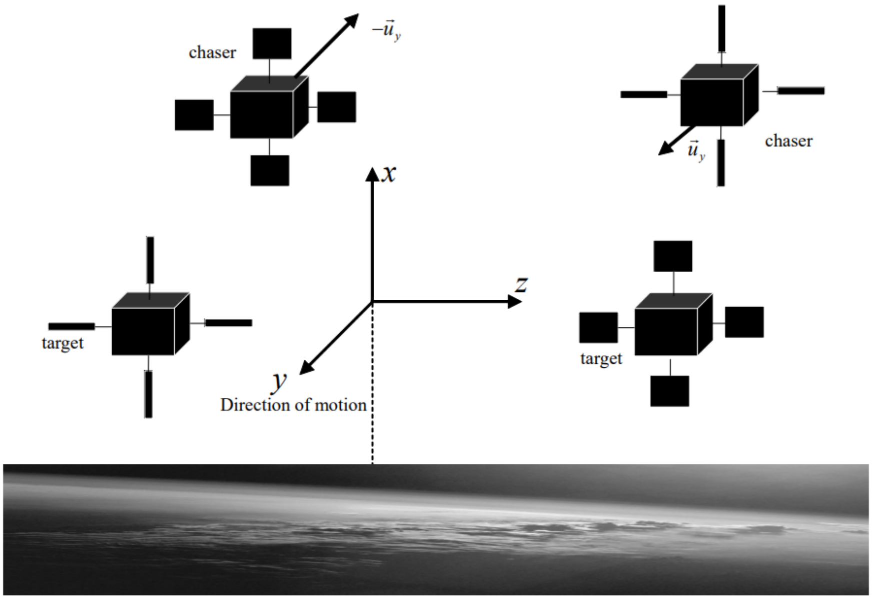


Figure 3.11: Conceptual explanation of differential drag control: if the Chaser deploys its drag plates, it experiences a relative deceleration compared to the Target (left); if the Target deploys its drag plates, it causes a relative acceleration of the Chaser with respect to the Target (right) [19]

3.4.2 Solar Radiation Pressure

This force results from photons applying pressure on the spacecraft's exposed surfaces, such as solar panels. Solar radiation produces a force on a spacecraft in the Sun–satellite direction:

$$\mathbf{F}_{\mathrm{SP}} = -\mathbf{p} \cdot A \cdot \mathbf{u}_S \tag{3.20}$$

where:

- $\mathbf{p}$ is the radiation momentum flux,
- A is the cross-sectional area of the satellite,
- $\mathbf{u}_S$ is the Sun–satellite direction unit vector.

3.4.3 Magnetic Force

This force is generated by the spacecraft's interaction with the Earth's magnetic field. Because the Earth is not a perfect sphere and its mass is unevenly distributed, these forces fluctuate throughout an orbital period. The Earth's gravitational potential can be approximated by a specific function:

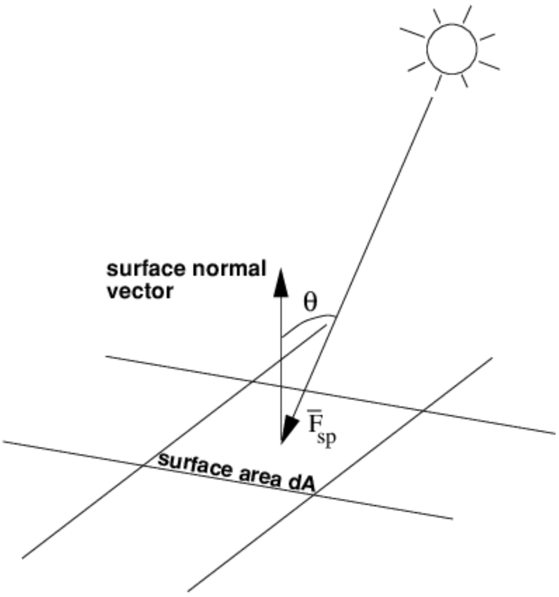


Figure 3.12: Solar pressure force on the satellite surface [20]

$$\Phi = \frac{\mu}{r}\left(1 - \sum_{n=2}^{\infty} J_n \left(\frac{R_E}{r}\right)^n P_n(\sin\phi)\right) \tag{3.21}$$

where:

- J_n are the harmonic coefficients of the potential,
- R_E is the Earth's radius at the equator,
- r is the distance of the satellite from the center of the Earth,
- P_n are the Legendre polynomials,
- ϕ is the latitude,
- μ is the gravitational constant of the Earth.

Chapter 4

Simulation Results

With the problem's physics and optimization strategy now defined, the next step is to set up the optimization problem, which involves determining the trajectory of a servicer spacecraft that needs to inspect and then dock with a target satellite. To achieve this, it is crucial to identify the objective function, control variables, and constraints that will best address the optimization problem. Subsequently, a comprehensive overview of the conducted analyses will be presented. The results demonstrate the effectiveness of the proposed optimization tool and provide insights into its practical applications.

4.1 Optimization Problem Setup

4.1.1 Objective Function

The objective of the developed optimization tool is to minimize the sum of the velocity increments to reduce propellant consumption:

$$J = \sum_{i=1}^{N} |\Delta V_i| \tag{4.1}$$

where ΔV_i represents the i-th impulse along the three axes (V-bar, H-bar, R-bar), and N is the total number of impulsive maneuvers executable by the Servicer. N is assigned at the beginning of each simulation, so it is not a free variable. Nevertheless, this constraint is effectively addressed by the optimizer, as will be demonstrated later.

4.1.2 Control Variables

The control variables chosen for this optimization problem include:

- $\boldsymbol{\Delta V}$: This is the velocity increment associated with each boost. It is a vector of size $N \times 3$, where N is the total number of impulses (assigned) and 3 represents the thrust components along each axis (V-bar, H-bar, R-bar).
- $\boldsymbol{\Delta t}$: This is the time interval between each boost. It is a vector of size $(N-1)$, where $N-1$ is the total number of segments into which the trajectory is divided.
- $\boldsymbol{A_c}$: This is the drag area of the Servicer. It is a vector of size 1.

All these variables are optimized to minimize the total velocity increment while ensuring the Servicer follows the intended trajectory around the Target. Two important points need to be highlighted. Firstly, in the impulsive model, the satellite's maneuver is considered as a sudden increase in velocity ($\Delta V > 0$) with zero thrust time ($\Delta t_{\Delta V} = 0$). Each trajectory segment (from t_i to t_{i+1}) corresponds to an impulsive ΔV (Figure 4.1), which can be analyzed using Keplerian propagation with respect to the primary body. For this reason, if no maneuver is needed at the beginning of a segment, the optimizer sets the corresponding ΔV to zero. In this way, the optimization of the number of impulses N is also addressed.

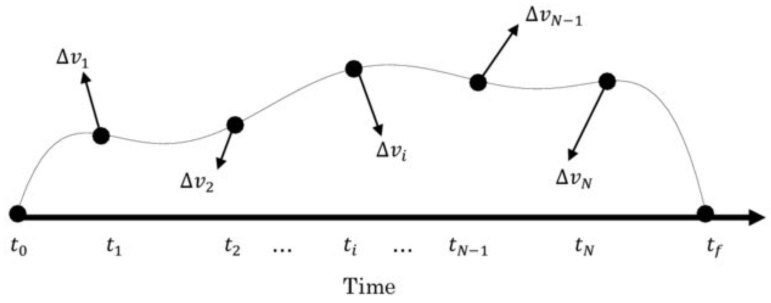


Figure 4.1: Impulsive discretization scheme [11]

Secondly, the drag area of the Servicer, as a control variable, directly affects the satellite's ballistic coefficient CB_c. As illustrated in Figure 4.2, by adjusting the drag area, the ballistic coefficient can be optimized. This optimization allows the Servicer to efficiently approach or retreat from the Target along the V-bar direction, thereby reducing propellant usage.

4.1.3 Constraints

The data listed in Table 4.1 outline the characteristics and initial orbit parameters for both the Servicer and the Target, with the latter modeled as a cubic-shaped satellite.

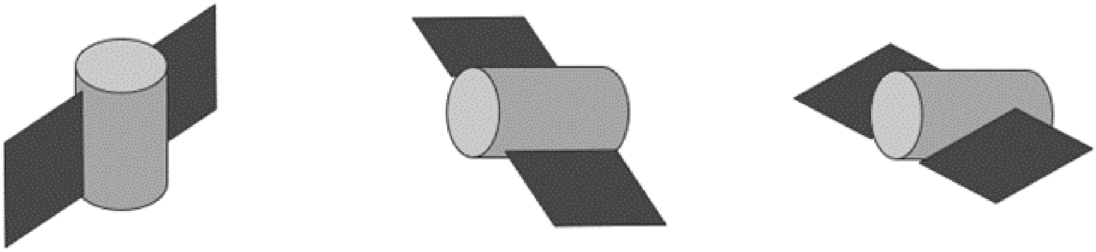


Figure 4.2: Illustration of the Servicer in different configurations: minimum ballistic coefficient or maximum relative drag (left), maximum ballistic coefficient or minimum relative drag (center), optimal ballistic coefficient or optimal relative drag (right)

Parameters	**Target**	**Servicer**
Earth's gravitational constant, μ [m^3/s^2]	$3.986004418 \times 10^{14}$	
Semi-major axis, a [m]	6928.14×10^3	6928.14×10^3
Orbital period, T [s]	5.7390×10^3	5.7390×10^3
Orbital angular frequency, n [rad/s]	0.0011	0.0011
Mean air density, ρ [kg/m^3]	10×10^{-12}	
Mass, m [kg]	200	2000
Drag coefficient, C_D	2.2	2.2
Drag area, A [m^2]	1	14
Side length, l [m]	5	-

Table 4.1: Initial orbital parameters and spacecraft characteristics

In the initial simulations, various initial states of the Servicer with respect to the Target in the LVLH frame were considered for testing purposes. After gaining a comprehensive understanding of the constraints, a specific state was selected for the inspection and docking phases.

The following lower and upper boundaries were considered for the control variables, expressed as inequalities:

$$1 \leq \boldsymbol{\Delta t} \leq t_f \,[\mathrm{s}] \tag{4.2}$$

$$-0.1 \leq \boldsymbol{\Delta V} \leq 0.1 \,[\mathrm{m/s}] \tag{4.3}$$

$$4 \leq \boldsymbol{A_c} \leq 24 \,[\mathrm{m^2}] \tag{4.4}$$

where $t_f = N \times T$ is the time of flight, i.e., the total mission time. The extreme values for ΔV and A_c were chosen considering typical values for this type of missions. For Δt, a minimum value of 1 s was considered between consecutive impulses to avoid overlap. Additionally, a time constraint was imposed on the sum of the Δt values, such that:

$$\sum_{i=1}^{N-1} \boldsymbol{\Delta t_i} \leq t_f \tag{4.5}$$

It's important to note that restricting the lower and upper boundaries associated with the control variables can significantly impact the performance of the optimizer in terms of:

1. **Convergence speed**: Tighter bounds can lead to faster convergence because the solution space is smaller, making it easier for the optimizer to find the optimal solution.
2. **Solution quality**: While it might speed up convergence, overly restrictive bounds can prevent the optimizer from finding the true global optimum, resulting in a sub-optimal solution.
3. **Feasibility**: Restrictive bounds might lead to infeasible solutions if the true optimal solution lies outside the specified bounds.

For this reason, it's important to start with wider bounds to allow the optimizer to explore the solution space, and then gradually tighten the bounds based on insights gained from initial runs, focusing the search in promising regions. Knowledge of the problem helps to set realistic bounds, ensuring they encapsulate all feasible solutions while excluding unlikely or physically impossible ones.

4.1.4 Implementation in MATLAB®

The development of the optimization tool was an iterative process. Initially, the code was created with a basic set of constraints to solve a simplified version of the problem. Over time, additional constraints were incrementally introduced to increase the complexity of the optimization problem. This gradual approach allowed for thorough testing and validation at each stage, ensuring that the code was robust and capable of handling the added constraints, ultimately achieving the desired performance for the Servicer's trajectory around the Target.

The constraints for the optimization problem are detailed as follows:

1. **Outer and inner control boxes**: Throughout the designated mission time, starting from the given initial conditions, the Servicer must always remain within the outer control box while staying outside the inner control box.

2. **Inspection points**: The Servicer must inspect specific points on the Target, ensuring that both the Field-of-View and the Line-of-Sight occlusion constraints are satisfied within the simulation time.

3. **Safety Ellipse**: During the inspection of the Target, the Servicer must stay within a cylindrical shell in order to follow the Safety Ellipse trajectory around the Target.

4. **Return to starting position**: Upon completing the inspection of the Target, the Servicer must return to its initial position.

5. **Docking**: In the final mission phase, the Servicer must dock with the Target while remaining within a cone-shaped approach corridor.

4.2 Inspection Phase

4.2.1 Control Boxes

The control box method was crucial for inspecting the Target, ensuring that the Servicer maintained its position and orientation relative to the Target. By defining specific control zones, namely the outer and inner control boxes, the Servicer's movements were restricted to safe regions, preventing it from straying too far or getting too close to the Target. This approach optimized the inspection phase and minimized collision risks. By leveraging control boxes, the Servicer could dynamically adjust its trajectory, staying within safe zones and ensuring a thorough and safe inspection of the Target. For more detailed information about this method, the reader can refer to the literature (see, e.g., [21]).

The following figures illustrate the trajectories of the Servicer around the Target, from an initial time $t_0 = 0$ s to a final time $t_f = N \times T$, evaluating different control box configurations. The constraints on the position of the Servicer are formulated in terms of inequality constraints as follows:

$$\|\mathbf{r}\| \leq \text{dimension of the outer control box} \tag{4.6}$$

$$\|\mathbf{r}\| \geq \text{dimension of the inner control box} \tag{4.7}$$

where $\boldsymbol{r} = (\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z})$ is the position vector of the Servicer in the LVLH frame, while the dimension of the control box corresponds to the characteristic length, which can be either the semi-side length or the radius, depending on the shape of the box.

Various geometries for the control boxes, including cubic, spherical, and cylindrical shapes, were tested to determine the most effective configuration for target inspection. The following plot figures are accompanied by two tables, which define the initial state of the Servicer with respect to the Target and the shapes and dimensions of the control boxes. For the cases of greatest practical interest, a third table provides the simulation results from the optimization tool.

In particular, the combination of a cubic outer control box and a spherical inner control box provided a balanced trade-off between safety and maneuverability, and is therefore adopted in the subsequent sections.

