POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Decentralized Spacecraft Formation Reconfiguration Exploiting a Shape-Based Method for Trajectory Optimization

Tesi di Laurea Magistrale in
Space Engineering - Ingegneria Spaziale

Author: **Matteo Brioschi**

Student ID: 969060
Advisor: Prof. Mauro Massari
Academic Year: 2021-22

Abstract

In recent years, the space segment has seen a fast increase in the number of satellites orbiting Earth as well as a reduction in the satellites' dimensions. With the availability of smaller satellites, formation-flying missions have also seen an increase in popularity due to their intrinsic higher flexibility and reliability. Additionally, it has been proven that, in many remote sensing applications, satellites flying in formation can achieve better performances than a monolithic platform. A requirement for satellites in formation flying is the capability of performing formation reconfiguration, when needed.

In this work, a tool has been developed to obtain, in a limited time, a strategy for formation reconfiguration with a decentralized approach. This could impact the space segment in two different ways: the first one is the increase in the autonomy of satellites from ground stations, the second one is the provision of an initial guess for more refined solvers. To achieve this goal, a genetic algorithm, that relies on shifting the initial maneuvering times of the involved satellites, has been exploited to obtain a strategy for formation reconfiguration; on the other hand, a fast trajectory optimization has been obtained by means of a shape-based method and exploiting the Bézier series as shape function. Such algorithms have been tested in two different scenarios of close-range formation flying and the results have been compared with data available from literature. Additionally, a study on how the order of the Bézier series affects the results of trajectory optimization has been performed in order to assess whether it is possible to define an ideal order that allows to obtain a low ΔV with a limited computational time.

Keywords: trajectory optimization, shape-based method, Bézier series, formation reconfiguration

Abstract in lingua italiana

Negli ultimi anni il settore spaziale è stato caratterizzato da un veloce aumento nel numero di satelliti in orbita intorno alla Terra e dalla riduzione della dimensione dei satelliti. Grazie alla disponibilità di satelliti di dimensioni minori, si nota un aumento dell'interesse verso le missioni caratterizzate dall'utilizzo di satelliti in formazione anche grazie alla maggiore adattabilità e affidabilità intrinseca. Inoltre, è appurato che in molte applicazioni di telerilevamento l'utilizzo di satelliti in formazione permette di ottenere prestazioni migliori rispetto a un singolo satellite monolitico. Un'esigenza richiesta dai satelliti in formazione è la capacità di poter effettuare la riconfigurazione della formazione quando necessario.
In questa tesi, è stato sviluppato uno strumento che permette di ottenere in tempi contenuti una strategia per la riconfigurazione di una formazione di satelliti tramite un approccio decentralizzato. Questo lavoro potrebbe apportare dei miglioramenti al settore spaziale in due modi: il primo è l'aumento dell'autonomia dei satelliti rispetto alle stazioni di terra, il secondo è poter utilizzare la soluzione ottenuta come valore iniziale per ottimizzatori più sofisticati. Per poter ottenere questo risultato è stato utilizzato un algoritmo genetico basato sulla variazione dei tempi iniziali di manovra dei satelliti coinvolti per ottenere una strategia per la riconfigurazione della formazione. È stato poi sviluppato in supporto all'algoritmo genetico un ottimizzatore di traiettoria che sfrutta un metodo shape-based e utilizza la serie di Bézier come shape function. Questi algoritmi sono stati testati in due scenari in cui sono presenti dei satelliti in formazione ravvicinata, e i risultati sono stati confrontanti con dati disponibili in letteratura. Inoltre, è stato effettuato uno studio per determinare come l'ordine della serie di Bézier influenzi i risultati dell'ottimizzazione di traiettoria; in particolare, era di interesse determinare se ci fosse un ordine ideale che permetta di ottenere un ΔV basso con un tempo computazionale limitato.

Parole chiave: ottimizzazione di traiettoria, metodo shape-based, serie di Bézier, riconfigurazione di formazione di satelliti

Contents

List of Figures

List of Tables

1 | Introduction

In the last two decades, the space segment has been characterized by two major trends, the first one is the increase in satellites' launch rates while the second one is the decrease in the satellites' dimensions [1].

The increase in satellites' launch rates can be explained by the increased interest in the space segment for telecommunication and navigation systems; indeed, satellites allow to obtain coverage that systems on Earth cannot obtain or can be obtained with higher costs. Additionally, satellites devoted to Earth monitoring, weather, and disasters monitoring also play an important role in the increasing number of satellites orbiting Earth.

The decrease in satellites' dimensions has been possible due to the miniaturization in the involved technologies. The biggest consequence of miniaturized satellites is the higher accessibility to space also for smaller companies and institutions due to the reduced development costs. Additionally, the availability of miniaturized satellites allows to increase the number of satellites that can be deployed during a single launch.

The decrease of satellites' dimensions has made more accessible the strategy of distributed systems in the form of constellations and formation flying. The latter has multiple benefits: in terms of performances of the overall mission's objective, in the increase in the reliability and flexibility, and, as a consequence, in the reduction of the costs [2]. In terms of performances, formation flying allows to obtain better results than a monolithic satellite, since the limitations imposed by the fairing of the launcher can be overcome; as a result, for example, when dealing with remote sensing, better performances in terms of resolution and coverage can be achieved [3, 4]. The increase in reliability with respect to a monolithic satellite is obtained naturally from the exploitation of multiple satellites; indeed, in case of a failure of a satellite it would be possible to replace the faulty unit at a much lower cost compared to the case of a single monolithic satellite. Finally, a higher flexibility in terms of mission's objectives, can be achieved in two ways: the first one is with the concept of modularity, therefore replacing or adding another satellite with a different one having a specific instrument for the given mission's objective [2]. The second, and most common one, is by varying the geometry of the formation, a process often called formation reconfiguration, to achieve a given objective [5]; to do so, the two most important aspects are to minimize the fuel consumption for each satellite during the reconfiguration process and, a mandatory condition, to enforce collision avoidance between all the involved satellites.

The main objective of this work is to develop a tool for formation reconfiguration for satellites flying in close-range; in particular, focus will be on satellites flying within 15 m from the leader satellite. Two cornerstones have been set, the first one is to perform the trajectory optimization by means of a shape-based method, while the second one is to perform the formation reconfiguration by means of a decentralized approach. The reason behind these cornerstones is that, as the number of satellites orbiting Earth increases, their autonomy with respect to ground stations should increase and being able to perform the task of trajectory optimization in an autonomous manner is a step further to reach such a goal. In particular, shape-based methods allow to obtain a feasible solution with a low computational effort, which is mandatory with the currently available computational power of satellites, when dealing with satellites capable to perform maneuvers with continuous thrust. A decentralized approach, instead, allows to spread the computational burden among all the satellites of the formation and increases the reliability of the overall mission, since each satellite is capable of performing its own trajectory optimization. Finally, it is worth noting that, even if in this work focus has been given to formation reconfiguration, shape-based methods could also be applied to trajectory optimization for standalone satellites and the obtained solutions of such methods could also be used as first guess for more refined optimizers.

1.1. Trajectory Optimization by Means of Shape-Based Methods

Trajectory optimization by means of a shape-based method is usually performed as follows: one or more functions, called shape functions, are used to define the trajectory of the satellite; such shape functions are defined by means of coefficients which become the optimization variables. By performing the time derivation of the trajectory, the velocity and the acceleration components of the satellite can also be obtained; after that, the equations of motion of the satellite are solved for the control, the thrust acceleration, which is integrated in time in order to obtain the objective function of the problem, the ΔV. The strength of shape-based methods for the solution of the trajectory optimization problem is that the differential problem, described by the equations of motion of the satellite, is transformed into an algebraic one which can be handled more easily by means of numerical methods.

The number of shape functions depends on the selected reference frame: two shape functions are needed when dealing with cylindrical or polar reference frames, while three are needed when dealing with spherical or 3D Cartesian reference frames.
In literature a wide variety of shape functions have been used for the trajectory optimization problem. In [6] the concept of trajectory optimization by means of a shape function is first introduced for solving interplanetary transfers by means of a polynomial. Later, in [7], the idea is further developed by employing the Chebyshev polynomials for the solution of general nonlinear control problems. In [8] and [9] the trajectory optimization with multiple gravity-assist is solved, respectively, by means of an exponential sinusoid shape function and by a set of parameterized pseudo-equinoctial elements. In [10] the optimization of an interplanetary trajectory is performed exploiting an inverse polynomial for the in-plane motion while a polynomial is used for the out-of-plane one. In [11] the fast Fourier series is used to solve constrained low-thrust trajectories. The strength of such shape function is that, by considering enough terms of the series and by varying the Fourier coefficients, any continuous trajectory can be obtained; additionally, the boundary conditions can be enforced by fixing some of the coefficients of the series. The same authors exploit the same shape function in [12] to solve the restricted three-body problem, while in [13] for three-dimensional low-thrust trajectory design. In [14] the idea of using a series as a shape function is further developed by considering the Bézier series for electric sail trajectory design; with the Bézier series the boundary conditions can be enforced in the same way of the finite Fourier one, but it has the advantage of not requiring trigonometric functions which can be complex to manage in the process of derivation and integration. In [15] the Bézier series is used for trajectory optimization of satellites in close-range formation; in particular, the trajectories of all the satellites of the formation is optimized at the same time with a centralized approach and enforcing collision avoidance. In this work, the Bézier series and the finite Fourier one are compared in terms of computation time and ΔV of the maneuver showing that with the former better results can be obtained in the considered application. In [16] both the finite Fourier series and the Bézier one are used as shape functions for trajectory generation of multispacecraft for multiasteroid exploration; also in this case the Bézier series is capable of obtaining better results with respect to the finite Fourier series for both the computation time and the overall ΔV.