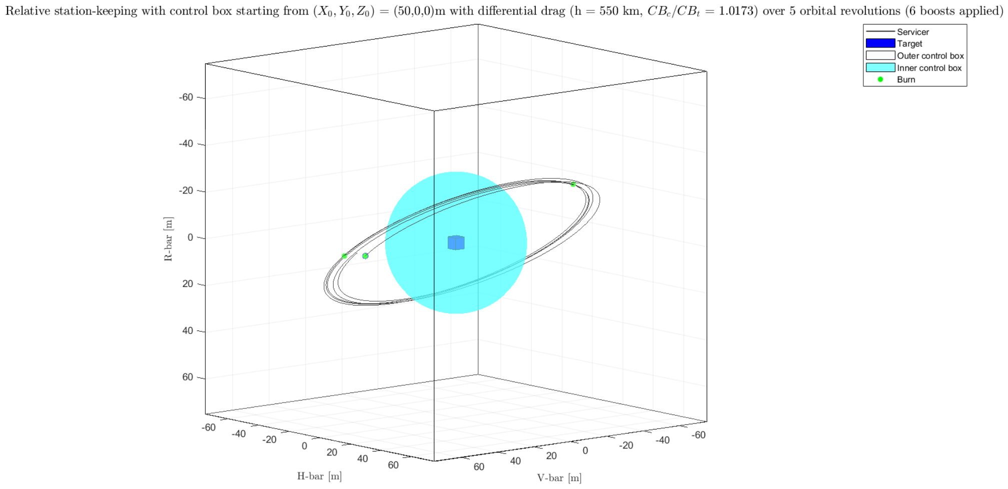


Figure 4.3: Optimized trajectory of the Servicer around the Target with constraints on the control boxes (Case 1.1)

Parameters	**Target**	**Servicer**
Initial position (x_0, y_0, z_0) [m]	(0, 0, 0)	(50, 0, 0)
Initial velocity ($\dot{x}_0$, $\dot{y}_0$, $\dot{z}_0$) [m/s]	(0, 0, 0)	(0, 0, 0)

Table 4.2: Initial state of the spacecraft (Case 1.1)

Control boxes	**Dimensions**
Cubic outer control box	Side length = 75 m
Spherical inner control box	Radius = 30 m

Table 4.3: Shape and dimensions of the control boxes (Case 1.1)

Parameters	**Simulation values**
Time of flight, t_f [s]	22955.986184
Total number of impulses applied, N	6
1$^{\text{st}}$ Optimal burn time [s]	0.000000
1$^{\text{st}}$ Optimal ΔV [m/s]	[-0.000619, 0.045069, -0.033727]
2$^{\text{nd}}$ Optimal burn time [s]	13412.911104
2$^{\text{nd}}$ Optimal ΔV [m/s]	[-0.000363, -0.000069, -0.000011]
3$^{\text{rd}}$ Optimal burn time [s]	22952.986173
3$^{\text{rd}}$ Optimal ΔV [m/s]	[0.000001, 0.000001, -0.000001]
4$^{\text{th}}$ Optimal time [s]	22953.986178
4$^{\text{th}}$ Optimal ΔV [m/s]	[0.000001, -0.000001, 0.000001]
5$^{\text{th}}$ Optimal burn time [s]	22954.986181
5$^{\text{th}}$ Optimal ΔV [m/s]	[-0.000000, 0.000000, 0.000000]
6$^{\text{th}}$ Optimal burn time [s]	22955.986184
6$^{\text{th}}$ Optimal ΔV [m/s]	[-0.000000, 0.000000, -0.000000]
Total ΔV [m/s]	0.056669
Optimal drag area, A_c [m^2]	9.830051
Optimal ballistic coefficient, CB_c [kg/m^2]	92.480789

Table 4.4: Optimization results (Case 1.1)

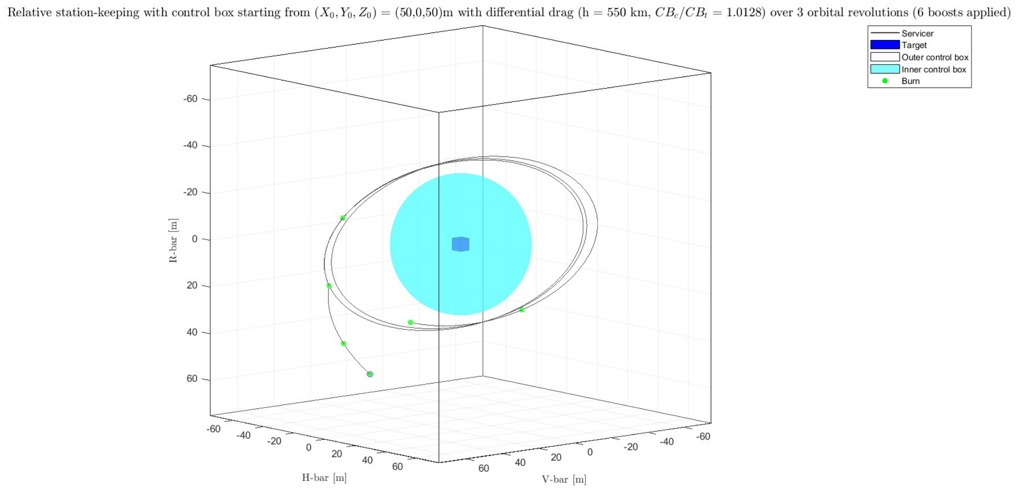


Figure 4.4: Optimized trajectory of the Servicer around the Target with constraints on the control boxes (Case 1.2)

Parameters	**Target**	**Servicer**
Initial position $(x_0,\, y_0,\, z_0)$ [m]	(0, 0, 0)	(50, 0, 50)
Initial velocity $(\dot{x}_0,\, \dot{y}_0,\, \dot{z}_0)$ [m/s]	(0, 0, 0)	(0, 0, 0)

Table 4.5: Initial state of the spacecraft (Case 1.2)

Control boxes	**Dimensions**
Cubic outer control box	Side length = 75 m
Spherical inner control box	Radius = 30 m

Table 4.6: Shape and dimensions of the control boxes (Case 1.2)

Parameters	**Simulation values**
Time of flight, t_f [s]	17216.989628
Total number of impulses applied, N	6
1st Optimal burn time [s]	0.000000
1st Optimal ΔV [m/s]	[0.078141, -0.000213, -0.064719]
2nd Optimal burn time [s]	234.034156
2nd Optimal ΔV [m/s]	[0.000000, -0.000000, -0.000000]
3rd Optimal burn time [s]	734.901643
3rd Optimal ΔV [m/s]	[0.031184, -0.000056, 0.001375]
4th Optimal time [s]	7206.695293
4th Optimal ΔV [m/s]	[-0.000000, 0.000000, -0.000000]
5th Optimal burn time [s]	16387.753909
5th Optimal ΔV [m/s]	[0.000000, -0.000000, 0.000000]
6th Optimal burn time [s]	17216.989628
6th Optimal ΔV [m/s]	[0.000000, -0.000000, -0.000000]
Total ΔV [m/s]	0.132678
Optimal drag area, A_c [m^2]	9.873360
Optimal ballistic coefficient, CB_c [kg/m^2]	92.075134

Table 4.7: Optimization results (Case 1.2)

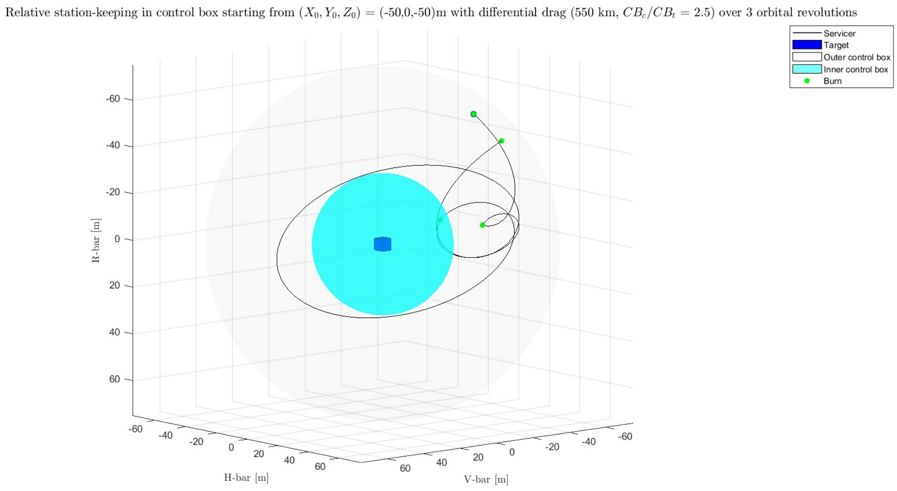


Figure 4.5: Optimized trajectory of the Servicer around the Target with constraints on the control boxes (Case 1.3)

Parameters	Target	Servicer
Initial position (x_0, y_0, z_0) [m]	(0, 0, 0)	(-50, 0, -50)
Initial velocity $(\dot{x}_0, \dot{y}_0, \dot{z}_0)$ [m/s]	(0, 0, 0)	(0, 0, 0)

Table 4.8: Initial state of the spacecraft (Case 1.3)

Control boxes	Dimensions
Spherical outer control box	Radius = 75 m
Spherical inner control box	Radius = 30 m

Table 4.9: Shape and dimensions of the control boxes (Case 1.3)

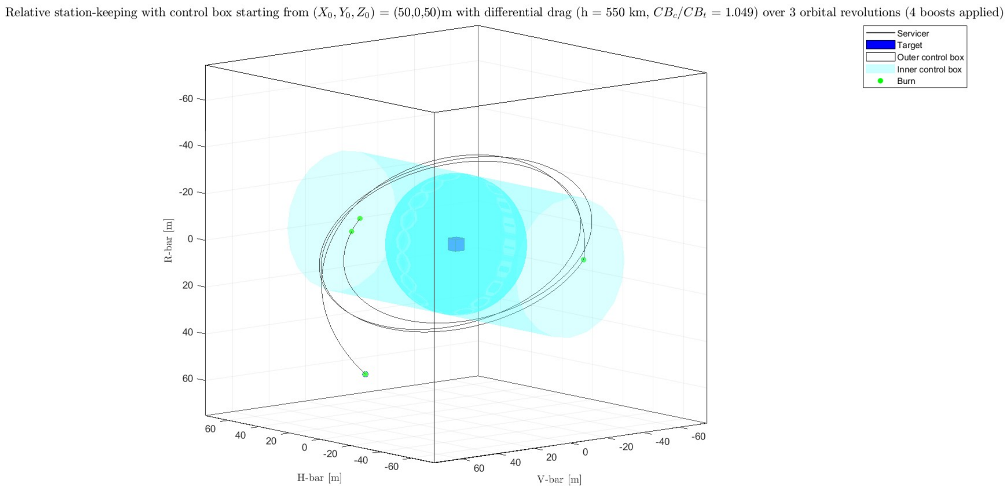


Figure 4.6: Optimized trajectory of the Servicer around the Target with constraints on the control boxes (Case 1.4)

Parameters	Target	Servicer
Initial position (x_0, y_0, z_0) [m]	(0, 0, 0)	(50, 0, 50)
Initial velocity ($\dot{x}_0$, $\dot{y}_0$, $\dot{z}_0$) [m/s]	(0, 0, 0)	(0, 0, 0)

Table 4.10: Initial state of the spacecraft (Case 1.4)

Control boxes	Dimensions
Cubic outer control box	Side length = 150 m
Spherical inner control box	Radius = 30 m

Cylindrical control boxes	Radius = 30 m
	Heigth = 150 m

Table 4.11: Shape and dimensions of the control boxes (case 1.4)

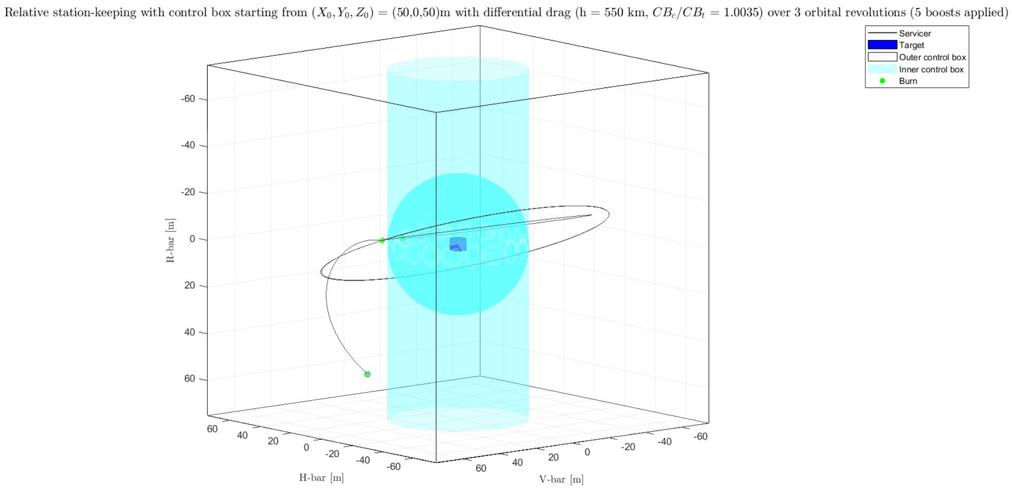


Figure 4.7: Optimized trajectory of the Servicer around the Target with constraints on the control boxes (Case 1.5)

Parameters	Target	Servicer
Initial position (x_0, y_0, z_0) [m]	(0, 0, 0)	(50, 0, 50)
Initial velocity $(\dot{x}_0, \dot{y}_0, \dot{z}_0)$ [m/s]	(0, 0, 0)	(0, 0, 0)

Table 4.12: Initial state of the spacecraft (Case 1.5)

Control boxes	Dimensions
Cubic outer control box	Side length = 150 m
Spherical inner control box	Radius = 30 m

Cylindrical control boxes	Radius = 30 m
	Heigth = 150 m

Table 4.13: Shape and dimensions of the control boxes (Case 1.5)

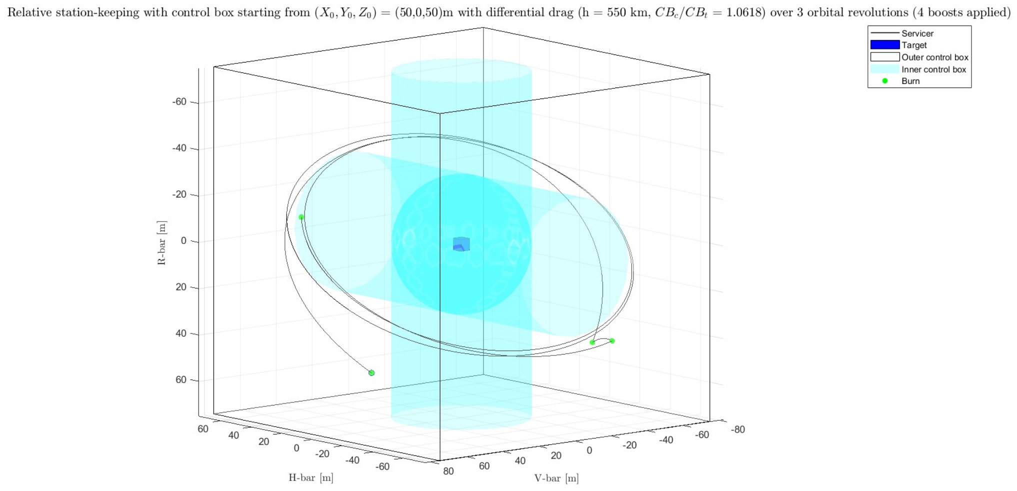


Figure 4.8: Trajectory of the Servicer around the Target with constraints on the control boxes (Case 1.6)

Parameters	**Target**	**Servicer**
Initial position $(x_0,\, y_0,\, z_0)$ [m]	(0, 0, 0)	(50, 0, 50)
Initial velocity $(\dot{x}_0,\, \dot{y}_0,\, \dot{z}_0)$ [m/s]	(0, 0, 0)	(0, 0, 0)

Table 4.14: Initial state of the spacecraft (Case 1.6)

Control boxes	**Dimensions**
Cubic outer control box	Side length = 150 m
Spherical inner control box	Radius = 30 m

Cylindrical control boxes	Radius = 30 m
	Heigth = 150 m

Table 4.15: Shape and dimensions of the control boxes (Case 1.6)

Parameters	**Simulation values**
Time of flight, t_f [s]	17216.989629
Total number of impulses applied, N	4
1st Optimal burn time [s]	0.000000
1st Optimal ΔV [m/s]	[0.091209, 0.049595, -0.077447]
2nd Optimal burn time [s]	3260.506026
2nd Optimal ΔV [m/s]	[-0.043757, -0.033931, -0.108022]
3rd Optimal burn time [s]	3637.770013
3rd Optimal ΔV [m/s]	[0.059851, 0.052575, 0.014536]
4th Optimal burn time [s]	17216.989629
4th Optimal ΔV [m/s]	[-0.043022, -0.021746, -0.013011]
Total ΔV [m/s]	0.381821
Optimal drag area, A_c [m^2] [m^2]	9.417880
Optimal ballistic coefficient, CB_c [kg/m^2]	96.528193

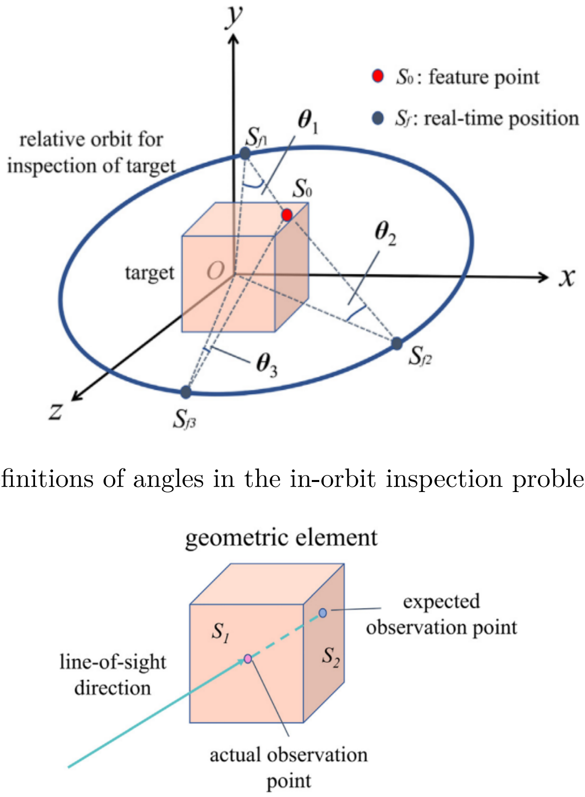
Table 4.16: Optimization results (Case 1.6)

4.2.2 Field-Of-View and Occlusion

The inspection of the Target involves closely monitoring specific points to ensure, for example, its structural integrity and operational safety. Two critical factors in this context are the *Field-of-View* (FOV) and the *Line-of-Sight* (LOS) *occlusion*.

The FOV constraint ensures that the inspection camera mounted on the Servicer can capture the necessary images of the Target without any distortion and within the optimal range (Figure 4.9a).

Occlusion occurs when the inspection point is obstructed by another part of the Target, blocking the LOS of the Servicer's camera (Figure 4.9b). Evaluating the occlusion constraints is essential for ensuring comprehensive inspection coverage. For a more detailed and comprehensive understanding of these concepts and their mathematical formulations, the reader can refer to related literature (see, e.g., [22]).



(a) Definitions of angles in the in-orbit inspection problem [22]

(b) Occlusion effect of a geometric element [22]

Figure 4.9: Representation of the FOV and occlusion constraints

As depicted in Figure 4.10, the inspection points are selected at the centers of the faces and the vertices of the cubic Target, resulting in a total of 14 points to be inspected. This approach balances the complexity of constraints while ensuring near-global coverage of the Target.

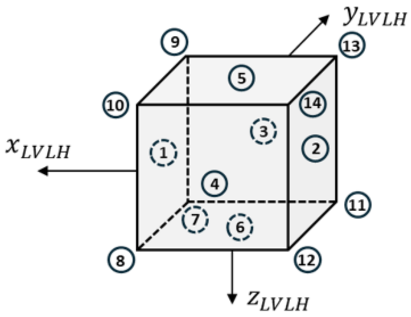


Figure 4.10: Inspection points of the target surface

The mathematical formulation of the FOV constraints is as follows [22]:

$$d_\mathrm{r} \le d_\mathrm{observe} \tag{4.8}$$

$$\tag{4.9}$$

$$\frac{(\mathbf{S_f} - \mathbf{S_0}) \cdot \mathbf{S_f}}{d_\mathrm{r} \cdot d_\mathrm{f}} \ge \cos\left(\frac{\theta_\mathrm{max}}{2}\right) \tag{4.10}$$

where:

- $\mathbf{S_0}$ is the position vector of the inspection points.
- $\mathbf{S_f}$ is the position vector of the Servicer.
- $\mathbf{d_0} = \|\mathbf{S_0}\|$ and $\mathbf{d_f} = \|\mathbf{S_f}\|$ are the magnitudes of the position vectors.
- $\mathbf{d_r} = \|\mathbf{S_f} - \mathbf{S_0}\|$ is the magnitude of the vector connecting the Servicer to each inspection point.
- $\mathbf{d}_\mathrm{observe} = 45\,\mathrm{m}$ is the maximum observation distance, i.e., the maximum distance at which the Servicer's camera can observe the target point without significant image distortion.
- $\theta_\mathrm{max} = 30\,^\circ$ is the FOV angle, i.e., the maximum angular width of the camera's observation cone, ensuring that the Target remains within the camera's view.

These constraints ensure that the distance d_r remains within the observation limit and that the angle between the Servicer and the inspection point is within the camera's FOV.

To account for occlusion, the following inequality constraint is considered:

$$\frac{(\mathbf{S_f} - \mathbf{S_0}) \cdot \mathbf{n}}{d_\mathrm{r} \cdot d_\mathrm{n}} \geq \cos(\theta_\mathrm{occlusion}) \tag{4.11}$$

where:

- $\theta_\mathrm{occlusion}$ is the occlusion angle, ensuring that any intervening structures do not block the LOS. Specifically, $\theta_\text{occlusion, face center} = 60\,°$ for the face centers and $\theta_\text{occlusion, vertex} = 105\,°$ for the vertices. These values were chosen as a trade-off between literature data and analytical evaluations of the problem under investigation.
- $\mathbf{n}$ is the normal vector to the inspection points.
- $\mathbf{d_n} = \|\mathbf{n}\|$ is the magnitude of the normal vector.

This equation ensures that the LOS is free from obstructions by maintaining the angle between the normal to the inspection point and the vector connecting the Servicer to the inspection points within an acceptable threshold.

Like in subsection 4.2.1, various geometries for the control boxes were considered in this optimization step. This approach aimed to validate the inspection constraints implemented across different scenarios. After defining the initial state of the Servicer with respect to the Target, all the plots are presented along with two tables, which specify respectively the shape and dimensions of the control boxes and the simulation results provided by the optimization tool.

Parameters	**Target**	**Servicer**
Initial position (x_0, y_0, z_0) [m]	(0, 0, 0)	(50, 0, 0)
Initial velocity ($\dot{x}_0$, $\dot{y}_0$, $\dot{z}_0$) [m/s]	(0, 0, 0)	(0, 0, 0)

Table 4.17: Initial state of the spacecraft (Cases 2.1, 2.2, 2.3, 2.4)

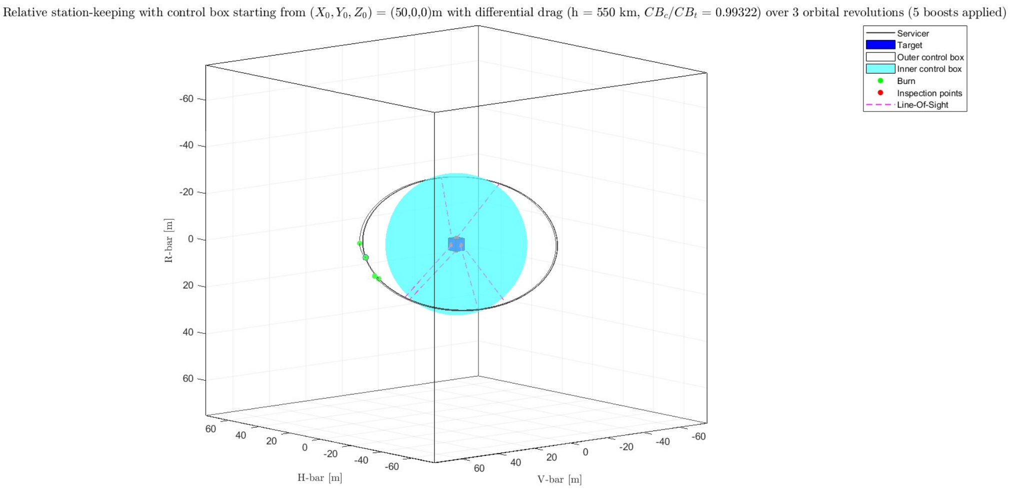


Figure 4.11: Optimized trajectory of the Servicer around the Target with constraints on the control boxes, FOV, and LOS occlusion (Case 2.1)

Control boxes	Dimensions
Cubic outer control box	Side length = 150 m
Spherical inner control box	Radius = 30 m

Table 4.18: Shape and dimensions of the control boxes (Case 2.1)

Parameters	**Simulation values**
Time of flight, t_f [s]	1232.364396
Total number of impulses applied, N	5
1st Optimal burn time [s]	0.000000
1st Optimal ΔV [m/s]	[0.006790, 0.007660, -0.009646]
2nd Optimal burn time [s]	449.540699
2nd Optimal ΔV [m/s]	[-0.006612, 0.009270, -0.012775]
3rd Optimal burn time [s]	11424.109337
3rd Optimal ΔV [m/s]	[0.000102, 0.000044, -0.000099]
4th Optimal burn time [s]	17215.817647
4th Optimal ΔV [m/s]	[-0.000030, -0.000010, -0.000107]
5th Optimal burn time [s]	17216.989628
5th Optimal ΔV [m/s]	[-0.000031, 0.000033, -0.000052]
Total ΔV [m/s]	0.031508
Optimal drag area, A_c [m^2]	10.068240
Optimal ballistic coefficient, CB_c [kg/m^2]	90.292931

Table 4.19: Optimization results (Case 2.1)

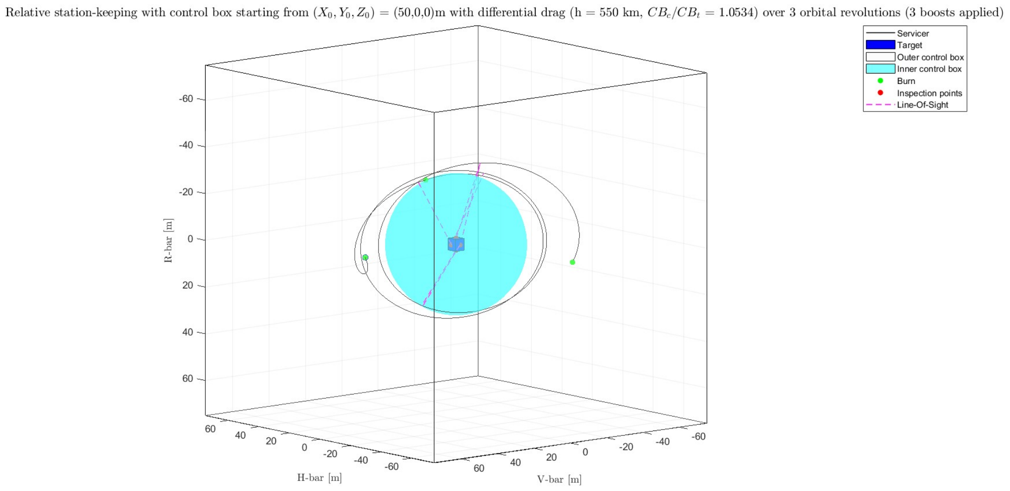


Figure 4.12: Optimized trajectory of the Servicer around the Target with constraints on the control boxes, FOV, and LOS occlusion (Case 2.2)

Control boxes	**Dimensions**
Cubic outer control box	Side length = 150 m
Spherical inner control box	Radius = 30 m

Table 4.20: Shape and dimensions of the control boxes (Case 2.2)

Parameters	**Simulation values**
Time of flight, t_f [s]	1232.364396
Total number of impulses applied, N	3
1$^{\text{st}}$ Optimal burn time [s]	0.000000
1$^{\text{st}}$ Optimal ΔV [m/s]	[0.005874, -0.016557, 0.011650]
2$^{\text{nd}}$ Optimal burn time [s]	3540.325359
2$^{\text{nd}}$ Optimal ΔV [m/s]	[-0.007074, 0.000524, -0.003945]
3$^{\text{rd}}$ Optimal burn time [s]	17216.989628
3$^{\text{rd}}$ Optimal ΔV [m/s]	[-0.000406, -0.000056, 0.000052]
Total ΔV [m/s]	0.029609
Optimal drag area, A_c [m^2]	9.492686
Optimal ballistic coefficient, CB_c [kg/m^2]	95.767507

Table 4.21: Optimization results (Case 2.2)

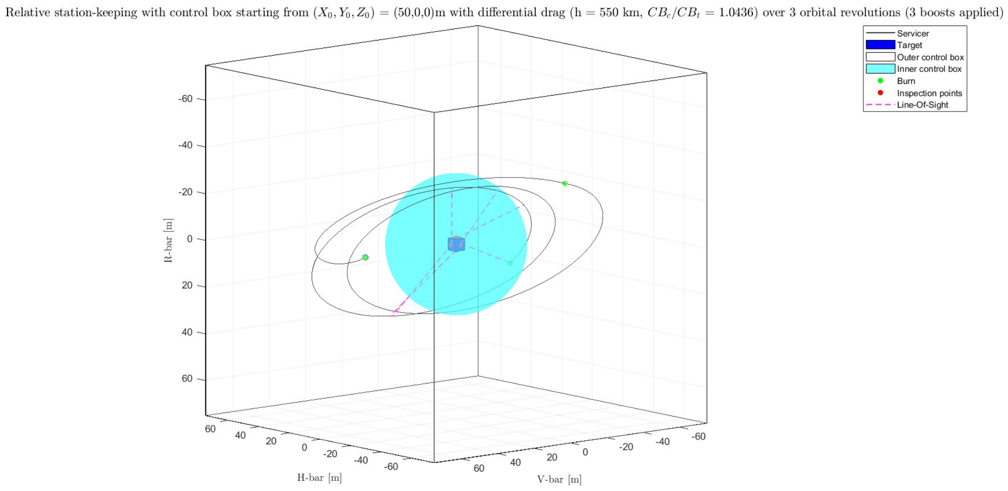


Figure 4.13: Optimized trajectory of the Servicer around the Target with constraints on the control boxes, FOV, and LOS occlusion (Case 2.3)

Control boxes	**Dimensions**
Cubic outer control box	Side length = 150 m
Spherical inner control box	Radius = 30 m

Table 4.22: Shape and dimensions of the control boxes (Case 2.3)

Parameters	**Simulation values**
Time of flight, t_f [s]	17216.989628
Total number of impulses applied, N	3
1st Optimal burn time [s]	0.000000
1st Optimal ΔV [m/s]	[0.006732, 0.016984, 0.010442]
2nd Optimal burn time [s]	4330.692078
2nd Optimal ΔV [m/s]	[-0.009316, 0.002218, -0.000583]
3rd Optimal burn time [s]	17216.989628
3rd Optimal ΔV [m/s]	[-0.000569, 0.000249, 0.000131]
Total ΔV [m/s]	0.031272
Optimal drag area, A_c [m^2]	9.581808
Optimal ballistic coefficient, CB_c [kg/m^2]	94.876758

Table 4.23: Optimization results (Case 2.3)

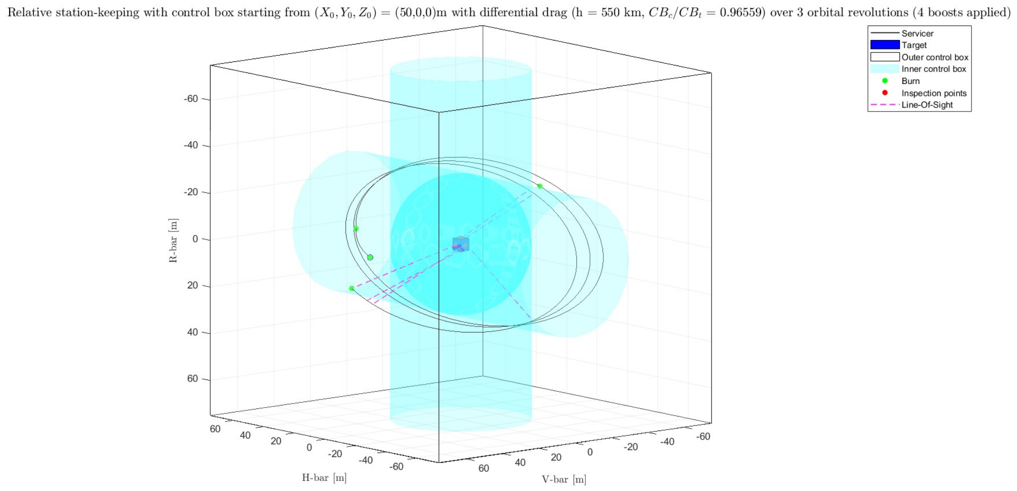


Figure 4.14: Trajectory of the Servicer around the Target with constraints on the control boxes, FOV, and LOS occlusion (Case 2.4)

Control boxes	**Dimensions**
Cubic outer control box	Side length = 150 m
Spherical inner control box	Radius = 30 m

Cylindrical control boxes	Radius = 30 m
	Heigth = 150 m

Table 4.24: Shape and dimensions of the control boxes (Case 2.4)

Parameters	**Simulation values**
Time of flight, t_f [s]	17216.989628
Total number of impulses applied, N	4
1st Optimal burn time [s]	0.000000
1st Optimal ΔV [m/s]	[0.008748, 0.019623, -0.013623]
2nd Optimal burn time [s]	592.758993
2nd Optimal ΔV [m/s]	[-0.008420, 0.015764, -0.010449]
3rd Optimal burn time [s]	8353.326343
3rd Optimal ΔV [m/s]	[0.000606, 0.000031, -0.000331]
4th Optimal burn time [s]	17216.989628
4th Optimal ΔV [m/s]	[-0.000033, -0.000002, -0.000017]
Total ΔV [m/s]	0.046871
Optimal drag area, A_c [m^2]	10.356388
Optimal ballistic coefficient, CB_c [kg/m^2]	87.780688

Table 4.25: Optimization results (Case 2.4)

4.2.3 Safety Ellipse

In IOS operations, ensuring the safety of both the Servicer and the Target is critical. An effective strategy to achieve this is through the use of a *Safety Ellipse* (SE), which involves a combination of out-of-plane and in-plane elliptical motions. By injecting the Servicer into a safety ellipse trajectory, we ensure that the Servicer never crosses the V-bar axis, i.e., the direction of motion of the Target. In this way, even in the event of a drift toward the Target, a collision will not occur, as the Servicer will move around the Target at a distance defined by the extensions of the ellipse along the in-plane and out-of-plane directions [16]. More detailed information about this approach can be found in the literature (see, e.g., [23], [24]).