1.2. Trajectory Optimization for Formation Reconfiguration

In this section the available techniques for trajectory optimization for formation reconfiguration are presented.

In [17] the reconfiguration problem has been tackled by means of a direct approach in which the constraints, maximum thrust and equations of motion, have been discretized and the controls parametrized, leading to a parametric optimization problem. Collision avoidance, instead, has been enforced by adding a penalty term in the objective function.

A different approach has been used in [18], in which the characteristics of swarms observed in nature are applied to formation reconfiguration by enforcing four rules. The first two rules allow to keep the satellites close to each other, but above an imposed minimum value, the third rule allows to keep the satellites on a given semi-major axis, and the last rule allows to control the formation by imposing a goal, for example, in terms of minimization of a distance with respect to an objective. Such method, often defined as collective control, has the advantage of being quite robust, since, based on the selected goal, any satellite could be defined as leader, and it can be more fuel efficient as the number of satellites increases. On the other hand, this approach might not be very suitable when performing the assembly around a central satellite and when dealing with complex maneuvers.

In [5] the formation reconfiguration problem is solved by means of disciplined convex programming, a technique in which the optimization problem is made convex in order to assure the existence of a solution. The downside of such method is that the process of making the problem convex is quite challenging and requires the discretization of the involved quantities.

Finally, as already mentioned in Sec. 1.1, in [15] the reconfiguration is performed with a centralized approach by means of a shape-based method enforcing collision avoidance between all the involved satellites.

Building upon these works, the objective of this thesis will be to develop a tool for formation reconfiguration for satellites flying in close-range by means of a shape-based method and a decentralized approach. The thesis is structured as follows: Chapter 2 presents the reference systems as well as the dynamical models, Chapter 3 deals with the trajectory optimization problem, Chapter 4 shows the testing of the developed model as well as an analysis on how the results of the optimization process are affected by the order of the Bézier series. Finally, Chapter 5 deals with the formation reconfiguration problem solving it by means of a decentralized approach.

2 | Reference Frames and Dynamical Models

In this chapter, two main aspects will be covered: the reference frames used for the leader and follower satellites and the dynamical model that has been used to describe the motion of the follower satellites.

2.1. Reference Frames

When dealing with relative motion between satellites, two reference frames are commonly used: the Earth-Centered Inertial (ECI) reference frame and the Local-Vertical-Local-Horizontal (LVHL) reference frame. Indeed, the ECI reference frame is suitable for the description of the overall, large-scale, motion of the leader satellite. On the other hand, the LVLH reference frame, since it is centered in the leader satellite and moves with it, is suitable for the description of the close-range motion of the follower satellite with respect to the leader one.
As illustrated in Figure 2.1, the following notation will be used to represent the two reference frames:

- The unit vectors representing the axis of the ECI reference frame are defined as $\hat{X}$, $\hat{Y}$, and $\hat{Z}$.
- The unit vectors representing the axis of the LVLH reference frame are defined as $\hat{x}$, $\hat{y}$, and $\hat{z}$.

The unit vectors representing the LVLH reference frame are defined as follows:

$$\begin{aligned} \hat{x} &= \frac{\boldsymbol{R_L}}{R_L} \\ \hat{y} &= \hat{z} \times \hat{x} \\ \hat{z} &= \frac{\boldsymbol{H_L}}{H_L} \end{aligned} \tag{2.1}$$

In which $\boldsymbol{R_L}$ and $\boldsymbol{H_L}$ are, respectively, the position vector and the angular momentum vector of the leader satellite in the ECI reference frame.

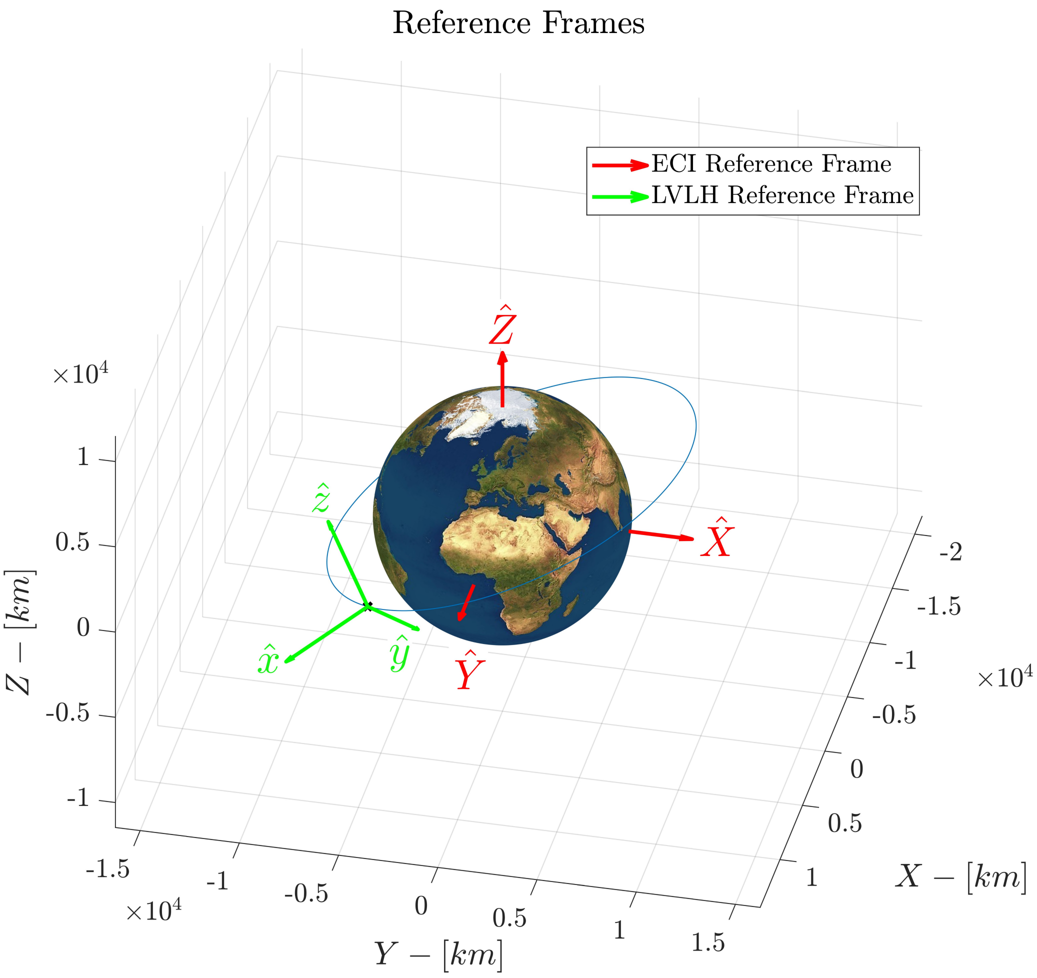

Figure 2.1: Reference Frames.

2.2. Dynamical Models

Before showing the dynamical models for the leader and follower satellites, it is necessary to introduce the notation that will be used:

- The quantities referred to the leader satellite, expressed in the ECI reference frame, will be denoted with a capital letter and the letter L at the subscript; for example, the position vector of the leader satellite will be defined as $\boldsymbol{R_L}$.
- The quantities referred to the follower satellite, expressed in the ECI reference frame, will be denoted with a capital letter and the letter F at the subscript; for example, the position vector of the follower satellite, in the ECI reference frame, will be defined as $\boldsymbol{R_F}$.

- The quantities referred to the follower satellite, expressed in the LVLH reference frame, will be denoted with a lowercase letter; for example, the position vector of the follower satellite, in the LVLH reference frame, will be defined as $\boldsymbol{r}$.

The dynamical model that describes the motion of both leader and follower satellites, in the ECI reference frame, is the following:

$$\ddot{\boldsymbol{R}}_{\boldsymbol{L}} = -\frac{\mu}{R_L^3}\boldsymbol{R}_{\boldsymbol{L}} + \boldsymbol{F}_{\boldsymbol{L}}(t) \tag{2.2}$$

$$\ddot{\boldsymbol{R}}_{\boldsymbol{F}} = -\frac{\mu}{R_F^3}\boldsymbol{R}_{\boldsymbol{F}} + \boldsymbol{F}_{\boldsymbol{F}}(t) \tag{2.3}$$

In which μ is the Earth's gravitational constant while $\boldsymbol{F}_{\boldsymbol{L}}$ and $\boldsymbol{F}_{\boldsymbol{F}}$ are, respectively, the control of the leader and follower satellites.

In this work, two different relative dynamics have been implemented, the Hill-Clohessy-Wiltshire (HCW) equations and the Tschauner-Hempel (TH) ones. The reason behind the selection of the HCW model is its simplicity, since it only requires one parameter of the leader satellite to describe the relative dynamics. On the other hand, the reason behind the selection of the TH model is that it is still a rather simple model but allows to overcome some of the limitations of the HCW one; additional details will be provided in the following sections.