The initial approach was to implement a so-called Walking Safety Ellipse, centered at a point along the in-track direction and moving along this direction with a certain drift velocity. Using the SE equations of motion found in the literature [24], the objective was to make the Servicer acquire and then follow the SE trajectory. The maximum out-of-plane and radial extensions of the ellipse $y_{\max}$ and $z_{\max}$, along with the drift velocity in the in-track direction $\dot{x}_c$, were adjusted to satisfy the constraints on the control boxes, FOV and LOS occlusion. An example of this trajectory is given in Figure 4.15. It should be noted that the SE provides only "short-term safety", as over a larger number of orbits, the uncertainty of the differential drag must be taken into account [16].

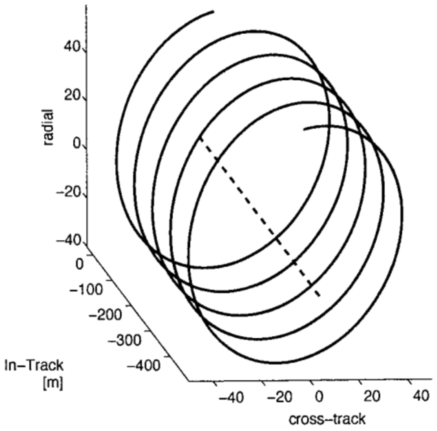


Figure 4.15: Relative motion in the Walking Safety Ellipse [25]

The problem with this approach was that it necessitated the introduction of a series of equality constraints to ensure that the Servicer's trajectory coincided with the SE trajectory, making the optimization problem excessively constrained and too computationally intensive for the optimizer to find a feasible solution, as shown in Figure 4.16.

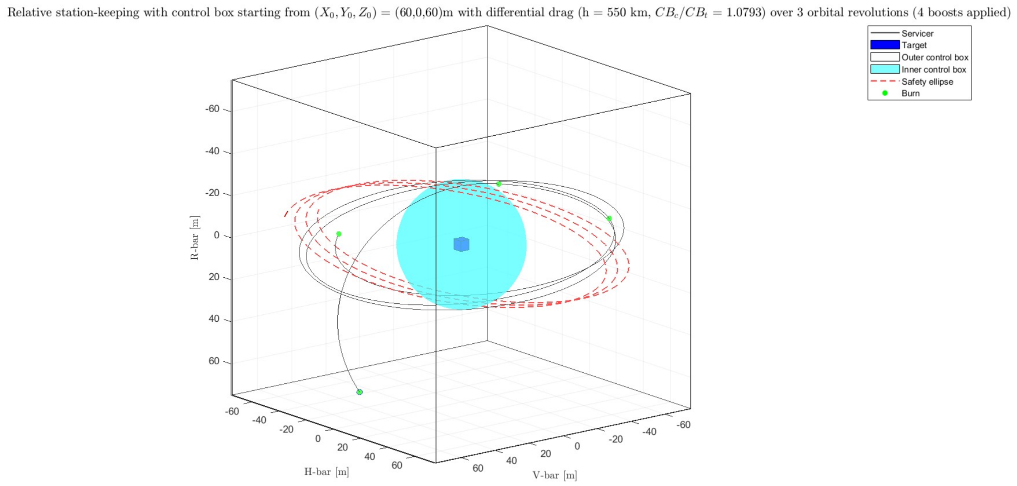


Figure 4.16: Safety Ellipse acquisition attempt

Due to this, it was necessary to redefine the strategy. It was decided to implement the safety ellipse by applying its theoretical definition, which involves applying an out-of-plane impulse (along $\pm H$-bar) at the starting position along the V-bar and an in-plane impulse (along $\pm R$-bar) a quarter orbit later, or vice versa, as shown in Figure 4.17. The lower and upper bounds defined in Equations 4.2 and 4.3 were modified as follows:

$$
\begin{aligned}
0 &\leq \Delta t_1 \leq T/4\,[\text{s}] && (4.12)\\
0 &\leq \Delta V_{1,x} \leq 0\,[\text{m/s}] && (4.13)\\
-0.1 &\leq \Delta V_{1,y} \leq 0.1\,[\text{m/s}] && (4.14)\\
0 &\leq \Delta V_{1,z} \leq 0\,[\text{m/s}] && (4.15)
\end{aligned}
$$

The second impulse is not constrained along the R-bar direction alone, as it may contain small components along the other two directions to compensate for any deviations that might have occurred due to the initial impulse. This flexibility ensures the trajectory remains within the safety ellipse, maintaining the required relative motion constraints.

To ensure the Servicer's motion along a spiral trajectory typical of the safety ellipse, two additional cylindrical control boxes, one inner and one outer, were introduced. These boxes create a cylindrical shell within which the Servicer is constrained to move during the inspection of the Target, effectively simulating a SE. The constraints were formulated using the same mathematical expressions defined in Equations 4.6 and 4.7.

Finally, to avoid potential collisions due to accidental drift of the Servicer along the direction of motion (V-bar) during passes in front of or behind the Target, a constraint was imposed such that the coordinates along H-bar and R-bar cannot be zero simultaneously during the inspection phase.

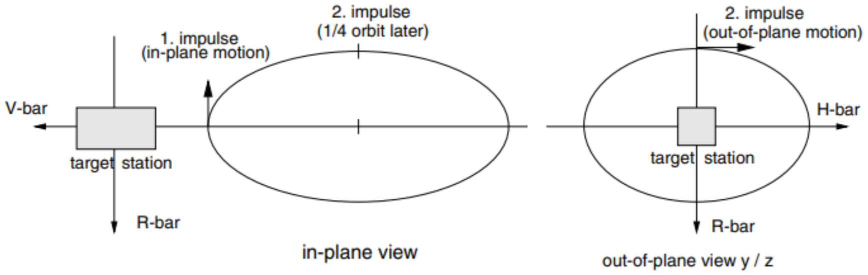


Figure 4.17: Safety Ellipse implementation [16]

The following figures illustrate the optimized trajectories achieved with this approach, considering all the constraints discussed so far: control boxes, FOV, LOS occlusion, and SE. This method ensures safe relative motion, minimizes collision risks, and relaxes constraints, while simultaneously providing favorable conditions for the visual inspection of the Target.

After defining the initial state of the Servicer with respect to the Target, as well as the dimensions of the control boxes and the cylindrical shell, all the plots are presented along with the table collecting the simulation results provided by the optimization tool.

Parameters	**Target**	**Servicer**
Initial position (x_0, y_0, z_0) [m]	(0, 0, 0)	(50, 0, 0)
Initial velocity ($\dot{x}_0$, $\dot{y}_0$, $\dot{z}_0$) [m/s]	(0, 0, 0)	(0, 0, 0)

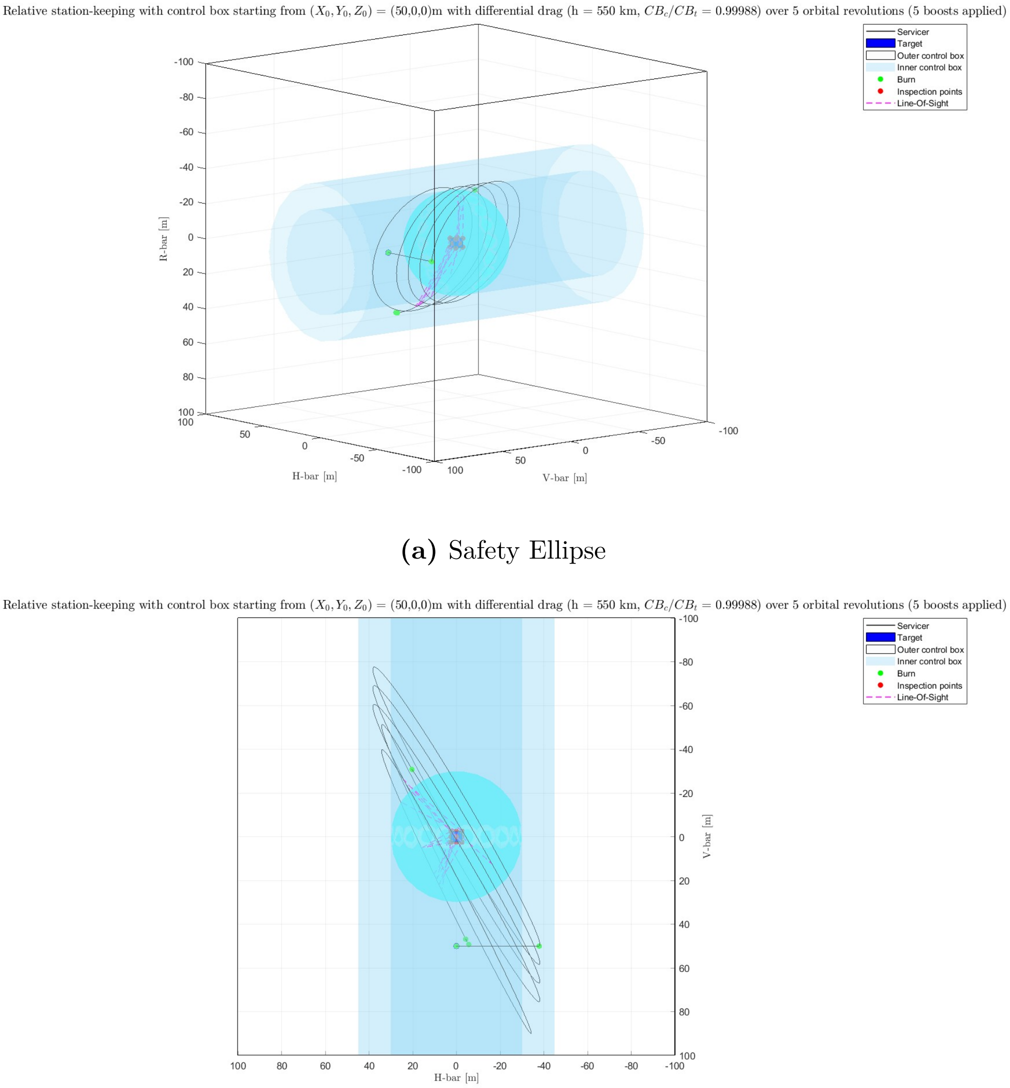
Table 4.26: Initial state of the spacecraft (Cases 3.1, 3.2, 3.3)

Control boxes	**Dimensions**
Cubic outer control box	Side length = 150 m
Spherical inner control box	Radius = 30 m
Cylindrical shell	Inner radius = 30 m Outer radius = 60 m Height = 150 m

Table 4.27: Shape and dimensions of the control boxes (Cases 3.1, 3.2, 3.3)

Parameters	**Simulation values**
Time of flight, t_f [s]	28694.982712
Total number of impulses applied, N	5
1st Optimal burn time [s]	0.000000
1st Optimal ΔV [m/s]	[0.000000, -0.042494, 0.000000]
2nd Optimal burn time [s]	1235.370621
2nd Optimal ΔV [m/s]	[-0.000490, 0.003597, -0.036057]
3rd Optimal burn time [s]	20519.709640
3rd Optimal ΔV [m/s]	[-0.000167, -0.005175, 0.001244]
4th Optimal burn time [s]	28662.318302
4th Optimal ΔV [m/s]	[-0.000031, -0.006093, -0.011671]
5th Optimal burn time [s]	28694.982712
5th Optimal ΔV [m/s]	[0.000346, 0.012485, 0.002100]
Total ΔV [m/s]	0.109890
Optimal drag area, A_c [m^2]	10.001191
Optimal ballistic coefficient, CB_c [kg/m^2]	90.898262

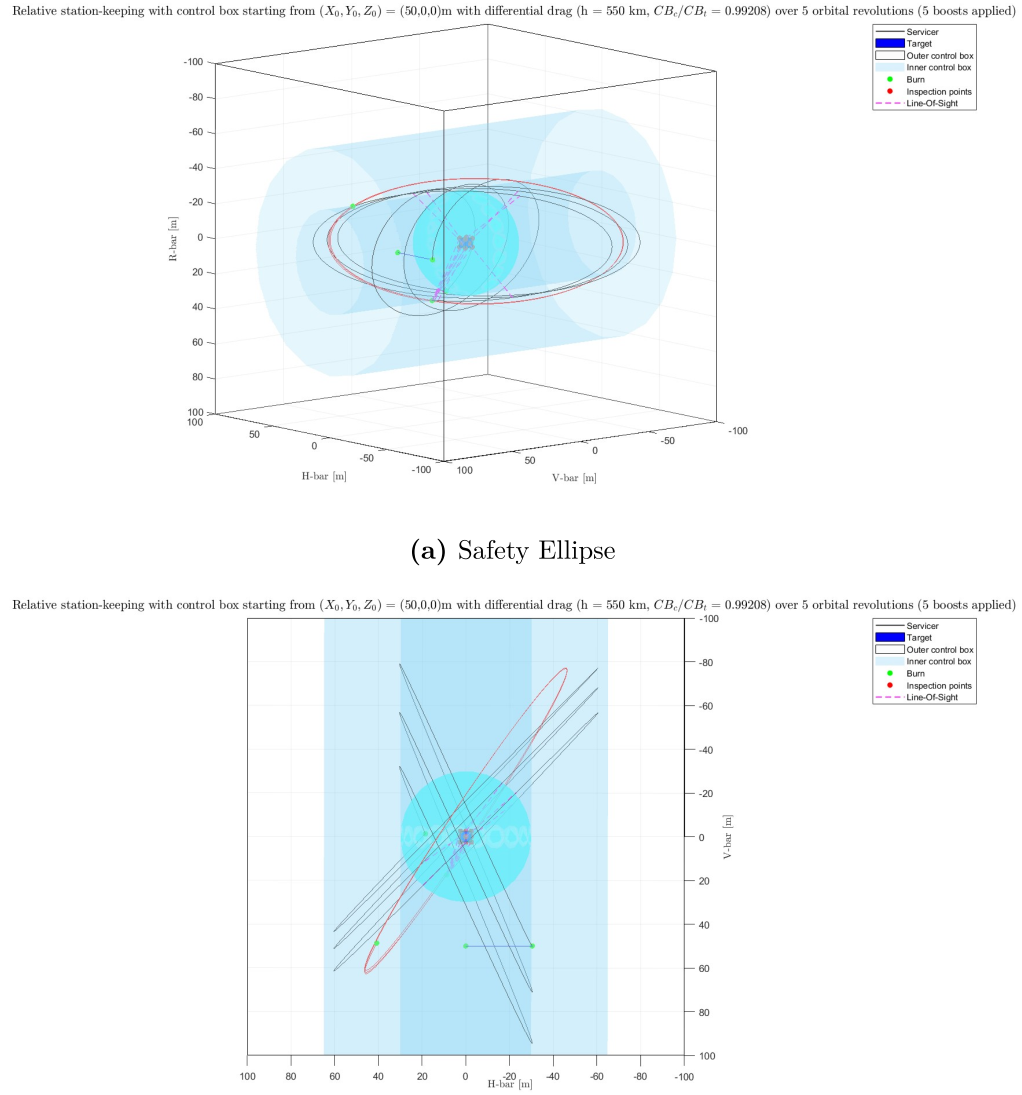
Table 4.28: Optimization results (Case 3.1)



(a) Safety Ellipse

(b) Out-of-plane view (x,y) of the motion of the SE

Figure 4.18: Optimized trajectory of the Servicer around the Target with constraints on the control boxes, FOV, LOS occlusion and SE (Case 3.1)



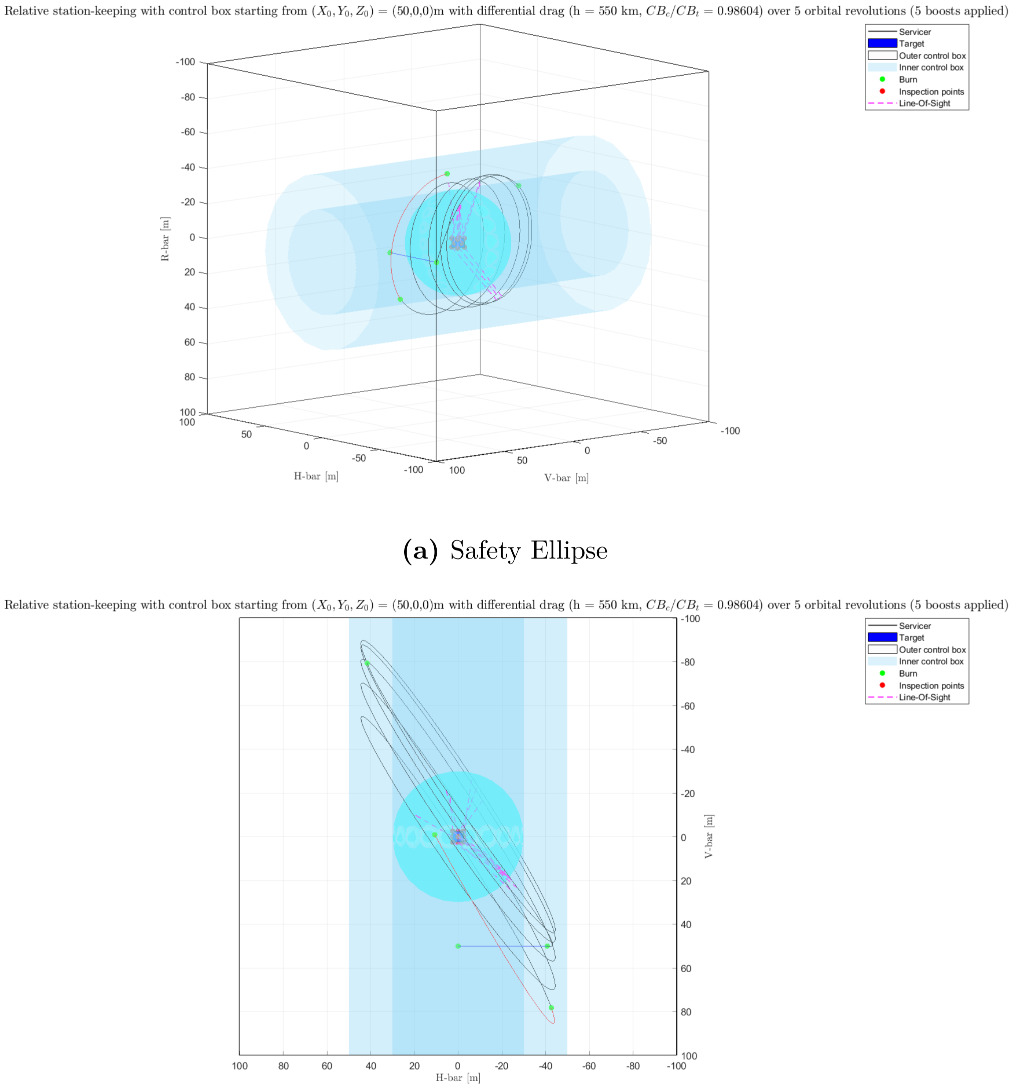
(a) Safety Ellipse

(b) Out-of-plane view (x,y) of the motion of the SE

Figure 4.19: Optimized trajectory of the Servicer around the Target with constraints on the control boxes, FOV, LOS occlusion and SE (Case 3.2)

Parameters	**Simulation values**
Time of flight, t_f [s]	46893.868751
Total number of impulses applied, N	5
1st Optimal burn time [s]	0.000000
1st Optimal ΔV [m/s]	[0.000000, -0.001112, 0.001184]
2nd Optimal burn time [s]	1524.517456
2nd Optimal ΔV [m/s]	[0.000930, 0.040544, -0.033446]
3rd Optimal burn time [s]	16647.129198
3rd Optimal ΔV [m/s]	[-0.001147, 0.089348, -0.015673]
4th Optimal burn time [s]	33709.663672
4th Optimal ΔV [m/s]	[0.040544, 0.000000, -0.038036]
5th Optimal burn time [s]	46893.868751
5th Optimal ΔV [m/s]	[-0.033539, -0.008619, -0.040544]
Total ΔV [m/s]	0.253824
Optimal drag area, A_c [m^2]	10.079848
Optimal ballistic coefficient, CB_c [kg/m^2]	90.188947

Table 4.29: Optimization results (Case 3.2)



(a) Safety Ellipse

(b) Out-of-plane view (x,y) of the motion of the SE

Figure 4.20: Optimized trajectory of the Servicer around the Target with constraints on the control boxes, FOV, LOS occlusion and SE (Case 3.3)

Parameters	**Simulation values**
Time of flight, t_f [s]	32179.213612
Total number of impulses applied, N	5
1^{st} Optimal burn time [s]	0.000000
1^{st} Optimal ΔV [m/s]	[0.000000, 0.005376, -0.005148]
2^{nd} Optimal burn time [s]	1468.477501
2^{nd} Optimal ΔV [m/s]	[-0.000538, 0.026069, -0.044660]
3^{rd} Optimal burn time [s]	4394.533927
3^{rd} Optimal ΔV [m/s]	[-0.017176, 0.004456, 0.001522]
4^{th} Optimal burn time [s]	30288.892684
4^{th} Optimal ΔV [m/s]	[0.008424, 0.000000, -0.022490]
5^{th} Optimal burn time [s]	32179.213612
5^{th} Optimal ΔV [m/s]	[0.007647, -0.010232, 0.018555]
Total ΔV [m/s]	0.123510
Optimal drag area, A_c [m^2]	10.141530
Optimal ballistic coefficient, CB_c [kg/m^2]	89.640413

Table 4.30: Optimization results (Case 3.3)

4.2.4 Return to Starting Position

After completing the Target inspection, as an additional constraint, the Servicer was required to return to its initial position, with zero final relative velocity:

$$(x, y, z)_{t_f} = (50, 0, 0)\,\mathrm{m} \tag{4.16}$$

$$(V_x, V_y, V_z)_{t_f} = (0, 0, 0)\,\mathrm{m/s} \tag{4.17}$$

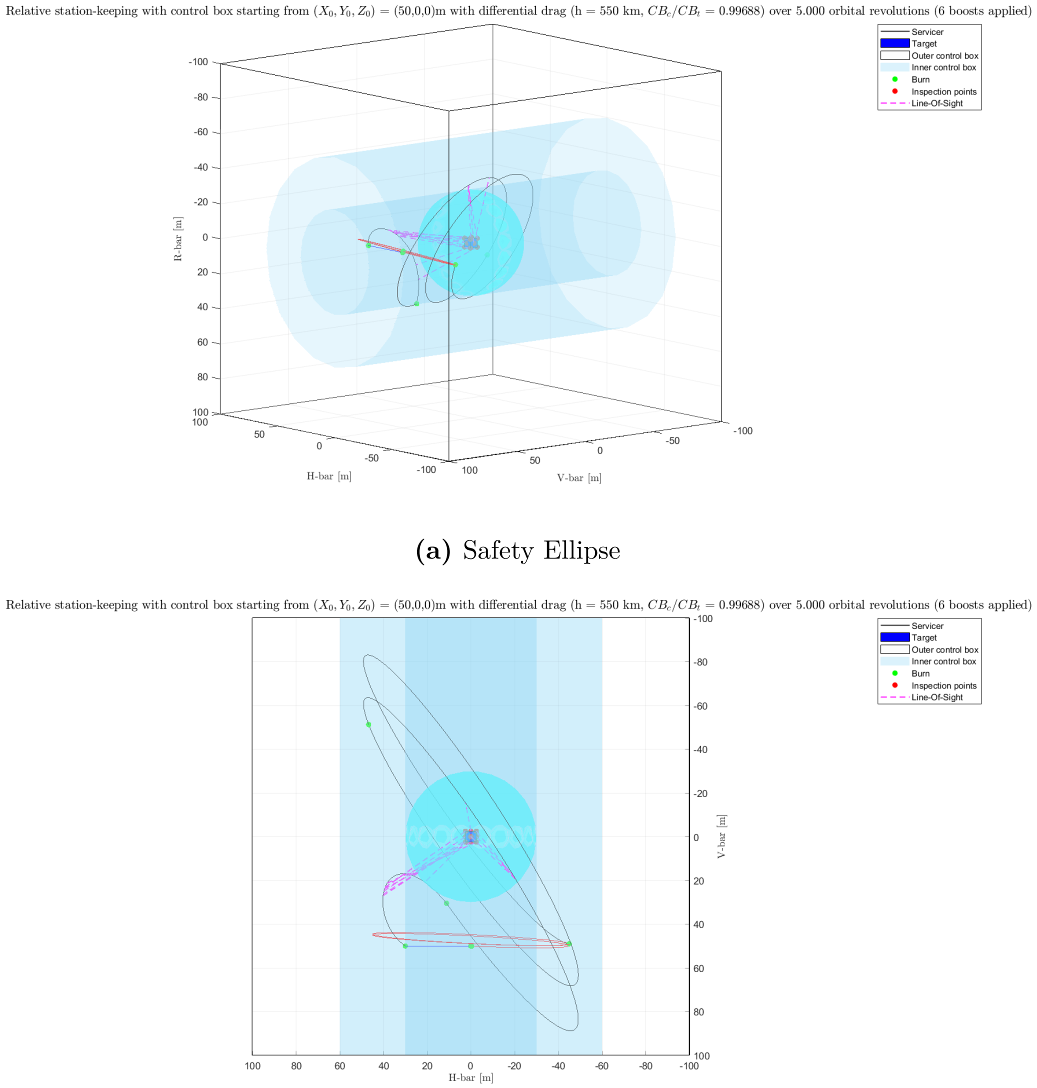
In the initial simulations, the Servicer returned to its starting position by executing "abrupt" maneuvers. To address this inconvenience, it was decided to impose an additional constraint: by the penultimate impulse, the Servicer must be aligned with the starting position in the xy-plane, ensuring minimal misalignment along the V-bar and R-bar axes. This should help smooth out the transition and avoid any abrupt changes in the trajectory, resulting in a more controlled and predictable return to the starting position. After defining the initial state of the Servicer with respect to the Target, as well as the dimensions of the control boxes and the cylindrical shell, all the plots are presented along with the table collecting the simulation results provided by the optimization tool.

Parameters	**Target**	**Servicer**
Initial position (x_0, y_0, z_0) [m]	(0, 0, 0)	(50, 0, 0)
Initial velocity ($\dot{x}_0$, $\dot{y}_0$, $\dot{z}_0$) [m/s]	(0, 0, 0)	(0, 0, 0)

Table 4.31: Initial state of the spacecraft (Cases 4.1, 4.2, 4.3)

Control boxes	**Dimensions**
Cubic outer control box	Side length = 150 m
Spherical inner control box	Radius = 30 m
Cylindrical shell	Inner radius = 30 m Outer radius = 80 m Heigth = 150 m

Table 4.32: Shape and dimensions of the control boxes (Cases 4.1, 4.2, 4.3)



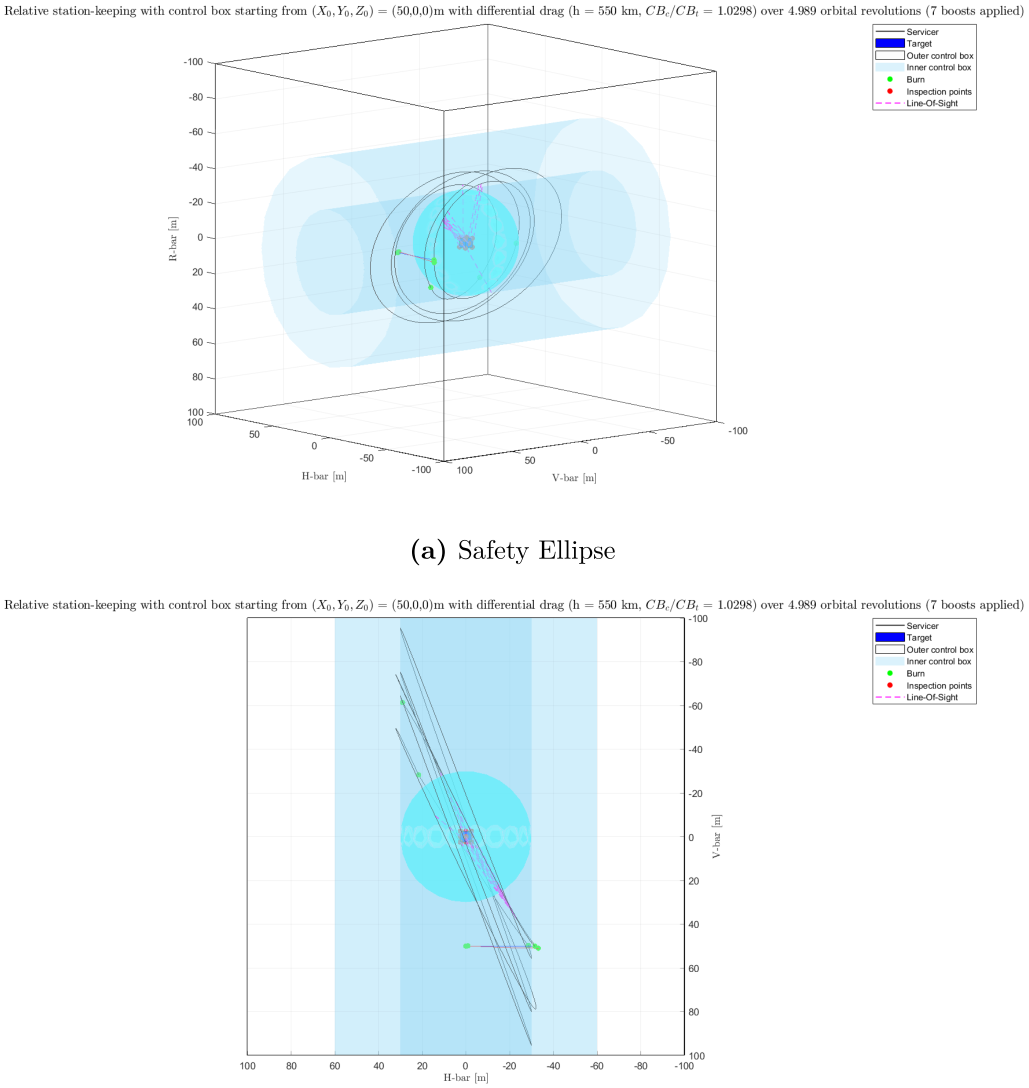
(a) Safety Ellipse

(b) Out-of-plane view (x,y) of the motion of the SE

Figure 4.21: Optimized trajectory of the Servicer around the Target with constraints on the control boxes, FOV, LOS occlusion, SE, and return (Case 4.1)

Parameters	**Simulation values**
Time of flight, t_f [s]	28694.982713
Total number of impulses applied, N	6
1$^{\text{st}}$ Optimal burn time [s]	0.000000
1$^{\text{st}}$ Optimal ΔV [m/s]	[0.000000, 0.034535, 0.000000]
2$^{\text{nd}}$ Optimal burn time [s]	1147.799309
2$^{\text{nd}}$ Optimal ΔV [m/s]	[-0.020353, 0.018810, -0.024940]
3$^{\text{rd}}$ Optimal burn time [s]	2993.910902
3$^{\text{rd}}$ Optimal ΔV [m/s]	[0.021633, -0.009968, -0.051681]
4$^{\text{th}}$ Optimal burn time [s]	7791.593330
4$^{\text{th}}$ Optimal ΔV [m/s]	[-0.000043, -0.000135, 0.000015]
5$^{\text{th}}$ Optimal burn time [s]	15732.610724
5$^{\text{th}}$ Optimal ΔV [m/s]	[-0.001045, 0.018619, 0.037448]
6$^{\text{th}}$ Optimal burn time [s]	28694.982713
6$^{\text{th}}$ Optimal ΔV [m/s]	[-0.000000, 0.000000, 0.000000]
Total ΔV [m/s]	0.170700
Optimal drag area, A_c [m^2]	10.031292
Optimal ballistic coefficient, CB_c [kg/m^2]	90.625505

Table 4.33: Optimization results (Case 4.1)