2.2.1. Hill-Clohessy-Wiltshire Model

The Hill-Clohessy-Wiltshire model is obtained by means of a linearization of the relative motion between two satellites; it is used when dealing with circular orbits and when two or more satellites are flying in close proximity. In this model, the dynamics of the follower satellite are described in the LVLH reference frame as follows:

$$\begin{aligned} &\ddot{x} - 2N\dot{y} - 3N^2x = f_x \\ &\ddot{y} + 2N\dot{x} = f_y \\ &\ddot{z} + N^2z = f_z \end{aligned} \tag{2.4}$$

In which:

- f_x, f_y, and f_z are the thrust acceleration components of the control.
- $N = \sqrt{\mu/a_L^3}$ is the average orbital angular velocity of the leader satellite and a_L is its semi-major axis.

It is important to note that this model is suitable when the leader satellite is flying on a circular orbit, when the propagation time is at most one orbital period, and when the satellites are in close formation. It is also worth noting that this model does not include perturbative effects such as Earth's oblateness, atmospheric drag, and solar radiation pressure; the differential effect, between leader and follower satellites, of these perturbations can be considered negligible when the distances between the considered satellites are within tens of meters [19].

2.2.2. Tschauner-Hempel Model

The Tschauner-Hempel model can be considered as an extension of the HCW model. In this model, the dynamics of the follower satellite are described in the LVLH reference frame as follows:

$$\begin{aligned}
&\ddot{x} - 2\omega_z \dot{y} - \left(\omega_z^2 + 2C_1\right) x - \dot{\omega}_z y = f_x \\
&\ddot{y} + 2\omega_z \dot{x} - \left(\omega_z^2 - C_1\right) y + \dot{\omega}_z x = f_y \\
&\ddot{z} + C_1 z = f_z
\end{aligned} \tag{2.5}$$

In which:

- $\omega_z = \dot{\theta}_L = \frac{N(1+e_L \cos\theta_L)^2}{\left(1-e_L^2\right)^{3/2}}$, e_L is the eccentricity of the leader satellite and θ_L is its true anomaly.
- $\dot{\omega}_z = \ddot{\theta}_L = -\frac{2\mu e_L \sin\theta_L}{R_L^3}$.
- $C_1 = \frac{\mu}{R_L^3}$.

The advantage of this model is that it is capable of correctly describing the relative dynamics of two satellites in close formation when the orbit of the leader satellite is not a circular one. Finally, it is important to note that also in this model the differential effects of the perturbations are neglected [19].

3 | Trajectory Optimization Problem

This chapter covers all the aspects related to the optimization problem by means of a shape-based method. This chapter is structured as follows: the first part provides an initial problem definition highlighting the objective function, the boundary conditions, and the nonlinear inequality constraints. The second part introduces the shape function, the Bézier series, from the mathematical standpoint as well as its exploitation in the optimization process. In the third part, the problem will be made dimensionless; this is done for two main reasons: first of all because, due to its mathematical definition, the Bézier series requires a dimensionless time; secondly because, by also making the space dimensionless, the optimization process will be eased, since the involved quantities can have a comparable order of magnitude. The process of making the problem dimensionless involves all the quantities of the optimization problem, therefore the boundary conditions, the inequality constraints, and the objective function. In the fourth part, the optimization process is, once again, presented and re-expressed by adding the information about the dimensionless quantities. Finally, the last part presents the MATLAB implementation.

3.1. Optimization Problem Definition

The minimization problem that must be solved is the following: minimize the ΔV of a satellite that has to change its configuration with respect to a leader satellite. Such satellite is subjected to two constraints:

- The total thrust acceleration must be lower than the maximum available one.
- The distance between the leader and the follower satellites must be higher than a given threshold value.

The mathematical statement of the optimization problem is the following:

$$\begin{aligned}
&\min_{x,y,z} \quad \Delta V \\
&s.t. \quad \begin{cases} f(t) - f_{max} \leq 0 \\ d_{min} - d_{LF}(t) \leq 0 \end{cases}
\end{aligned} \tag{3.1}$$

In which $d_{LF}(t)$ is the distance between the leader and the follower satellites as a function of time.

In the following sections, details about the objective function, the boundary conditions, and the nonlinear inequality constraints will be provided.

3.1.1. Objective Function

As already stated in the previous section, the objective function is the following:

$$\Delta V = \int_0^{ToF} f(t)\, dt \tag{3.2}$$

In which ToF is the time of flight of the maneuver, and $f(t)$ is the total thrust acceleration, the control, of the follower satellite; the latter is computed as follows:

$$f(t) = \sqrt{(f_x(t))^2 + (f_y(t))^2 + (f_z(t))^2} \tag{3.3}$$

In which $f_x(t)$, $f_y(t)$, and $f_z(t)$ are the thrust acceleration components of the follower satellite in the LVLH reference frame.

3.1.2. Boundary Conditions

The boundary conditions of the follower satellite are the initial and the final states which are imposed. Fixing the initial and the final states gives rise to twelve boundary conditions:

$$\begin{aligned}
x(0) &= x_0 & \dot{x}(0) &= \dot{x}_0 \\
y(0) &= y_0 & \dot{y}(0) &= \dot{y}_0 \\
z(0) &= z_0 & \dot{z}(0) &= \dot{z}_0 \\
x\,(ToF) &= x_{ToF} & \dot{x}\,(ToF) &= \dot{x}_{ToF} \\
y\,(ToF) &= y_{ToF} & \dot{y}\,(ToF) &= \dot{y}_{ToF} \\
z\,(ToF) &= z_{ToF} & \dot{z}\,(ToF) &= \dot{z}_{ToF}
\end{aligned} \tag{3.4}$$

In which the subscript 0 refers to the initial conditions while the subscript ToF refers to the final ones.
To make the notation more compact the following will be used:

$$\begin{aligned} \boldsymbol{x}(0) &= \begin{bmatrix} x_0 & y_0 & z_0 & \dot{x}_0 & \dot{y}_0 & \dot{z}_0 \end{bmatrix} \\ \boldsymbol{x}\left(ToF\right) &= \begin{bmatrix} x_{ToF} & y_{ToF} & z_{ToF} & \dot{x}_{ToF} & \dot{y}_{ToF} & \dot{z}_{ToF} \end{bmatrix} \end{aligned} \tag{3.5}$$

3.1.3. Nonlinear Inequality Constraints

The nonlinear inequality constraints of the problem are the following:

- Maximum total thrust acceleration available, f_{max}.
- Minimum distance between leader and follower satellites, d_{min}.

These constraints can be expressed in mathematical form as follows:

$$\begin{aligned} f(t) &= \sqrt{\left(f_x(t)\right)^2 + \left(f_y(t)\right)^2 + \left(f_z(t)\right)^2} \leq f_{max} \\ d_{LF}(t) &= \sqrt{\left(x(t)\right)^2 + \left(y(t)\right)^2 + \left(z(t)\right)^2} \geq d_{min} \end{aligned} \tag{3.6}$$

Note that instead of the constraints expressed above, their squares will be used:

$$\begin{aligned} &\left(f_x(t)\right)^2 + \left(f_y(t)\right)^2 + \left(f_z(t)\right)^2 \leq f_{max}^2 \\ &\left(x(t)\right)^2 + \left(y(t)\right)^2 + \left(z(t)\right)^2 \geq d_{min}^2 \end{aligned} \tag{3.7}$$

The reason behind this choice is to provide to the optimizer a region of feasible solutions broader than the one provided by Eq. (3.6).

3.2. Shape Function

The shape function, the Bézier series, has been used to approximate the position, velocity, and acceleration components of the follower satellite. In the following sections, the Bézier series and its derivatives will be introduced.

3.2.1. Bézier Series and Its Derivatives

The Bézier series is defined as follows [20]:

$$a(\tau) = \sum_{j=0}^{n} B_j(\tau) P_j \tag{3.8}$$

In which n is an integer that represents the order of the Bézier series, P_j are the geometric coefficients, and $B_j(\tau)$ are the Bernstein polynomials:

$$B_j(\tau) = \binom{n}{j} \tau^j (1-\tau)^{n-j} = \frac{n! \tau^j (1-\tau)^{n-j}}{j!(n-j)!} \tag{3.9}$$

It is worth highlighting that the Bernstein polynomials form a complete basis over the interval $[0,1]$ for all polynomials of degree $\leq n$ [20].
The first τ derivative of the Bézier series is the following:

$$a'(\tau) = \sum_{j=0}^{n} B'_j(\tau) P_j \tag{3.10}$$

Note that the geometric coefficients do not depend on τ and remain constant upon derivation.
The first τ derivative of the Bernstein polynomials can be expressed as follows:

$$B'_j(\tau) = \binom{n}{j} \left(j\tau^{j-1}(1-\tau)^{n-j} + \tau^j (j-n)(1-\tau)^{n-j-1} \right) \tag{3.11}$$