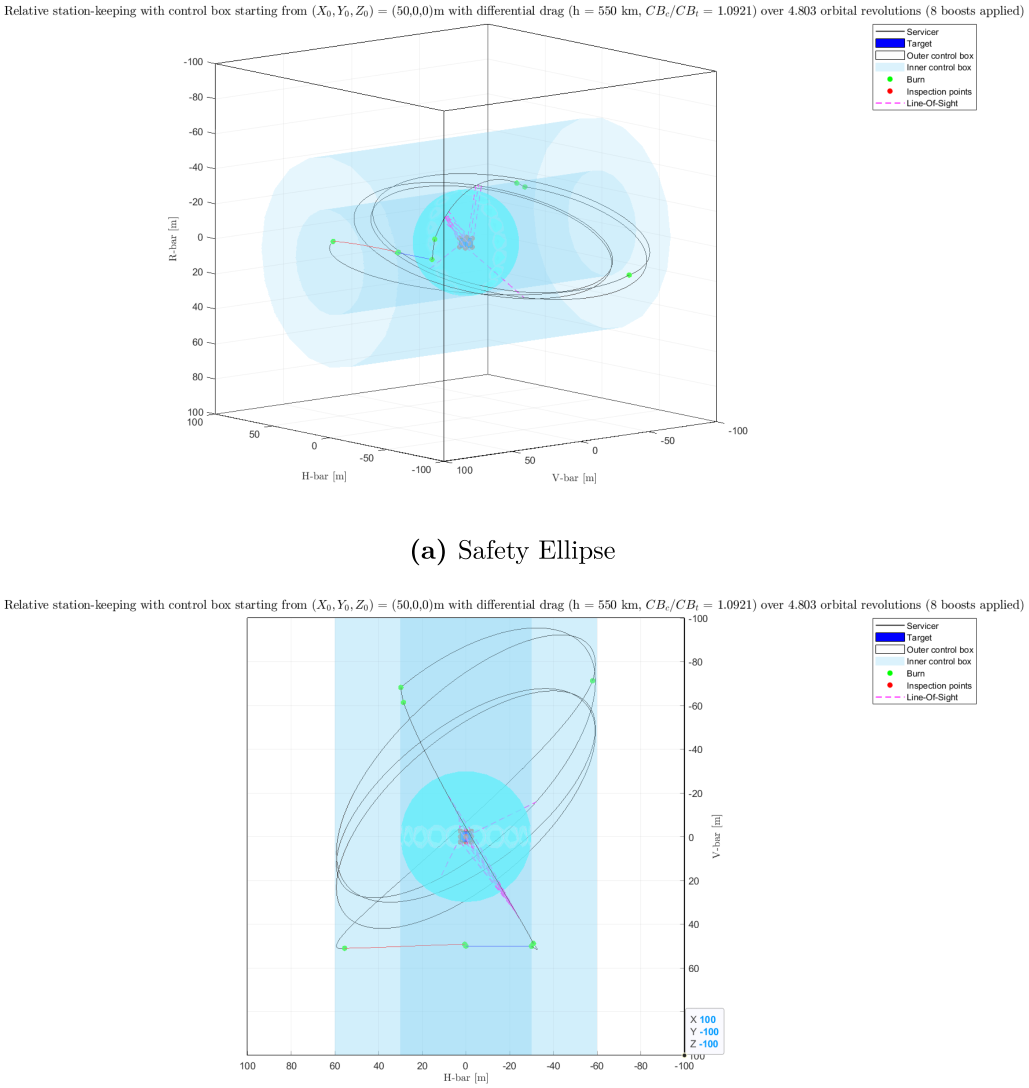
(a) Safety Ellipse

(b) Out-of-plane view (x,y) of the motion of the SE

Figure 4.22: Optimized trajectory of the Servicer around the Target with constraints on the control boxes, FOV, LOS occlusion, SE, and return (Case 4.2)

Parameters	**Simulation values**
Time of flight, t_f [s]	28632.462324
Total number of impulses applied, N	7
1$^{\text{st}}$ Optimal burn time [s]	0.000000
1$^{\text{st}}$ Optimal ΔV [m/s]	[0.000000, -0.002039, 0.009477]
2$^{\text{nd}}$ Optimal burn time [s]	1147.901596
2$^{\text{nd}}$ Optimal ΔV [m/s]	[-0.010205, 0.012993, -0.015738]
3$^{\text{rd}}$ Optimal burn time [s]	10353.180071
3$^{\text{rd}}$ Optimal ΔV [m/s]	[0.000104, -0.036424, 0.017506]
4$^{\text{th}}$ Optimal burn time [s]	11513.770505
4$^{\text{th}}$ Optimal ΔV [m/s]	[-0.014062, 0.024328, -0.016755]
5$^{\text{th}}$ Optimal burn time [s]	26390.037122
5$^{\text{th}}$ Optimal ΔV [m/s]	[0.089627, -0.000029, -0.000000]
6$^{\text{th}}$ Optimal burn time [s]	28212.959866
6$^{\text{th}}$ Optimal ΔV [m/s]	[-0.038023, -0.005538, -0.006056]
7$^{\text{th}}$ Optimal burn time [s]	28632.462324
7$^{\text{th}}$ Optimal ΔV [m/s]	[0.002919, 0.048595, 0.000007]
Total ΔV [m/s]	0.282847
Optimal drag area, A_c [m^2]	9.710563
Optimal ballistic coefficient, CB_c [kg/m^2]	93.618763

Table 4.34: Optimization results (Case 4.2)



(a) Safety Ellipse

(b) Out-of-plane view (x,y) of the motion of the SE

Figure 4.23: Optimized trajectory of the Servicer around the Target with constraints on the control boxes, FOV, LOS occlusion, SE, and return (Case 4.3)

Parameters	**Simulation values**
Time of flight, t_f [s]	27562.069162
Total number of impulses applied, N	8
1$^{\text{st}}$ Optimal burn time [s]	0.000000
1$^{\text{st}}$ Optimal ΔV [m/s]	[0.000000, 0.009366, -0.004801]
2$^{\text{nd}}$ Optimal burn time [s]	1154.212816
2$^{\text{nd}}$ Optimal ΔV [m/s]	[0.000297, -0.008626, -0.008982]
3$^{\text{rd}}$ Optimal burn time [s]	1826.503970
3$^{\text{rd}}$ Optimal ΔV [m/s]	[0.003652, -0.000024, -0.034457]
4$^{\text{th}}$ Optimal burn time [s]	3992.151301
4$^{\text{th}}$ Optimal ΔV [m/s]	[-0.003684, -0.001515, -0.006200]
5$^{\text{th}}$ Optimal burn time [s]	4158.481741
5$^{\text{th}}$ Optimal ΔV [m/s]	[-0.060363, 0.000101, -0.001680]
6$^{\text{rd}}$ Optimal burn time [s]	23476.779701
6$^{\text{rd}}$ Optimal ΔV [m/s]	[-0.000122, 0.000000, -0.011035]
7$^{\text{th}}$ Optimal burn time [s]	26485.254293
7$^{\text{th}}$ Optimal ΔV [m/s]	[-0.002008, -0.000647, -0.010176]
8$^{\text{th}}$ Optimal burn time [s]	27562.069162
8$^{\text{th}}$ Optimal ΔV [m/s]	[0.005277, 0.015180, -0.000007]
Total ΔV [m/s]	0.162885
Optimal drag area, A_c [m^2]	9.157002
Optimal ballistic coefficient, CB_c [kg/m^2]	99.278222

Table 4.35: Optimization results (Case 4.3)

4.3 Docking Phase

The final step was to optimize the docking maneuver with the Target, starting from the state acquired by the Servicer after completing the inspection and returning to the initial position. For this simulation, during the approach phase, the Servicer was constrained to move within a conical approach corridor originating from the Target's mating point, as shown in Figure 4.24.

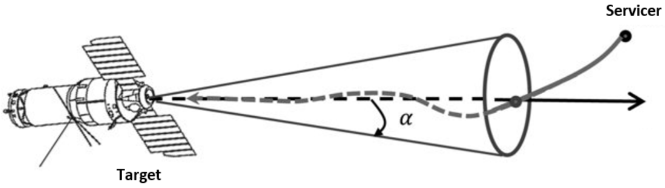


Figure 4.24: Conic approach corridor aligned with the docking axis and with an apex half-angle α [26]

In the initial iterations, to assist the optimizer in finding feasible solutions to be used subsequently as initial guesses with `fmincon`, a secondary cost function was defined, in addition to the one defined in Equation 4.1, specifically aimed at minimizing the duration of the maneuver:

$$J = \sum_{i=1}^{N-1} \Delta t_i \tag{4.18}$$

The constraints considered to ensure that the Servicer approaches the Target while remaining within the conical corridor have been formulated similarly to the constraints on FOV and LOS occlusion defined in Equations 4.10 and 4.11:

$$\frac{(\mathbf{S_f} - \mathbf{D_0}) \cdot \mathbf{c}}{d_{\text{r,app}} \cdot d_{\text{c}}} \geq \cos(\alpha) \tag{4.19}$$

where:

- $\mathbf{S_f}$ is the position vector of the Servicer.
- $\mathbf{D_0}$ is the position vector of the target docking port.
- $\mathbf{c}$ is the cone axis vector.
- $\mathbf{d}_{\text{r,app}} = \|\mathbf{S_f} - \mathbf{D_0}\|$ is the magnitude of the vector connecting the Servicer to the docking port on the Target.
- $\mathbf{d_c} = \|\mathbf{c}\|$ is the magnitude of the cone axis vector.

- $\alpha = 15\,^{\circ}$ is the half cone angle.

The preferred approach for docking with a target spacecraft is typically along the ±V-bar axis, which aligns with the direction of the Target's orbital velocity. This approach is favored due to its inherent safety and ease of control. Approaching along the ±V-bar axis allows for gradual adjustments in speed and trajectory, reducing the risk of collision and ensuring smoother docking operations ([16]).

Direct approaches along the ±H-bar axis are considered inherently unsafe due to the potential for uncontrolled drift towards the target in the event of a failure. This drift can lead to increased relative speeds and potential collisions. Therefore, ±H-bar approaches are generally avoided ([16]).

In some cases, an R-bar approach may be utilized, particularly when transitioning from a higher to a lower orbit or for specific berthing operations. However, this method often requires initial maneuvers along the V-bar axis to ensure proper alignment and safety before transitioning to the R-bar approach ([16]).

To avoid making the optimization problem too computationally intensive, the inspection phase and the docking phase were optimized separately. Different locations of the docking port were considered, specifically along the -V-bar and ±R-bar, in order to test the robustness of the code. This choice is not primarily about selecting the most favorable docking port, but rather about testing the robustness of the code by challenging the Servicer to execute more challenging maneuvers.

To facilitate the identification of feasible solutions, it was decided to shift the starting point of the docking phase along the V-bar. This adjustment allows for better maneuver planning and reduces the likelihood of violating the operational constraints of this phase.

After defining the initial state of the Servicer with respect to the Target, as well as the dimensions of the control boxes, all the plots are presented along with two tables, which specify respectively the location of the docking port and the simulation results provided by the optimization tool.

Parameters	**Target**	**Servicer**
Initial position (x_0, y_0, z_0) [m]	(0, 0, 0)	(80, 0, 0)
Initial velocity ($\dot{x}_0$, $\dot{y}_0$, $\dot{z}_0$) [m/s]	(0, 0, 0)	(0, 0, 0)

Table 4.36: Initial state of the spacecraft (Cases 5.1, 5.2, 5.3)

Control boxes	Dimensions
Cubic outer control box	Side length = 150 m
Spherical inner control box	Radius = 30 m

Table 4.37: Shape and dimensions of the control boxes (Cases 5.1, 5.2, 5.3)

Docking port	
Origin	(-2.5, 0, 0) m
Axis direction	-V-bar

Table 4.38: Conic approach corridor (Case 5.1)

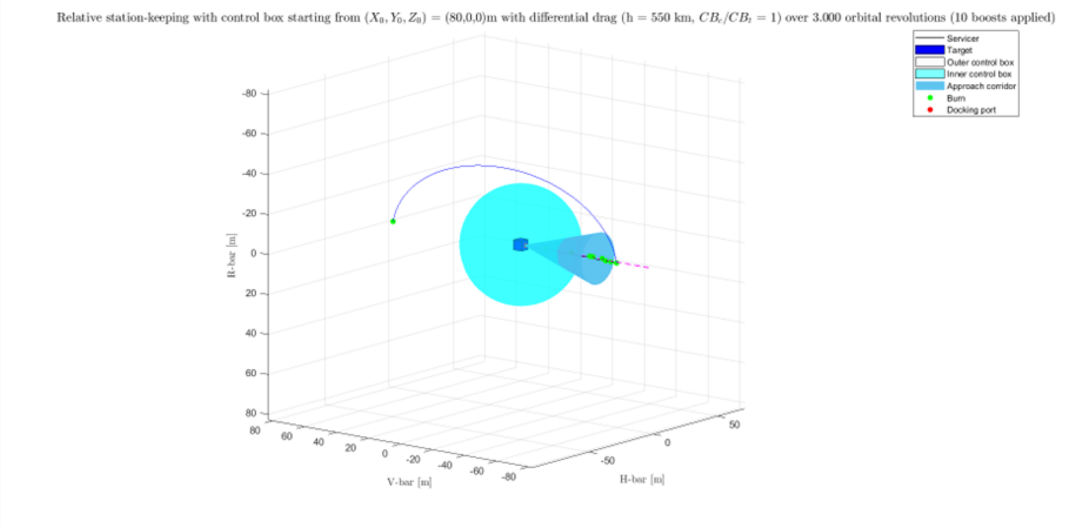


Figure 4.25: Optimized trajectory of the Servicer around the Target for docking within a conic approach corridor (Case 5.1)

Parameters	**Simulation values**
Time of flight, t_f [s]	17216.989628
Total number of impulses applied, N	10
1^{st} Optimal burn time [s]	0.000000
1^{st} Optimal ΔV [m/s]	[-0.014468, 0.014194, -0.000000]
2^{nd} Optimal burn time [s]	2091.309174
2^{nd} Optimal ΔV [m/s]	0.010196, -0.001141, -0.001253]
3^{rd} Optimal burn time [s]	6375.852591
3^{rd} Optimal ΔV [m/s]	[-0.007535, 0.033044, 0.036312]
4^{th} Optimal burn time [s]	10378.181738
4^{th} Optimal ΔV [m/s]	[-0.000000, 0.000000, -0.000000]
5^{th} Optimal burn time [s]	10593.777253
5^{th} Optimal ΔV [m/s]	[-0.000000, -0.000001, 0.000003]
6^{st} Optimal burn time [s]	11238.570101
6^{st} Optimal ΔV [m/s]	[-0.000001, 0.000001, -0.000004]
7^{nd} Optimal burn time [s]	11421.618483
7^{nd} Optimal ΔV [m/s]	[0.000007, -0.000000, -0.063039]
8^{rd} Optimal burn time [s]	16650.366573
8^{rd} Optimal ΔV [m/s]	[-0.063287, 0.000000, -0.002264]
9^{th} Optimal burn time [s]	16985.400127
9^{th} Optimal ΔV [m/s]	[0.013238, 0.007727, 0.001619]
10^{th} Optimal burn time [s]	17216.989628
10^{th} Optimal ΔV [m/s]	[0.012315, 0.029763, 0.000000]
Total ΔV [m/s]	0.254273
Optimal drag area, A_c [m^2]	9.999990
Optimal ballistic coefficient, CB_c [kg/m^2]	90.909180

Table 4.39: Optimization results (Case 5.1)

Docking port	
Origin	(0, 0, -2.5) m
Axis direction	-R-bar

Table 4.40: Conic approach corridor (Case 5.2)

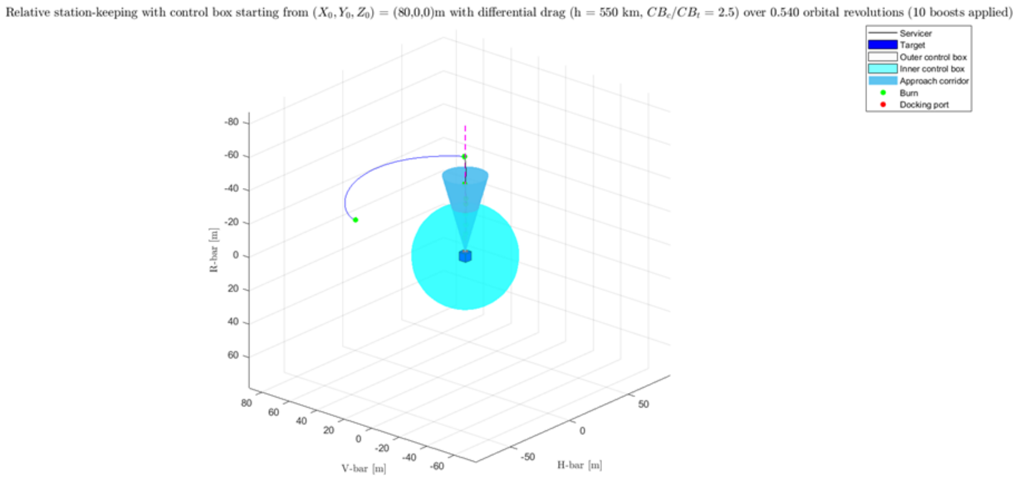


Figure 4.26: Optimized trajectory of the Servicer around the Target for docking within a conic approach corridor (Case 5.2)

Parameters	**Simulation values**
Time of flight, t_f [s]	3096.243067
Total number of impulses applied, N	10
1st Optimal burn time [s]	0.000000
1st Optimal ΔV [m/s]	[0.016135, -0.000000, 0.000833]
2nd Optimal burn time [s]	2316.099193
2nd Optimal ΔV [m/s]	[0.071815, -0.000011, 0.100000]
3rd Optimal burn time [s]	2317.099193
3rd Optimal ΔV [m/s]	[0.017569, 0.000006, 0.025330]
4th Optimal burn time [s]	2517.946752
4th Optimal ΔV [m/s]	[-0.032808, 0.000019, 0.015525]
5th Optimal burn time [s]	2517.846757
5th Optimal ΔV [m/s]	[-0.004795, -0.000000, 0.001711]
6st Optimal burn time [s]	2686.048456
6st Optimal ΔV [m/s]	[-0.008830, -0.000009, 0.002840]
7nd Optimal burn time [s]	2793.833947
7nd Optimal ΔV [m/s]	[-0.018909, -0.000004, 0.004940]
8rd Optimal burn time [s]	2913.029980
8rd Optimal ΔV [m/s]	[-0.008688, 0.000000, 0.001871]
9th Optimal burn time [s]	3027.944429
9th Optimal ΔV [m/s]	[-0.000000, -0.000000, 0.000000]
10th Optimal burn time [s]	3096.243067
10th Optimal ΔV [m/s]	[0.000000, -0.000000, 0.000000]
Total ΔV [m/s]	0.249194
Optimal drag area, A_c [m^2]	4.000000
Optimal ballistic coefficient, CB_c [kg/m^2]	227.272727

Table 4.41: Optimization results (Case 5.2)

Docking port	
Origin	(0, 0, 2.5) m
Axis direction	R-bar

Table 4.42: Conic approach corridor (Case 5.3)

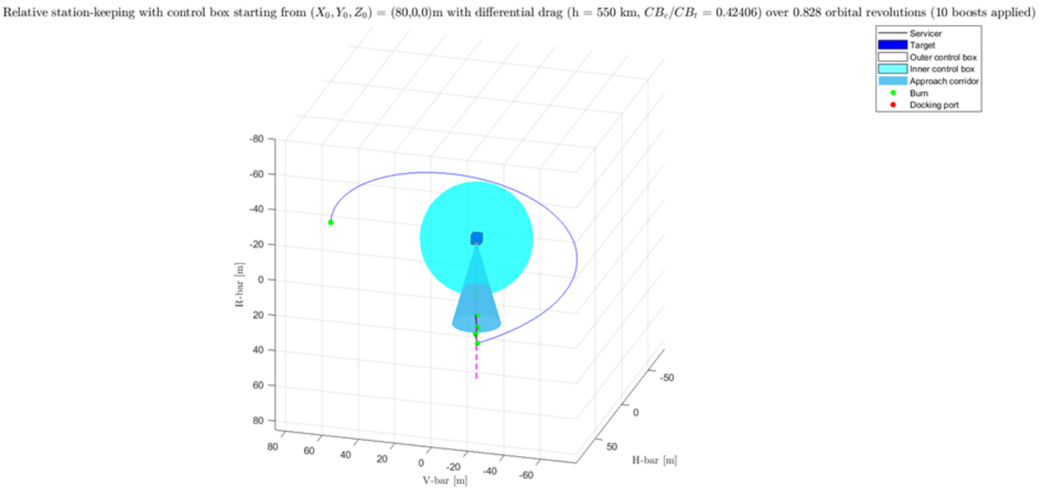


Figure 4.27: Optimized trajectory of the Servicer around the Target for docking within a conic approach corridor (Case 5.3)

Parameters	**Simulation values**
Time of flight, t_f [s]	17216.989628
Total number of impulses applied, N	10
1st Optimal burn time [s]	0.000000
1st Optimal ΔV [m/s]	[-0.014468, 0.000000, -0.063039]
2nd Optimal burn time [s]	2091.309174
2nd Optimal ΔV [m/s]	[0.014194, -0.000000, -0.063287]
3rd Optimal burn time [s]	6375.852951
3rd Optimal ΔV [m/s]	[-0.000000, -0.000000, 0.000000]
4th Optimal burn time [s]	10378.181738
4th Optimal ΔV [m/s]	[0.010196, -0.000001, -0.002264]
5th Optimal burn time [s]	10593.777253
5th Optimal ΔV [m/s]	[-0.001141, 0.000003, 0.013238]
6st Optimal burn time [s]	11238.570101
6st Optimal ΔV [m/s]	[-0.001253, -0.000001, 0.007727]
7nd Optimal burn time [s]	11421.618483
7nd Optimal ΔV [m/s]	[-0.007535, 0.000001, 0.001619]
8rd Optimal burn time [s]	16650.366573
8rd Optimal ΔV [m/s]	[0.033044, -0.000004, 0.012315]
9th Optimal burn time [s]	16985.400127
9th Optimal ΔV [m/s]	[0.036312, 0.000007, 0.029763]
10th Optimal burn time [s]	17216.989628
10th Optimal ΔV [m/s]	[-0.000000, -0.000000, 0.000000]
Total ΔV [m/s]	0.251019
Optimal drag area, A_c [m^2]	9.999999
Optimal ballistic coefficient, CB_c [kg/m^2]	90.909180

Table 4.43: Optimization results (Case 5.3)

4.4 Inspection of an Asteroid

To further test the robustness of the optimization tool, we modeled the Target as an ellipsoid with arbitrary dimensions, listed in Table 4.45. The number of inspection points was increased from 14 to 32, chosen randomly but in sufficient quantity to achieve satisfactory coverage of the Target. It is important to note that this different modeling of the Target affects only its shape and size, while all other parameters listed in Table 4.1 (orbital parameters, mass, ballistic coefficient) remain unchanged. In terms of constraints, all previously discussed constraints (control boxes, FOV, LOS occlusion, SE, and return) were considered.

Figure 4.28 illustrates the Servicer's trajectory around the ellipsoid-shaped Target, showcasing the algorithm's capability to handle varied target shapes.

Parameters	**Target**	**Servicer**
Initial position (x_0, y_0, z_0) [m]	(0, 0, 0)	(80, 0, 0)
Initial velocity ($\dot{x}_0$, $\dot{y}_0$, $\dot{z}_0$) [m/s]	(0, 0, 0)	(0, 0, 0)

Table 4.44: Initial state of the spacecraft (Case 6.1)

Target dimensions	
Semi-axes length [m]	5 x 5 x 1.5
Orientation around x, y, z [°]	(0, 45, 45)

Table 4.45: Dimensions of the Target (Case 6.1)

Control boxes	**Dimensions**
Cubic outer control box	Side length = 150 m
Spherical inner control box	Radius = 30 m
Cylindrical shell	Inner radius = 30 m Outer radius = 40 m Heigth = 150 m

Table 4.46: Shape and dimensions of the control boxes (Case 6.1)

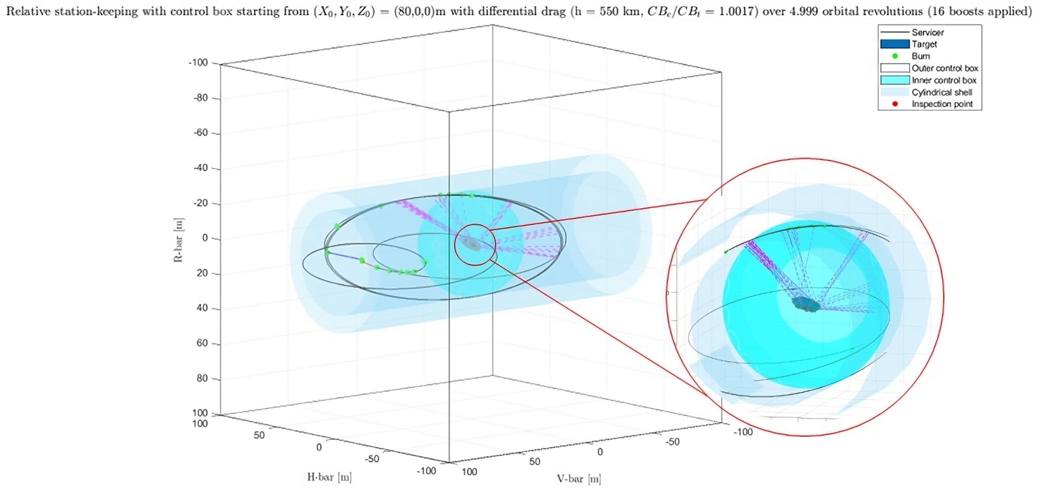


Figure 4.28: Optimized trajectory of the Servicer around the Target with constraints on the control boxes, FOV, LOS occlusion, SE, and return (Case 6.1)

Parameters	**Simulation values**
Time of flight, t_f [s]	28688.203720
Total number of impulses applied, N	16
1st Optimal burn time [s]	0.000000
1st Optimal ΔV [m/s]	[0.000000, 0.034240, 0.000000]
2nd Optimal burn time [s]	1173.081731
2nd Optimal ΔV [m/s]	[0.007821, 0.001709, -0.015599]
3rd Optimal burn time [s]	1908.819190
3rd Optimal ΔV [m/s]	[-0.006354, -0.000324, -0.004771]
4th Optimal burn time [s]	3150.203717
4th Optimal ΔV [m/s]	[-0.001442, -0.000267, 0.000405]
5th Optimal burn time [s]	13876.099504
5th Optimal ΔV [m/s]	[0.000000, 0.000000, 0.000000]
6th Optimal burn time [s]	14363.102613
6th Optimal ΔV [m/s]	[0.000467, -0.000027, 0.000076]

Table 4.47: Optimization results (Case 6.1) - Part 1

Parameters	**Simulation values**
7th Optimal burn time [s]	14432.187208
7th Optimal ΔV [m/s]	[0.000084, 0.000000, 0.000000]
8th Optimal burn time [s]	14525.441990
8th Optimal ΔV [m/s]	[0.002060, -0.000047, 0.000093]
9rd Optimal burn time [s]	16900.863803
9rd Optimal ΔV [m/s]	[-0.005752, 0.000783, -0.001453]
10th Optimal burn time [s]	27089.759470
10th Optimal ΔV [m/s]	[0.007889, 0.001067, -0.002551]
11th Optimal burn time [s]	27675.231870
11th Optimal ΔV [m/s]	[0.000000, 0.000000, 0.000000]
12th Optimal burn time [s]	27845.816668
12th Optimal ΔV [m/s]	[0.000000, 0.000000, 0.000000]
13rd Optimal burn time [s]	27985.356908
13rd Optimal ΔV [m/s]	[0.000000, 0.000000, 0.000000]
14th Optimal burn time [s]	28188.000783
14th Optimal ΔV [m/s]	[0.000000, 0.000000, 0.000000]
15th Optimal burn time [s]	28415.612438
15th Optimal ΔV [m/s]	[0.000000, 0.000000, 0.000000]
16th Optimal burn time [s]	28688.203720
16th Optimal ΔV [m/s]	[0.000000, 0.000000, 0.000000]
Total ΔV [m/s]	0.078210
Optimal drag area, A_c [m^2]	9.983421
Optimal ballistic coefficient, CB_c [kg/m^2]	91.060056

Table 4.48: Optimization results (Case 6.1) - Part 2

Subsequently, the 3D model of the asteroid Didymos was employed. Didymos, a near-Earth asteroid belonging to the Apollo group, was chosen due to its significant scientific interest and the complexity posed by its considerable size and irregular surface, which present substantial challenges in identifying optimal inspection trajectories. To facilitate the optimizer's work, only the constraints related to the control boxes, FOV, and LOS occlusion were considered. To accommodate the size of the Target, the limit values for Equations 4.10 and 4.11 were modified as follows:

- $\mathbf{d}_{\text{observe}} = 500\,\text{m}$,
- $\theta_{\max} = 120\,°$,
- $\theta_{\text{occlusion}} = 120\,°$.

so that a large area of the asteroid is covered in a single image.

In terms of constraints, only those related to control boxes, FOV, and LOS occlusion were considered. As stated for the previous case, it is important to note that this different modeling of the Target affects only its shape and size, while all other parameters listed in Table 4.1 (orbital parameters, mass, ballistic coefficient) remain unchanged. The system's dynamic model, specifically the Hill relative model for near-circular orbits in LEO, has not been changed. However, this model may not be sufficiently accurate for simulating the Servicer's trajectory around Didymos due to the unique dynamic conditions of asteroids. Therefore, future implementations should include the gravitational force of the asteroid, potentially addressing the three-body problem, to provide a more accurate simulation of the Servicer's trajectory around Didymos. Additionally, only a very limited number of inspection points were considered, due to the complexity of the problem.

Parameters	**Target**	**Servicer**
Initial position (x_0, y_0, z_0) [m]	(0, 0, 0)	(640, 0, 0)
Initial velocity ($\dot{x}_0$, $\dot{y}_0$, $\dot{z}_0$) [m/s]	(0, 0, 0)	(0, 0, 0)

Table 4.49: Initial state of the spacecraft

Target dimensions	
Semi-axes length [m]	516 x 518 x 488

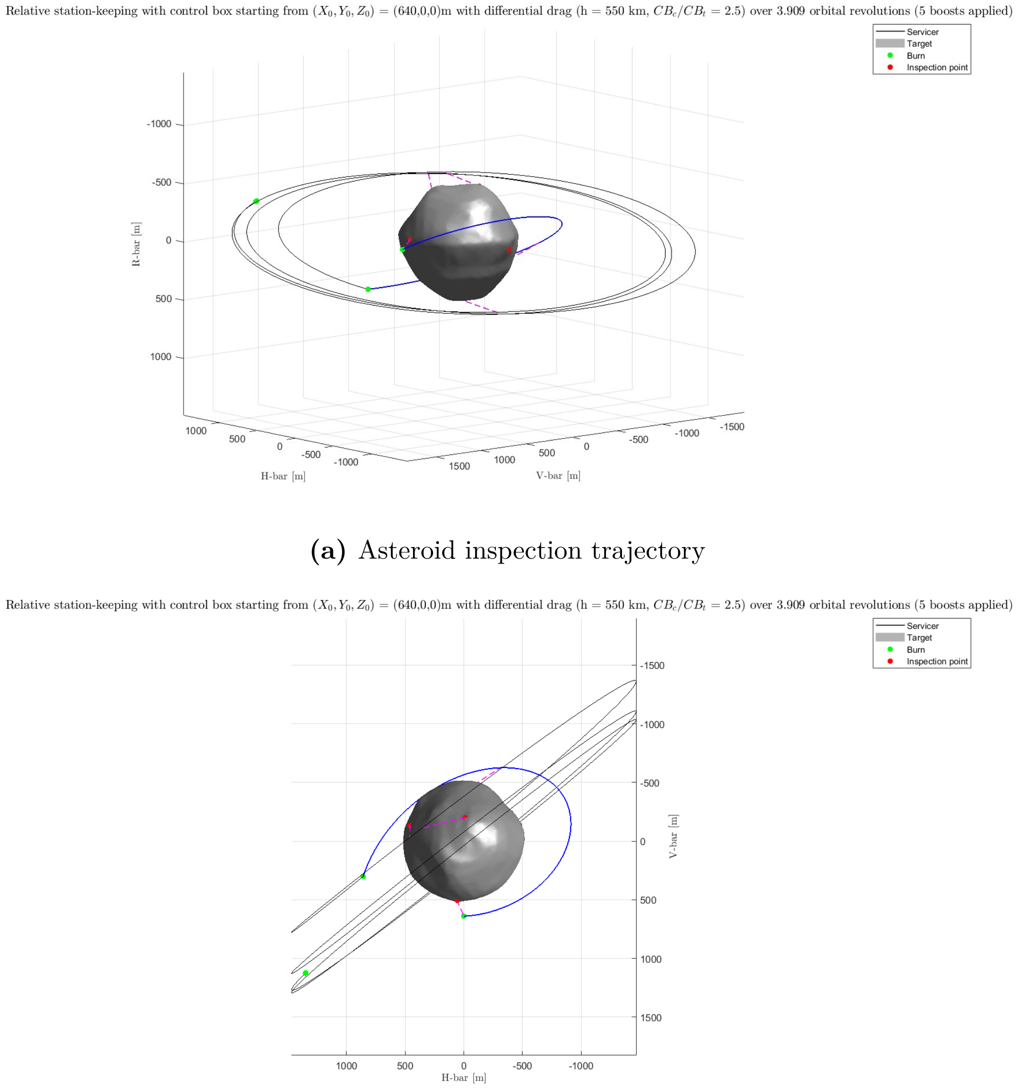
Table 4.50: Dimensions of the Target (Case 6.2)

Control boxes	**Dimensions**
Cubic outer control box	Side length = 3000 m
Spherical inner control box	Radius = 570 m

Table 4.51: Shape and dimensions of the control boxes (Case 6.2)

Parameters	**Simulation values**
Time of flight, t_f [s]	21458.882603
Total number of impulses applied, N	5
1st Optimal burn time [s]	0.000000
1st Optimal ΔV [m/s]	[0.641365, -0.602313, 0.585671]
2nd Optimal burn time [s]	165.934536
2nd Optimal ΔV [m/s]	[-0.662803, 0.176192, -0.300884]
3rd Optimal burn time [s]	6850.928934
3rd Optimal ΔV [m/s]	[-0.392839, -0.560084, -0.582511]
4th Optimal burn time [s]	7372.481582
4th Optimal ΔV [m/s]	[0.414169, 0.620834, -0.551894]
5th Optimal burn time [s]	21458.882603
5th Optimal ΔV [m/s]	[0.605800, -0.605800, -0.605800]
Total ΔV [m/s]	4.681865
Optimal drag area, A_c [m^2]	10.820987
Optimal ballistic coefficient, CB_c [kg/m^2]	84.011828

Table 4.52: Optimization results (Case 6.2)



(a) Asteroid inspection trajectory

(b) Out-of-plane view (x,y) of the Servicer's motion

Figure 4.29: Optimized trajectory of the Servicer around the Target with constraints on the control boxes, FOV and LOS occlusion (Case 6.2)

4.5 Genetic Algorithm Refinement

An additional development involved using genetic algorithms, specifically the MATLAB function `ga`, to refine the solutions obtained from `fmincon`, combined with `multistart`, by iteratively improving the population of potential solutions. As discussed in Chapter 2, genetic algorithms are particularly advantageous for their ability to handle complex, multi-modal landscapes and find the global minimum in the presence of multiple local minima.