The second τ derivative of the Bézier series is the following:

$$a''(\tau) = \sum_{j=0}^{n} B''_j(\tau) P_j \tag{3.12}$$

The second τ derivative of the Bernstein polynomials can be expressed as follows:

$$\begin{aligned} B''_j(\tau) = \binom{n}{j} \Big(& j(j-1)\tau^{j-2}(1-\tau)^{n-j} + 2j\tau^{j-1}(j-n)(1-\tau)^{n-j-1} + ... \\ & ... + \tau^j (j-n)(j-n-1)(1-\tau)^{n-j-2} \Big) \end{aligned} \tag{3.13}$$

One remark, the following notation will be used:

$$a(\tau) = \sum_{j=0}^{n} B_j(\tau)P_j = \begin{bmatrix} B_0(\tau) & ... & B_n(\tau) \end{bmatrix} * \begin{bmatrix} P_0 & ... & P_n \end{bmatrix}^T = \boldsymbol{B}(\tau)\boldsymbol{P}$$
$$a'(\tau) = \sum_{j=0}^{n} B'_j(\tau)P_j = \begin{bmatrix} B'_0(\tau) & ... & B'_n(\tau) \end{bmatrix} * \begin{bmatrix} P_0 & ... & P_n \end{bmatrix}^T = \boldsymbol{B'}(\tau)\boldsymbol{P} \tag{3.14}$$
$$a''(\tau) = \sum_{j=0}^{n} B''_j(\tau)P_j = \begin{bmatrix} B''_0(\tau) & ... & B''_n(\tau) \end{bmatrix} * \begin{bmatrix} P_0 & ... & P_n \end{bmatrix}^T = \boldsymbol{B''}(\tau)\boldsymbol{P}$$

3.2.2. Position, Velocity, and Acceleration Components by Means of the Bézier Series

Due to the fact that the Bézier series forms a complete set over the interval $[0, 1]$, it is necessary to define a scaled time that ranges in that interval as well; to do so, the following scaling can be performed:

$$\tau = \frac{t}{ToF} \tag{3.15}$$

In which t is the time variable that ranges between 0 and ToF; in this way a dimensionless time that ranges in the interval $[0, 1]$ has been defined.

With this scaling, the velocity and the acceleration components are now expressed as follows:

$$\begin{aligned} \dot{x}(t) * ToF = x'(\tau) \quad \ddot{x}(t) * ToF^2 = x''(\tau) \\ \dot{y}(t) * ToF = y'(\tau) \quad \ddot{y}(t) * ToF^2 = y''(\tau) \\ \dot{z}(t) * ToF = z'(\tau) \quad \ddot{z}(t) * ToF^2 = z''(\tau) \end{aligned} \tag{3.16}$$

In which the notation $\cdot$ represents the time derivative while the subscript $'$ represents the τ derivative.

It is now possible to express the position, velocity, and acceleration components by means of the Bézier series; for the x component:

$$x(\tau) = \sum_{j=0}^{n_x} B_j(\tau)P_{j,x} = \boldsymbol{B_x}(\tau)\boldsymbol{P_x}$$
$$x'(\tau) = \sum_{j=0}^{n_x} B'_j(\tau)P_{j,x} = \boldsymbol{B'_x}(\tau)\boldsymbol{P_x} \tag{3.17}$$
$$x''(\tau) = \sum_{j=0}^{n_x} B''_j(\tau)P_{j,x} = \boldsymbol{B''_x}(\tau)\boldsymbol{P_x}$$

For the y component:

$$\begin{aligned}
y(\tau) &= \sum_{j=0}^{n_y} B_j(\tau)P_{j,y} = \boldsymbol{B_y}(\tau)\boldsymbol{P_y} \\
y'(\tau) &= \sum_{j=0}^{n_y} B'_j(\tau)P_{j,y} = \boldsymbol{B'_y}(\tau)\boldsymbol{P_y} \\
y''(\tau) &= \sum_{j=0}^{n_y} B''_j(\tau)P_{j,y} = \boldsymbol{B''_y}(\tau)\boldsymbol{P_y}
\end{aligned} \tag{3.18}$$

For the z component:

$$\begin{aligned}
z(\tau) &= \sum_{j=0}^{n_z} B_j(\tau)P_{j,z} = \boldsymbol{B_z}(\tau)\boldsymbol{P_z} \\
z'(\tau) &= \sum_{j=0}^{n_z} B'_j(\tau)P_{j,z} = \boldsymbol{B'_z}(\tau)\boldsymbol{P_z} \\
z''(\tau) &= \sum_{j=0}^{n_z} B''_j(\tau)P_{j,z} = \boldsymbol{B''_z}(\tau)\boldsymbol{P_z}
\end{aligned} \tag{3.19}$$

Note that the subscripts x, y, and z indicate that each component might have a different order of the Bézier series and that the geometric coefficients are different for each component.

Also note that the Bernstein polynomials are completely defined by the dimensionless time τ; therefore, in order to distinguish a trajectory, and the corresponding velocity and acceleration histories from another, the only possibility is to vary the geometric coefficients, which become the optimization variables of the problem.

3.2.3. Bézier Series and Boundary Conditions

The boundary conditions allow to compute four geometric coefficients for each component; this is possible due to the fact that for the Bernstein polynomials the following is valid:

$$\begin{aligned}
\boldsymbol{B}(0) &= \begin{cases} 1 & j = 0 \\ 0 & j \in [1, n] \end{cases} \qquad
\boldsymbol{B'}(0) = \begin{cases} -n & j = 0 \\ n & j = 1 \\ 0 & j \in [2, n] \end{cases} \\
\boldsymbol{B}(1) &= \begin{cases} 0 & j \in [0, n-1] \\ 1 & j = n \end{cases} \qquad
\boldsymbol{B'}(1) = \begin{cases} 0 & j \in [0, n-2] \\ -n & j = n-1 \\ n & j = n \end{cases}
\end{aligned} \tag{3.20}$$

Therefore:

$$\begin{aligned}
&x(0) = \boldsymbol{B_x}(0)\boldsymbol{P_x} = P_{0,x} && x'(0) = \boldsymbol{B'_x}(0)\boldsymbol{P_x} = -n_x P_{0,x} + n_x P_{1,x} \\
&y(0) = \boldsymbol{B_y}(0)\boldsymbol{P_y} = P_{0,y} && y'(0) = \boldsymbol{B'_y}(0)\boldsymbol{P_y} = -n_y P_{0,y} + n_y P_{1,y} \\
&z(0) = \boldsymbol{B_z}(0)\boldsymbol{P_z} = P_{0,z} && z'(0) = \boldsymbol{B'_z}(0)\boldsymbol{P_z} = -n_z P_{0,z} + n_z P_{1,z} \\
&x(1) = \boldsymbol{B_x}(1)\boldsymbol{P_x} = P_{n_x,x} && x'(1) = \boldsymbol{B'_x}(1)\boldsymbol{P_x} = -n_x P_{n_x-1,x} + n_x P_{n_x,x} \\
&y(1) = \boldsymbol{B_y}(1)\boldsymbol{P_y} = P_{n_y,y} && y'(1) = \boldsymbol{B'_y}(1)\boldsymbol{P_y} = -n_y P_{n_y-1,y} + n_y P_{n_y,y} \\
&z(1) = \boldsymbol{B_z}(1)\boldsymbol{P_z} = P_{n_z,z} && z'(1) = \boldsymbol{B'_z}(1)\boldsymbol{P_z} = -n_z P_{n_z-1,z} + n_z P_{n_z,z}
\end{aligned} \tag{3.21}$$

Rewriting in dimensionless time form the boundary conditions associated with the velocity components expressed in Eq. (3.4), leads to the following:

$$\begin{aligned}
&x'(0) = \dot{x}_0 * ToF && x'(1) = \dot{x}_{ToF} * ToF \\
&y'(0) = \dot{y}_0 * ToF && y'(1) = \dot{y}_{ToF} * ToF \\
&z'(0) = \dot{z}_0 * ToF && z'(1) = \dot{z}_{ToF} * ToF
\end{aligned} \tag{3.22}$$

Combining the boundary conditions of the position components, expressed in Eq. (3.4), the boundary conditions of the velocity components, expressed in the equation above, with Eq. (3.21), leads to the following:

$$\begin{aligned}
&x(0) = P_{0,x} = x_0 && x'(0) = -n_x P_{0,x} + n_x P_{1,x} = \dot{x}_0 * ToF \\
&y(0) = P_{0,y} = y_0 && y'(0) = -n_y P_{0,y} + n_y P_{1,y} = \dot{y}_0 * ToF \\
&z(0) = P_{0,z} = z_0 && z'(0) = -n_z P_{0,z} + n_z P_{1,z} = \dot{z}_0 * ToF \\
&x(1) = P_{n_x,x} = x_{ToF} && x'(1) = -n_x P_{n_x-1,x} + n_x P_{n_x,x} = \dot{x}_{ToF} * ToF \\
&y(1) = P_{n_y,y} = x_{ToF} && y'(1) = -n_y P_{n_y-1,y} + n_y P_{n_y,y} = \dot{y}_{ToF} * ToF \\
&z(1) = P_{n_z,z} = x_{ToF} && z'(1) = -n_z P_{n_z-1,z} + n_z P_{n_z,z} = \dot{z}_{ToF} * ToF
\end{aligned} \tag{3.23}$$

The set of equations above clearly shows that, as already mentioned, for each component four geometric coefficients can be determined by means of the boundary conditions. Also note that, if $n_x = n_y = n_z = 3$, there will be four unknown geometric coefficients for each component that can all be determined by imposing the boundary conditions; in this case there is not an optimization to be performed.

3.3. Dimensionless Quantities

In order to perform the optimization by means of the Bézier series as shape function, it is necessary to make the remaining quantities dimensionless in time; additionally, also the space will be made dimensionless. This process of making the quantities dimensionless is done to ease the optimization; indeed, if all the involved quantities range between 0 and 1, or at least have the same order of magnitude, the optimization process is eased.

3.3.1. Dimensionless Time

The quantities that require to be made dimensionless with respect to the time variable are the following:

- Maximum total thrust acceleration available, f_{max}:

$$f_{max,dim\,time} = f_{max} * ToF^2 \tag{3.24}$$

- If the HCW model is used, the average orbital angular velocity of the leader satellite, N, requires to be made dimensionless in time:

$$N_{dim\,time} = N * ToF \tag{3.25}$$

- If the TH model is used, the following quantities require to be made dimensionless in time:

$$\begin{aligned} \omega_{z,dim\,time} &= \omega_z * ToF \\ \dot{\omega}_{z,dim\,time} &= \dot{\omega}_z * ToF^2 \\ C_{1,dim\,time} &= C_1 * ToF^2 \end{aligned} \tag{3.26}$$

Note that the boundary conditions of the follower satellite do not require to be made dimensionless in time, since this process is already present in the definition of the Bézier series.

3.3.2. Dimensionless Space

In order to make the space dimensionless, it is necessary to define a scaling parameter similar to the time of flight in the case of the dimensionless time; to do so, the following process has been performed:

1. From the boundary conditions of the follower satellite, defined in Eq. (3.4), compute the geometric coefficients as shown in Eq. (3.23).

2. Define the trajectory of the follower satellite by means of a third order Bézier series for all the components.

3. Compute the trajectory of the follower satellite and obtain the maximum, in absolute value, for each component.

4. Use the maximum among the three components as a scaling parameter, namely s_{scal}, for the space variable.

The quantities that require to be made dimensionless with respect to the space variable are the following:

- The boundary conditions of the follower satellite:

$$\begin{aligned} \boldsymbol{x}_{\boldsymbol{dim\ space}}(0) &= \frac{\boldsymbol{x}(0)}{s_{scal}} \\ \boldsymbol{x}_{\boldsymbol{dim\ space}}(ToF) &= \frac{\boldsymbol{x}(ToF)}{s_{scal}} \end{aligned} \tag{3.27}$$

- Maximum total thrust acceleration available, f_{max}:

$$f_{max,dim\ space} = \frac{f_{max}}{s_{scal}} \tag{3.28}$$

- Minimum distance between leader and follower satellites, d_{min}:

$$d_{min,dim\ space} = \frac{d_{min}}{s_{scal}} \tag{3.29}$$

Therefore, by combining the quantities dimensionless in time with those dimensionless in space, it is possible to define the following dimensionless quantities:

- The boundary conditions of the follower satellite:

$$\begin{aligned} \boldsymbol{x}_{\boldsymbol{dim}}(0) &= \frac{\boldsymbol{x}(0)}{s_{scal}} \\ \boldsymbol{x}_{\boldsymbol{dim}}(ToF) &= \frac{\boldsymbol{x}(ToF)}{s_{scal}} \end{aligned} \tag{3.30}$$

- Maximum total thrust acceleration available, f_{max}:

$$f_{max,dim} = \frac{f_{max} * ToF^2}{s_{scal}} \tag{3.31}$$

- Minimum distance between leader and follower satellites, d_{min}:

$$d_{min,dim} = \frac{d_{min}}{s_{scal}} \tag{3.32}$$

- If the HCW model is used:

$$N_{dim} = N * ToF \tag{3.33}$$

- If the TH model is used:

$$\begin{aligned} \omega_{z,dim} &= \omega_z * ToF \\ \dot{\omega}_{z,dim} &= \dot{\omega}_z * ToF^2 \\ C_{1,dim} &= C_1 * ToF^2 \end{aligned} \tag{3.34}$$

In order to make the whole problem dimensionless, also the representation of the position, velocity, and acceleration components by means of the Bézier series must be changed; this is done by computing again the geometric coefficients using as boundary conditions $\boldsymbol{x_{dim}}(0)$ and $\boldsymbol{x_{dim}}(ToF)$:

$$\begin{aligned}
x_{dim}(0) &= P_{0,x_{dim}} = \frac{x_0}{s_{scal}} & x'_{dim}(0) &= -n_x P_{0,x_{dim}} + n_x P_{1,x_{dim}} = \dot{x}_0 * \frac{ToF}{s_{scal}} \\
y_{dim}(0) &= P_{0,y_{dim}} = \frac{y_0}{s_{scal}} & y'_{dim}(0) &= -n_y P_{0,y_{dim}} + n_y P_{1,y_{dim}} = \dot{y}_0 * \frac{ToF}{s_{scal}} \\
z_{dim}(0) &= P_{0,z_{dim}} = \frac{z_0}{s_{scal}} & z'_{dim}(0) &= -n_z P_{0,z_{dim}} + n_z P_{1,z_{dim}} = \dot{z}_0 * \frac{ToF}{s_{scal}} \\
x_{dim}(1) &= P_{n_x,x_{dim}} = \frac{x_{ToF}}{s_{scal}} & x'_{dim}(1) &= -n_x P_{n_x-1,x_{dim}} + n_x P_{n_x,x_{dim}} = \dot{x}_{ToF} * \frac{ToF}{s_{scal}} \\
y_{dim}(1) &= P_{n_y,y_{dim}} = \frac{y_{ToF}}{s_{scal}} & y'_{dim}(1) &= -n_y P_{n_y-1,y_{dim}} + n_y P_{n_y,y_{dim}} = \dot{y}_{ToF} * \frac{ToF}{s_{scal}} \\
z_{dim}(1) &= P_{n_z,z_{dim}} = \frac{z_{ToF}}{s_{scal}} & z'_{dim}(1) &= -n_z P_{n_z-1,z_{dim}} + n_z P_{n_z,z_{dim}} = \dot{z}_{ToF} * \frac{ToF}{s_{scal}}
\end{aligned} \tag{3.35}$$

As a consequence, the position, velocity, and acceleration components by means of the Bézier series become the following:

$$\begin{aligned}
x_{dim}(\tau) &= \boldsymbol{B_x}(\tau)\boldsymbol{P_{x,dim}} & x'_{dim} &= \boldsymbol{B'_x}(\tau)\boldsymbol{P_{x,dim}} & x''_{dim}(\tau) &= \boldsymbol{B''_x}(\tau)\boldsymbol{P_{x,dim}} \\
y_{dim}(\tau) &= \boldsymbol{B_y}(\tau)\boldsymbol{P_{y,dim}} & y'_{dim} &= \boldsymbol{B'_y}(\tau)\boldsymbol{P_{y,dim}} & y''_{dim}(\tau) &= \boldsymbol{B''_y}(\tau)\boldsymbol{P_{y,dim}} \\
z_{dim}(\tau) &= \boldsymbol{B_z}(\tau)\boldsymbol{P_{z,dim}} & z'_{dim} &= \boldsymbol{B'_z}(\tau)\boldsymbol{P_{z,dim}} & z''_{dim}(\tau) &= \boldsymbol{B''_z}(\tau)\boldsymbol{P_{z,dim}}
\end{aligned} \tag{3.36}$$

Finally, also the thrust acceleration components must be expressed in dimensionless form by rewriting the relative dynamics presented in Sec. 2.2. If the HCW model is used the dynamics can be rewritten as follows:

$$\begin{aligned}
f_{x,dim}(\tau) &= f_x(t) * \frac{ToF^2}{s_{scal}} = x''_{dim} - 2N_{dim}y'_{dim} - 3N_{dim}^2 x_{dim} = ... \\
&... = \boldsymbol{B''_x}(\tau)\boldsymbol{P_{x,dim}} - 2N_{dim}\boldsymbol{B'_y}(\tau)\boldsymbol{P_{y,dim}} - 3N_{dim}^2\boldsymbol{B_x}(\tau)\boldsymbol{P_{x,dim}} \\
f_{y,dim}(\tau) &= f_y(t) * \frac{ToF^2}{s_{scal}} = y''_{dim} + 2N_{dim}x'_{dim} = ... \\
&... = \boldsymbol{B''_y}(\tau)\boldsymbol{P_{y,dim}} + 2N_{dim}\boldsymbol{B'_x}(\tau)\boldsymbol{P_{x,dim}} \\
f_{z,dim}(\tau) &= f_z(t) * \frac{ToF^2}{s_{scal}} = z''_{dim} + N_{dim}^2 z_{dim} = ... \\
&... = \boldsymbol{B''_z}(\tau)\boldsymbol{P_{z,dim}} + N_{dim}^2\boldsymbol{B_z}(\tau)\boldsymbol{P_{z,dim}}
\end{aligned} \tag{3.37}$$