Initial solutions were obtained from `fmincon` combined with `multistart`, exemplifying the scenario shown in Figure 4.21. These solutions were used to form the initial population for `gamultiobj`. From a graphical standpoint, the resulting trajectory appears identical, but as shown in Table 4.53, this method achieves a lower objective function value, specifically a lower total ΔV. However, it was found to be slower for our specific case, which limited its applicability.

Parameters	**Simulation values**
Time of flight, t_f [s]	28694.559539
Total number of impulses applied, N	6
1st Optimal burn time [s]	0.000000
1st Optimal ΔV [m/s]	[0.000000, 0.034546, 0.000000]
2nd Optimal burn time [s]	1147.829666
2nd Optimal ΔV [m/s]	[-0.020347, 0.018836, -0.024921]
3rd Optimal burn time [s]	2993.327898
3rd Optimal ΔV [m/s]	[0.021634, -0.009584, -0.051412]
4th Optimal burn time [s]	7791.059156
4th Optimal ΔV [m/s]	[0.000000, 0.000000, 0.000000]
5th Optimal burn time [s]	15732.227750
5th Optimal ΔV [m/s]	[-0.001043, 0.018736, 0.037431]
6th Optimal burn time [s]	28694.559539
6th Optimal ΔV [m/s]	[0.000000, 0.000000, 0.000000]
Total ΔV [m/s]	0.170294
Optimal drag area, A_c [m^2]	10.031143
Optimal ballistic coefficient, CB_c [kg/m^2]	90.626855

Table 4.53: Optimization results (Case 7)

4.6 Pareto Front

The final test conducted focused on the Pareto front, which represents the trade-off between the total velocity increment ΔV and the number of impulses applied N. The trend observed in Figure 4.30 validates the effectiveness of the optimization code.

The Pareto front was generated using the `gamultiobj` function from MATLAB's Global Optimization Toolbox, which employs a genetic algorithm. Initial solutions were obtained from `fmincon`, combined with the `multistart` function, simulating the simplest scenario where the only constraints are the inner and outer control boxes. These solutions were used to form the initial population for `gamultiobj`. The ballistic coefficient CB_c was fixed at 14 to ensure that the Servicer moves to counterbalance the effect of differential aerodynamic drag.

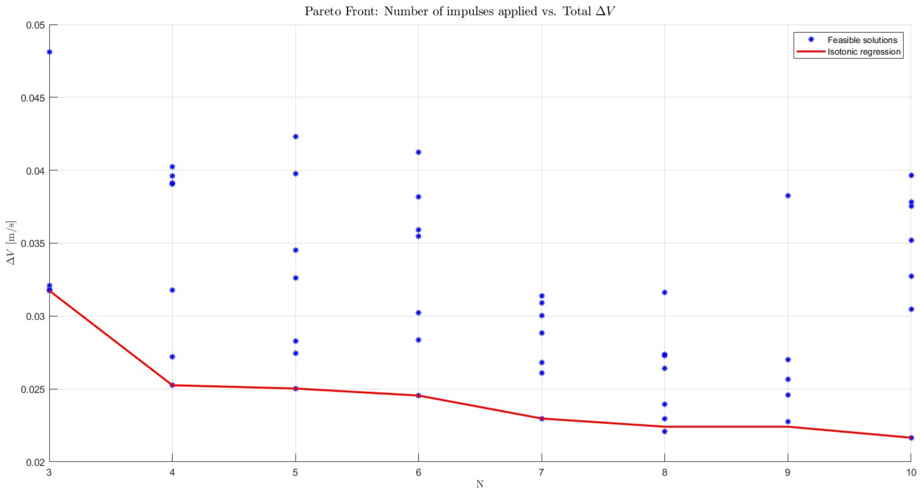


Figure 4.30: Pareto front showing the trade-off between total ΔV and number of impulses N

Figure 4.30 shows that as the number of impulses increases, the total ΔV required decreases, indicating more efficient propellant usage with a higher number of impulses.

Chapter 5

Conclusions and Future Developments

This thesis focused on optimizing in-orbit inspection relative trajectories between two spacecraft. Through a gradual increase in the complexity of the optimization problem, the developed tool has demonstrated the ability to effectively employ direct methods for optimizing IOS trajectories in LEO. The developed tool not only exhibited robust performance and adaptability to various operational scenarios, but also showed significant potential for commercial applications.

The whole process involved several stages: defining the optimization problem, formulating the physical model, developing an optimization approach, and implementing and testing the method across various mission scenarios.

The key results of this thesis highlight the effectiveness of the proposed optimization techniques for the preliminary design of in-orbit inspection missions.

Based on the findings of this research, several paths for future work have been identified. Future studies should incorporate higher-fidelity dynamics to account for perturbations such as aerodynamic drag using an atmospheric database, the J2 effect, and solar radiation pressure. This will allow for more accurate simulation of the relative motion between the Servicer and the Target. The implementation of a continuous thrust model would improve the accuracy of the Servicer's maneuvers, and consequently, it would yield more realistic optimized trajectories. This approach was briefly tested by introducing new control variables, i.e., the acceleration along the three axis (constant in magnitude and equal to thrust/mass and the thrust duration, as shown in Figure 5.1.

Exploring advanced optimization techniques, such as machine learning-based approaches, could further enhance the robustness and efficiency of trajectory planning. These techniques have the potential to provide better initial guesses and handle more complex optimization landscapes.

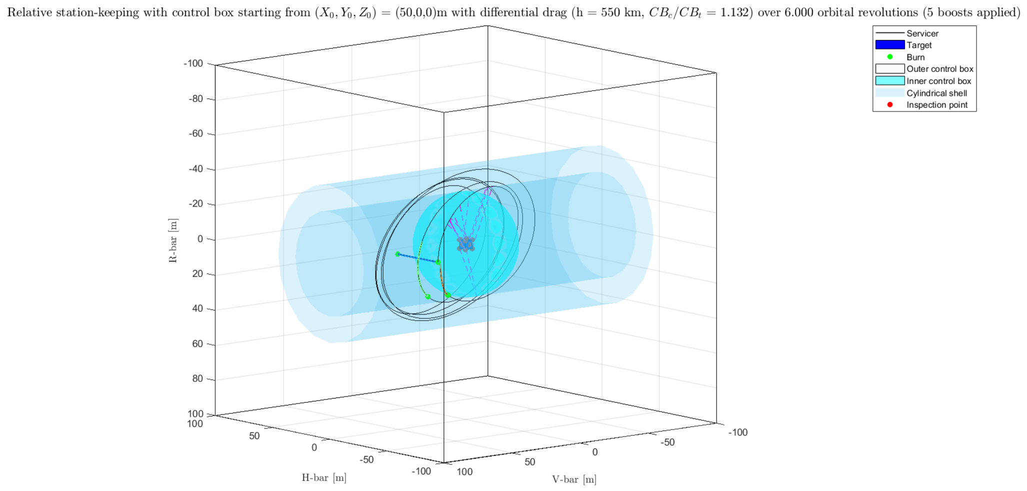


Figure 5.1: Continuous thrust modeling for more realistic simulation: the colored segments represent the propulsive arcs of the trajectory

Validating the developed algorithms with actual space missions will be crucial to assess their performance under real-world conditions. This validation will help identify practical challenges and refine the optimization tool accordingly. Additionally, applying the optimization framework to more complex mission scenarios, including multi-target servicing and simultaneous operations of multiple servicers, can expand its applicability. This will test the tool's scalability and flexibility in handling more intricate mission requirements.

In conclusion, this thesis has established a solid foundation for optimizing IOS trajectories in LEO. The developed tool not only demonstrates the feasibility and effectiveness of the proposed methods but also opens up numerous possibilities for future research and practical applications in the field of space mission analysis and operations.

I would like to conclude this thesis with a quote that deeply resonates with me, spoken by Lyndon B. Johnson, who later became the President of the United States, when he was the Senate Majority Leader and Chairman of the Special Committee on Space and Astronautics [27]:

> *"There are no blueprints or road maps which clearly mark out the course... The limits of the job are no less than the limits of the universe. And those are limits which can be stated but are virtually impossible to describe."*

Bibliography

[1] *Thales Alenia Space guiderà il progetto europeo EROSS IOD per l'In-Orbit Servicing.* URL: https://www.astrospace.it/2022/09/21/thales-alenia-space-guidera-il-progetto-europeo-eross-iod-per-lin-orbit-servicing/ (cit. on p. 1).

[2] *ESA moves ahead with In-Orbit Servicing missions.* URL: https://www.esa.int/Enabling_Support/Preparing_for_the_Future/Discovery_and_Preparation/ESA_moves_ahead_with_In-Orbit_Servicing_missions2 (cit. on pp. 1, 5, 7).

[3] *Missions to Hubble.* URL: https://science.nasa.gov/mission/hubble/observatory/missions-to-hubble/ (cit. on pp. 3, 4).

[4] *ESA advances Clearspace-1 development.* URL: https://www.esa.int/Space_Safety/ClearSpace-1/ESA_advances_Clearspace-1_development (cit. on p. 5).

[5] *Servicing spacecraft approaches a satellite.* URL: https://www.esa.int/ESA_Multimedia/Images/2023/07/Servicing_spacecraft_approaches_a_satellite (cit. on p. 6).

[6] *ClearSpace-1 captures Vespa.* URL: https://www.esa.int/ESA_Multimedia/Images/2020/11/ClearSpace-1_captures_Vespa (cit. on p. 6).

[7] *MEV-1 and 2 (Mission Extension Vehicle-1 and -2).* URL: https://www.eoportal.org/satellite-missions/mev-1#eop-quick-facts-section (cit. on pp. 5, 7).

[8] *Firmato il contratto ASI-Thales Alenia Space per la prima missione italiana di In Orbit-Servicing.* URL: https://www.asi.it/2023/05/firmato-il-contratto-asi-thales-alenia-space-per-la-prima-missione-italiana-di-in-orbit-servicing/ (cit. on p. 6).

[9] Giorgio Fasano and János D. Pintér, eds. *Modeling and Optimization in Space Engineering.* Vol. 73. Springer Optimization and Its Applications. New York: Springer, 2013. DOI: 10.1007/978-1-4614-4469-5 (cit. on p. 9).

[10] Danylo Malyuta, Taylor P. Reynolds, Michael Szmuk, Thomas Lew, Riccardo Bonalli, Marco Pavone, and Behçet Açıkmeşe. «Convex Optimization for Trajectory Generation: A Tutorial on Generating Dynamically Feasible Trajectories Reliably and Efficiently». In: *IEEE Control Systems Magazine* 42.5 (Oct. 2022), pp. 40–113. DOI: `10.1109/MCS.2022.3187542` (cit. on p. 9).

[11] Abolfazl Shirazi, Spain Josu Ceberio, and Spain Jose A. Lozano. «Spacecraft Trajectory Optimization: A review of Models, Objectives, Approaches and Solutions». In: *Progress in Aerospace Sciences* 102 (2018). DOI: `10.1016/j.asr.2023.10.0107` (cit. on pp. 9, 14, 33).

[12] *Unlocking the Power of Concave and Convex Functions in Machine Learning.* URL: `https://towardsai.net/p/machine-learning/unlocking-the-power-of-concave-and-convex-functions-in-machine-learning` (cit. on p. 10).

[13] Peter Korošec, Jurij Silc, Marian Vajtersic, and Rade Kutil. «A Shared-Memory ACO-Based Algorithm for Numerical Optimization». In: *Proc. of the 25th IEEE International Symposium on Parallel and Distributed Processing.* May 2011, pp. 352–357. DOI: `10.1109/IPDPS.2011.176` (cit. on p. 15).

[14] Mathieu Balesdent. «Multidisciplinary Design Optimization of Launch Vehicles». PhD thesis. École Centrale de Nantes, Nov. 2011 (cit. on pp. 16, 17).

[15] Juan Carretero and Meyer Nahon. «A genetic algorithm for calculating minimum distance between convex and concave bodies». In: *Proc. of the 6th International Symposium on Artificial Intelligence, Robotics and Automation in Space.* June 2001 (cit. on p. 18).

[16] Wigbert Fehse. *Automated Rendezvous and Docking of Spacecraft.* Ed. by Michael J. Rycroft and Wei Shyy. Cambridge Aerospace Series 16. The Edinburgh Building, Cambridge CB2 2RU, United Kingdom: Cambridge University Press, 2003 (cit. on pp. 20, 22–28, 57, 59, 74).

[17] Nicoletta Bloise, Elisa Capello, Matteo Dentis, and Elisabetta Punta. «Obstacle Avoidance with Potential Field Applied to a Rendezvous Maneuver». In: *Applied Sciences* 7.1042 (Oct. 2017). DOI: `10.3390/app7101042` (cit. on p. 21).

[18] *INTRODUCTION TO AEROSPACE FLIGHT VEHICLES - ATMOSPHERIC PROPERTIES.* URL: `https://eaglepubs.erau.edu/introductiontoaerospaceflightvehicles/chapter/international-standard-atmosphere-isa/` (cit. on p. 29).

[19] Riccardo Bevilacqua and Marcello Romano. «Rendezvous Maneuvers of Multiple Spacecraft Using Differential Drag Under J(2) Perturbation». In: *Journal of Guidance Control and Dynamics* 31 (Aug. 2008), pp. 1595–1607. DOI: 10.2514/1.36362 (cit. on p. 30).

[20] Stephen Kemble. «Automated Rendezvous and Docking of Spacecraft». In: *Proceedings of the Institution of Mechanical Engineers, Part G: Journal of Aerospace Engineering* 221 (2007), pp. 997–997. URL: https://api.semanticscholar.org/CorpusID:56632051 (cit. on p. 31).

[21] Dr. Matthew C. Ruschmann, Dr. Brenton Duffy, Rafael de la Torre, and Dr. Sun Hur-Diaz. «EFFICIENT STATION-KEEPING FOR CLUSTER FLIGHT». In: 2014. URL: https://api.semanticscholar.org/CorpusID:164211509 (cit. on p. 36).

[22] Zhuojun Hou, Bohan Jiao, and Zhaohui Dang. «Relative orbit design of CubeSats for on-orbit visual inspection of China space station». In: *Advances in Space Research* 73.1 (2024), pp. 170–186. ISSN: 0273-1177. URL: https://doi.org/10.1016/j.asr.2023.10.010 (cit. on pp. 46, 47).

[23] Roberto Opromolla, Giancarmine Fasano, Giancarlo Rufino, and Michele Grassi. «Design of relative trajectories for in orbit proximity operations». In: *Acta Astronautica* 145 (2018), pp. 342–356. ISSN: 0094-5765. URL: https://doi.org/10.1016/j.actaastro.2018.01.062 (cit. on p. 57).

[24] David E. Gaylor and Brent William Barbee. «Algorithms for safe spacecraft proximity operations». In: *Advances in the Astronautical Sciences* 127 (2007), pp. 133–152 (cit. on p. 57).

[25] Bo Naasz. «Safety Ellipse Motion with Coarse Sun Angle Optimization». In: Oct. 2005 (cit. on p. 57).

[26] Ping Lu and Xinfu Liu. «Autonomous Trajectory Planning for Rendezvous and Proximity Operations by Conic Optimization». In: *Journal of Guidance Control and Dynamics* 36 (Mar. 2013), pp. 375–389. DOI: 10.2514/1.58436 (cit. on p. 73).

[27] T. Gangale. *Space Exploration in the United States: A Documentary History.* ABC-CLIO, 2020. ISBN: 9798216017165. URL: https://books.google.it/books?id=VFIVOAEACAAJ (cit. on p. 91).