If, instead, the TH model is used:

$$\begin{aligned}
f_{x,dim}(\tau) &= x''_{dim} - 2\omega_{z,dim}y'_{dim} - \left(\omega_{z,dim}^2 + 2C_{1,dim}\right)x_{dim} - \dot{\omega}_{z,dim}y_{dim} = ...\\
&... = \boldsymbol{B''_x}(\tau)\boldsymbol{P_{x,dim}} - 2\omega_{z,dim}\boldsymbol{B'_y}(\tau)\boldsymbol{P_{y,dim}} + ...\\
&... - \left(\omega_{z,dim}^2 + 2C_{1,dim}\right)\boldsymbol{B_x}(\tau)\boldsymbol{P_{x,dim}} - \dot{\omega}_{z,dim}\boldsymbol{B_y}(\tau)\boldsymbol{P_{y,dim}}\\
f_{y,dim}(\tau) &= y''_{dim} + 2\omega_{z,dim}x'_{dim} - \left(\omega_{z,dim}^2 - C_{1,dim}\right)y_{dim} + \dot{\omega}_{z,dim}x_{dim} = ...\\
&... = \boldsymbol{B''_y}(\tau)\boldsymbol{P_{y,dim}} + 2\omega_{z,dim}\boldsymbol{B'_x}(\tau)\boldsymbol{P_{x,dim}} + ...\\
&... - \left(\omega_{z,dim}^2 - C_{1,dim}\right)\boldsymbol{B_y}(\tau)\boldsymbol{P_{y,dim}} + \dot{\omega}_{z,dim}\boldsymbol{B_x}(\tau)\boldsymbol{P_{x,dim}}\\
f_{z,dim}(\tau) &= z''_{dim} + C_{1,dim}z_{dim} = \boldsymbol{B''_z}(\tau)\boldsymbol{P_{z,dim}} + C_{1,dim}\boldsymbol{B_z}(\tau)\boldsymbol{P_{z,dim}}
\end{aligned} \tag{3.38}$$

One final remark: in order to make the notation less complex, in the following sections the term "dim" at the subscript will be dropped.

3.4. Optimization Problem by Means of a Shape-Based Method

After having introduced the shape function as well as the dimensionless quantities, it is now possible to define the optimization problem with additional details.
The statement of the optimization problem is the following:

$$\begin{aligned}
&\min_{\boldsymbol{P_{opt\ var,x}},\boldsymbol{P_{opt\ var,y}},\boldsymbol{P_{opt\ var,z}}} && \int_0^1 f(\tau)\,d\tau = \int_0^1 \sqrt{(f_x(\tau))^2 + (f_y(\tau))^2 + (f_z(\tau))^2}\,d\tau\\
&s.t. && \begin{cases} (f_x(\tau))^2 + (f_y(\tau))^2 + (f_z(\tau))^2 - f_{max}^2 \leq 0\\ d_{min}^2 - \left((x(\tau))^2 + (y(\tau))^2 + (z(\tau))^2\right) \leq 0 \end{cases}
\end{aligned} \tag{3.39}$$

In which $\boldsymbol{P_{opt\ var,x}}$, $\boldsymbol{P_{opt\ var,y}}$, and $\boldsymbol{P_{opt\ var,z}}$ are the optimization variables while the thrust acceleration components are computed by means of either Eq. (3.37) or Eq. (3.38).
Note that in the objective function the term "ΔV" has been removed due to the fact that, after the process of making the quantities dimensionless, the computed quantity loses the dimensions of a velocity.

3.4.1. Optimization Variables

The optimization variables of the problem are the unknown geometric coefficients of the Bézier series. The number of optimization variables can be computed as follows:

- Number of geometric coefficients: $n_x + 1 + n_y + 1 + n_z + 1 = n_x + n_y + n_z + 3$

- Number of geometric coefficients computed by imposing the boundary conditions: $4 + 4 + 4 = 12$
- Number of optimization variables: $n_x + n_y + n_z + 3 - 12 = n_x + n_y + n_z - 9$

Due to the fact that the boundary conditions allow to compute the first two and the last two geometric coefficients of each component, it is possible to define the following vectors:

$$\begin{aligned}
\boldsymbol{P_x} &= \begin{bmatrix} P_{0,x} & P_{1,x} & \boldsymbol{P_{opt\ var,x}} & P_{n_x-1,x} & P_{n_x,x} \end{bmatrix} \\
\boldsymbol{P_y} &= \begin{bmatrix} P_{0,y} & P_{1,y} & \boldsymbol{P_{opt\ var,y}} & P_{n_y-1,y} & P_{n_y,y} \end{bmatrix} \\
\boldsymbol{P_z} &= \begin{bmatrix} P_{0,z} & P_{1,z} & \boldsymbol{P_{opt\ var,z}} & P_{n_z-1,z} & P_{n_z,z} \end{bmatrix}
\end{aligned} \tag{3.40}$$

This means that the unknown vector of geometric coefficients for each component is the following:

$$\begin{aligned}
\boldsymbol{P_{opt\ var,x}} &= \begin{bmatrix} P_{2,x} & \dots & P_{n_x-2,x} \end{bmatrix} = [\boldsymbol{P_{opt\ var,x}}]_{1\times n_x-3} \\
\boldsymbol{P_{opt\ var,y}} &= \begin{bmatrix} P_{2,y} & \dots & P_{n_y-2,y} \end{bmatrix} = [\boldsymbol{P_{opt\ var,y}}]_{1\times n_y-3} \\
\boldsymbol{P_{opt\ var,z}} &= \begin{bmatrix} P_{2,z} & \dots & P_{n_z-2,z} \end{bmatrix} = [\boldsymbol{P_{opt\ var,z}}]_{1\times n_z-3}
\end{aligned} \tag{3.41}$$

Note that the time of flight is not an optimization variable. There are two main reasons for this choice. The first one is due to the fact that the maneuver that minimizes the ΔV is obtained when the time of flight tends to infinity; since a maneuver is desirable to be completed in a finite time, a constraint on the maximum time of flight available for the maneuver would be required:

$$ToF \leq ToF_{max} \tag{3.42}$$

Also in this case the maneuver that minimizes the ΔV is obtained when the time of flight is close to the maximum one.

The second reason is that having the time of flight as an optimization variable would make the problem much more complex from the implementation point of view and would lead to much higher computation times.

3.4.2. Initial Guess

The process for the computation of the initial guess is the following:

1. Compute the geometric coefficients exploiting the initial and final states of the follower satellite, as shown in Eq. (3.23).
2. Define the trajectory of the follower satellite by means of a third order Bézier series.

3. Compute the space scaling parameter, s_{scal}, and the dimensionless boundary conditions, as already explained in Sec. 3.3.2.
4. Compute the new geometric coefficients with the dimensionless boundary conditions using Eq. (3.35).
5. Define the trajectory of the follower satellite by means of a third order Bézier series.
6. Compute the geometric coefficients for each component and for the desired order of the Bézier series, that could be different for each coordinate, by means of data fitting from the trajectory obtained at the previous step.

3.5. MATLAB Implementation

The overall MATLAB implementation of the proposed optimization, outlined in Figure 3.1, is the following:

1. Define the initial data for the leader and follower satellites, the time of flight, the maximum thrust acceleration available, the minimum distance between the two satellites, and the order of the Bézier series.
2. Based on the relative dynamics model, HCW or TH, compute the quantities that define the motion of the leader satellite as presented in Sec. 2.2.
3. Make the involved quantities dimensionless.
4. Based on the order of the Bézier series, define the Bernstein polynomials and its derivatives.
5. Compute the geometric coefficients based on the dimensionless boundary conditions.
6. Compute a scaling parameter for the objective function.
7. Compute the initial guess for the optimization process.
8. Perform the optimization.

Each block of the algorithm will be discussed in the following sections.

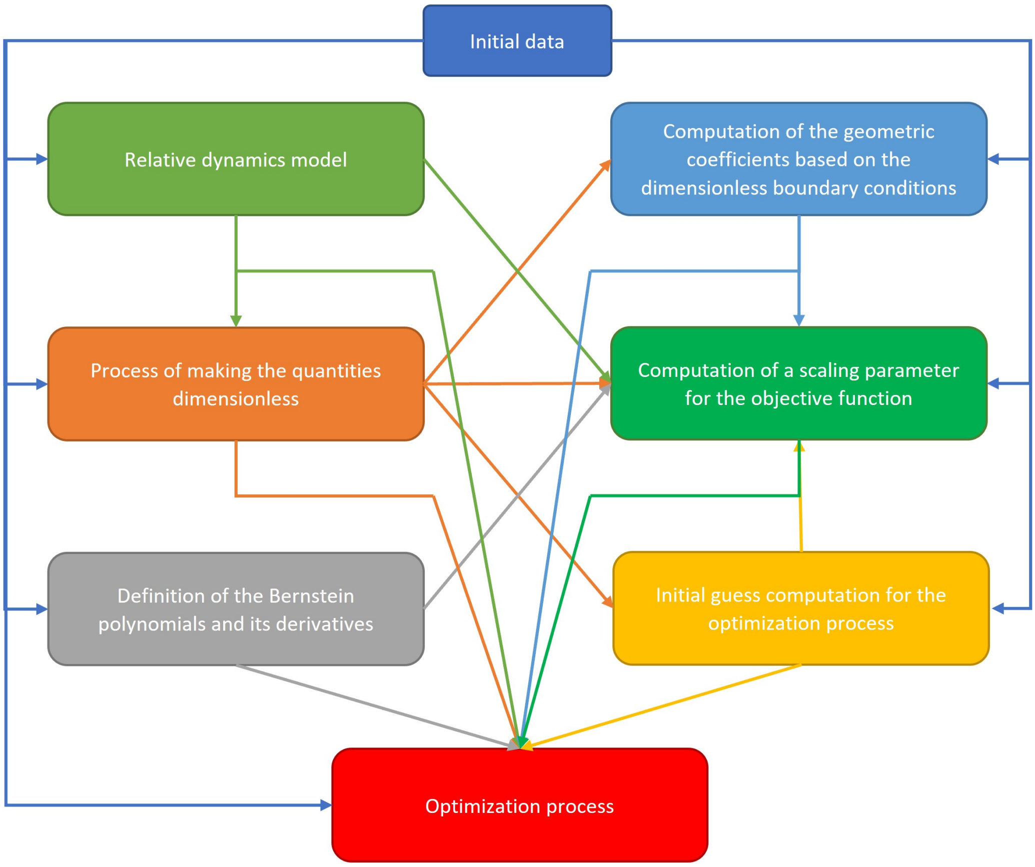
Initial data
Relative dynamics model
Computation of the geometric coefficients based on the dimensionless boundary conditions
Process of making the quantities dimensionless
Computation of a scaling parameter for the objective function
Definition of the Bernstein polynomials and its derivatives
Initial guess computation for the optimization process
Optimization process

Figure 3.1: MATLAB's Workflow.

3.5.1. Initial Data

The initial data that are required for the optimization process are the following:

- The initial conditions of the leader satellite, in particular:
  - If the HCW model is used, the semi-major axis, a_L, of the leader satellite is required.
  - If the TH model is used, the following information are needed:
    - Semi-major axis, a_L
    - Eccentricity, e_L
    - Inclination, i_L
    - Right ascension of the ascending node, Ω_L
    - Argument of the perigee, ω_L
    - The true anomaly at $t = 0$, $\theta_{L,0}$

- The time of flight, ToF
- The boundary conditions of the follower satellite, $\boldsymbol{x}(0)$ and $\boldsymbol{x}(ToF)$
- The maximum total thrust acceleration available, f_{max}
- The minimum distance between leader and follower satellites, d_{min}
- The order of the Bézier series for each component, $\boldsymbol{n} = [n_x \quad n_y \quad n_z]$
- The number of discretization points, n_{points}, which allows to define:
  - A discretized dimensionless time vector: $\boldsymbol{\tau} = [0 \quad ... \quad 1]_{1\times n_{points}}$
  - A discretized time vector: $\boldsymbol{t} = \frac{\boldsymbol{\tau}}{ToF}$

Note that the points of the dimensionless time vector, and therefore the ones of the time vector as well, are not uniformly distributed, but rather they are the roots of the Legendre polynomial of degree equal to n_{points} (Legendre-Gauss distribution of discretization points), as in [15].
Also note that this discretization is kept for all the quantities that require a discretization.

3.5.2. Relative Dynamics Model

In this part of the algorithm some of the coefficients, needed for the computation of the thrust acceleration components, are evaluated. As already presented in Sec. 2.2, two different models have been implemented: the HCW model and the TH one.
If the HCW model is used, the only parameter that needs to be computed is the average orbital angular velocity, N.
If the TH model is used, the following steps are needed:

1. From the Keplerian elements of the leader satellite compute its initial state:

$$\boldsymbol{X_L}(0) = \begin{bmatrix} X_{L,0} & Y_{L,0} & Z_{L,0} & \dot{X}_{L,0} & \dot{Y}_{L,0} & \dot{Z}_{L,0} \end{bmatrix} = [\boldsymbol{X_L}(0)]_{1\times 6} \tag{3.43}$$

2. Perform the orbit propagation of the leader satellite in the interval $\boldsymbol{t}$ in order to obtain a matrix having in each row the state of the leader satellite at a given time:

$$\boldsymbol{X_L} = \begin{bmatrix} X_{L,0} & Y_{L,0} & Z_{L,0} & \dot{X}_{L,0} & \dot{Y}_{L,0} & \dot{Z}_{L,0} \\ & & & \vdots & & \\ X_{L,ToF} & Y_{L,ToF} & Z_{L,ToF} & \dot{X}_{L,ToF} & \dot{Y}_{L,ToF} & \dot{Z}_{L,ToF} \end{bmatrix} = [\boldsymbol{X_L}]_{n_{points}\times 6} \tag{3.44}$$

3. For each row of $\boldsymbol{X_L}$, compute the true anomaly, θ_L, and the position, R_L.
4. Compute ω_z, $\dot{\omega}_z$, and C_1, as shown in Sec. 2.2.2, for each time instant prescribed by $\boldsymbol{t}$.

3.5.3. Definition of the Bernstein Polynomials and its Derivatives

The mathematical framework of the Bernstein polynomials and its derivatives has already been presented in Sec. 3.2.1.

Concerning the MATLAB implementation, it is worth to highlight that the Bernstein polynomials and its derivatives are computed for each component only once in order to save computational time. To do so, the order of the Bézier series, $\boldsymbol{n}$, and the dimensionless time vector, $\boldsymbol{\tau}$, are needed.

3.5.4. Computation of the Geometric Coefficients Based on the Boundary Conditions

The geometric coefficients based on the boundary conditions are computed twice, once before the process of making the quantities dimensionless, and the second time after this process.

The computation of the geometric coefficients based on the boundary conditions has already been addressed in Sec. 3.2.3 and in Sec. 3.3.2, for the dimensionless case; in particular, such coefficients are computed by means of Eq. (3.23), when computed the first time, and by means of Eq. (3.35), when computed with the dimensionless boundary conditions.

3.5.5. Process of Making the Quantities Dimensionless

The process used in order to make the quantities dimensionless has already been explained in detail in Sec. 3.3; Figure 3.2 shows the workflow of the process.

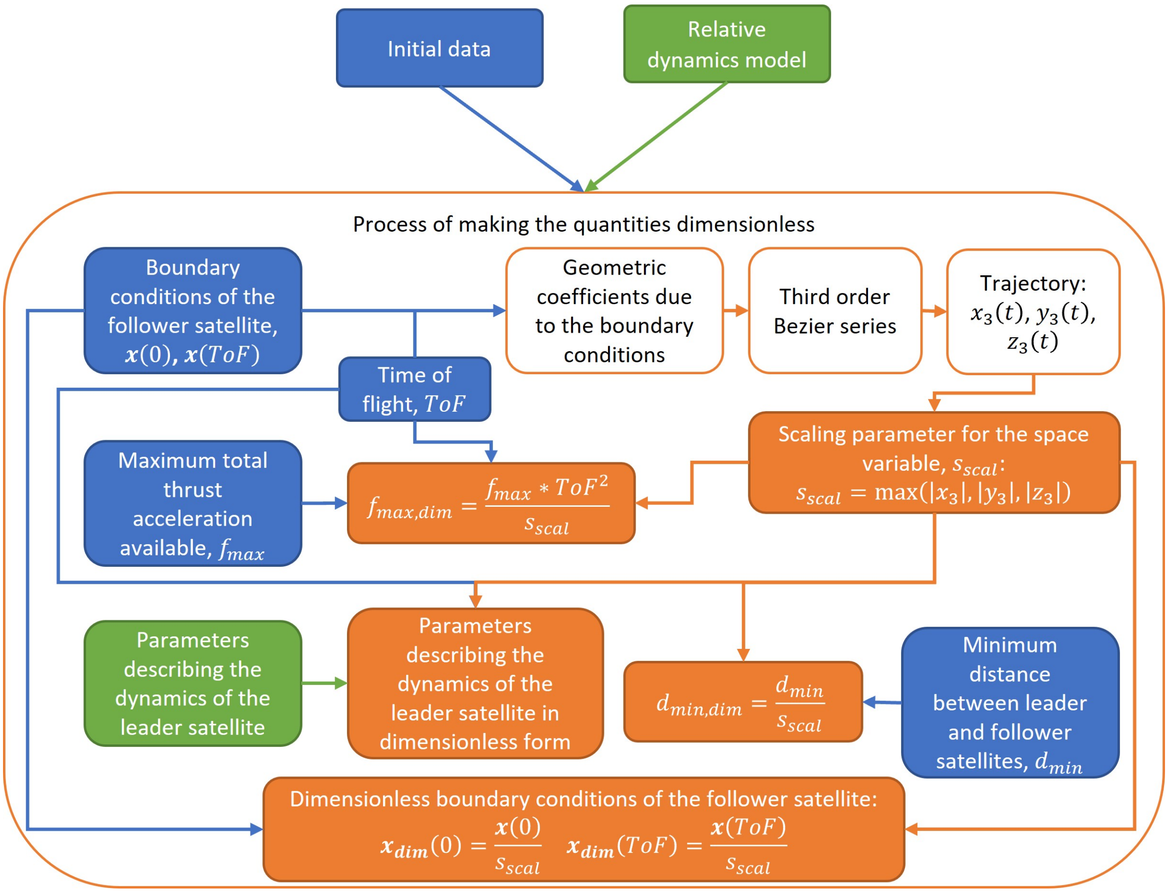
Initial data
Relative dynamics model
Process of making the quantities dimensionless
Boundary conditions of the follower satellite, $\boldsymbol{x}(0), \boldsymbol{x}(ToF)$
Geometric coefficients due to the boundary conditions
Third order Bezier series
Trajectory: $x_3(t), y_3(t), z_3(t)$
Time of flight, ToF
Scaling parameter for the space variable, s_{scal}: $s_{scal} = \max(|x_3|, |y_3|, |z_3|)$
Maximum total thrust acceleration available, f_{max}
$f_{max,dim} = \frac{f_{max} * ToF^2}{s_{scal}}$
Parameters describing the dynamics of the leader satellite
Parameters describing the dynamics of the leader satellite in dimensionless form
$d_{min,dim} = \frac{d_{min}}{s_{scal}}$
Minimum distance between leader and follower satellites, d_{min}
Dimensionless boundary conditions of the follower satellite:
$\boldsymbol{x}_{dim}(0) = \frac{\boldsymbol{x}(0)}{s_{scal}}$ $\boldsymbol{x}_{dim}(ToF) = \frac{\boldsymbol{x}(ToF)}{s_{scal}}$

Figure 3.2: Workflow of the Process of Making the Quantities Dimensionless.

3.5.6. Initial Guess Computation

The initial guess computation has already been addressed in Sec. 3.4.2; in particular, the operations that are performed are the following:

1. With the geometric coefficients computed in Sec. 3.5.4, define the trajectory by means of a third order Bézier series for all the components.

2. Compute the geometric coefficients for each component and for the desired order of the Bézier series, that could be different for each coordinate, by means of data fitting from the trajectory computed at the previous step.

Note that for the computation of such geometric coefficients the MATLAB's built-in $lsqcurvefit$ function has been exploited.

3.5.7. Computation of a Scaling Parameter for the Objective Function

In this section a scaling parameter related to the objective function is presented. The reasons for the introduction of such parameter are mainly two:

- Due to the fact that the objective function is the result of an integration, numerical errors might arise if the involved quantities are too small or too big; in order to avoid such issues, the thrust acceleration is multiplied by a value, the scaling parameter of the objective function, before the integration.
- One of the goals of the process of making the quantities dimensionless was to have values ranging between 0 and 1 during the optimization; in order to achieve this also for the objective function a scaling parameter is introduced.

The process that allows the computation of the scaling parameter, outlined in Figure 3.3, is the following:

1. Compute the position, velocity, and acceleration components of the follower satellite by means of the Bézier series and its derivatives. Note that four geometric coefficients for each component are obtained from the boundary conditions in dimensionless form while the remaining ones are the ones computed in Sec. 3.5.6.
2. Compute the thrust acceleration components, as already shown in Eq. (3.37) or Eq. (3.38), and the total thrust acceleration.
3. Compute the definite integral, between 0 and 1, of the total thrust acceleration; note that this is the objective function as defined in Eq. (3.39).
4. Compute the scaling of the objective function as follows:

$$obj\_fun\_scaling = \frac{0.5}{obj\_fun} \tag{3.45}$$

The reason for this choice is to have the objective function equal to 0.5 at the first iteration of the optimization process and between 0 and 1, or at least in an order of magnitude close to this range, in the following iterations.

Three remarks:

- All the involved quantities are in dimensionless form.
- The position, velocity, and acceleration components, the thrust acceleration components, and the total thrust acceleration are computed at discrete times based on the dimensionless time vector, $\boldsymbol{\tau}$.
- The integral is computed by means of the MATLAB's built-in *trapz* function.

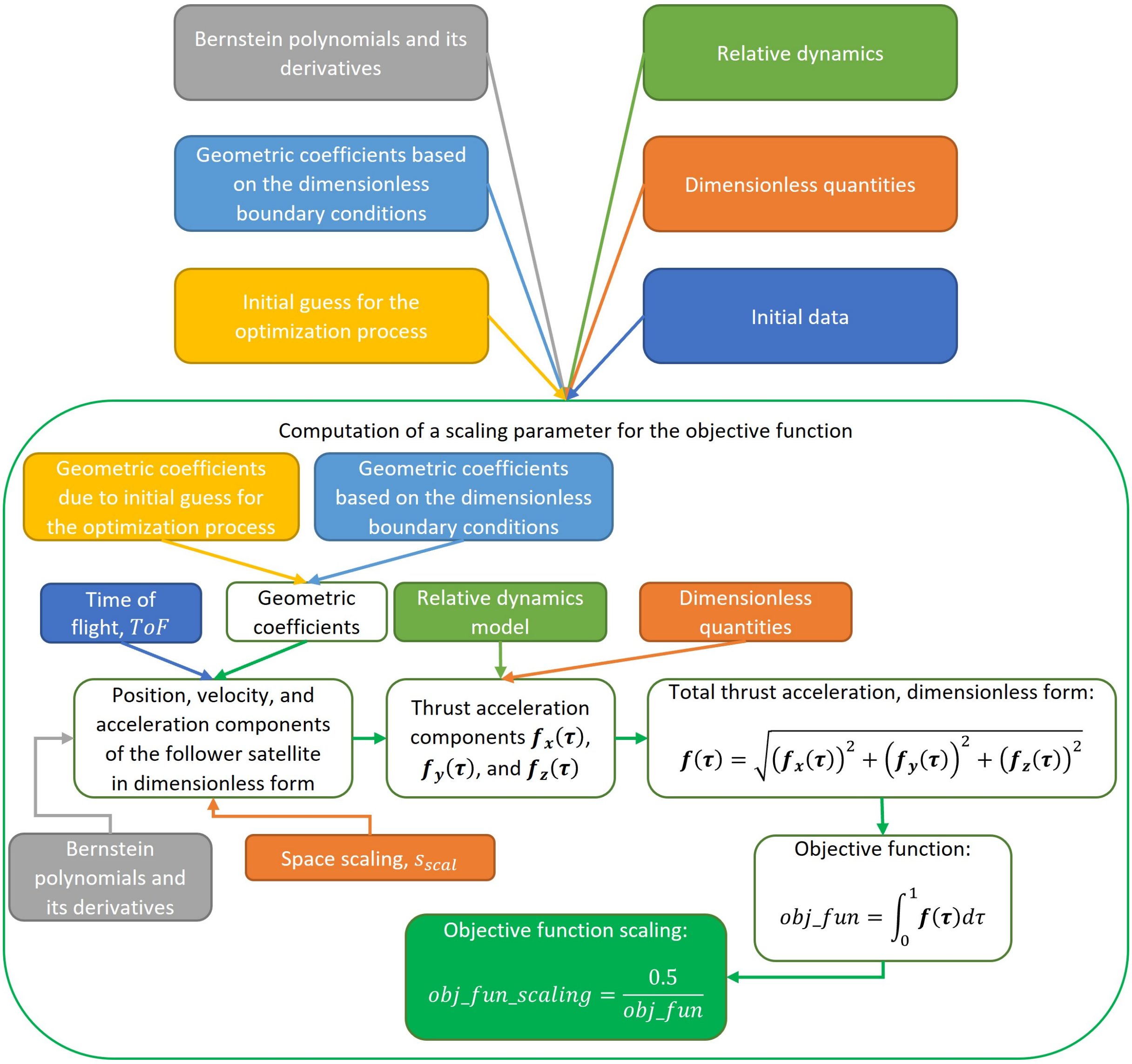
Bernstein polynomials and its derivatives
Relative dynamics
Geometric coefficients based on the dimensionless boundary conditions
Dimensionless quantities
Initial guess for the optimization process
Initial data
Computation of a scaling parameter for the objective function
Geometric coefficients due to initial guess for the optimization process
Geometric coefficients based on the dimensionless boundary conditions
Time of flight, ToF
Geometric coefficients
Relative dynamics model
Dimensionless quantities
Position, velocity, and acceleration components of the follower satellite in dimensionless form
Thrust acceleration components $\boldsymbol{f_x}(\boldsymbol{\tau})$, $\boldsymbol{f_y}(\boldsymbol{\tau})$, and $\boldsymbol{f_z}(\boldsymbol{\tau})$
Total thrust acceleration, dimensionless form:
$\boldsymbol{f}(\boldsymbol{\tau}) = \sqrt{(\boldsymbol{f_x}(\boldsymbol{\tau}))^2 + (\boldsymbol{f_y}(\boldsymbol{\tau}))^2 + (\boldsymbol{f_z}(\boldsymbol{\tau}))^2}$
Bernstein polynomials and its derivatives
Space scaling, s_{scal}
Objective function:
$obj\_fun = \int_0^1 \boldsymbol{f}(\boldsymbol{\tau})d\tau$
Objective function scaling:
$obj\_fun\_scaling = \frac{0.5}{obj\_fun}$

Figure 3.3: Workflow of the Computation of a Scaling Parameter for the Objective Function.

3.5.8. Optimization Process

The optimization has been performed by means of the MATLAB's built-in *fmincon* function; this function, for the problem at hand, requires the following:

- A function that computes the objective function.
- A function that computes the nonlinear inequality constraints.
- An initial guess.

The process for the computation of the objective function, outlined in Figure 3.4, is similar to the one presented in Sec. 3.5.7:

1. Compute the position, velocity, and acceleration components of the follower satellite by means of the Bézier series and its derivatives.
2. Compute the thrust acceleration components and the total thrust acceleration.
3. Compute the objective function as:

$$obj\_fun = \int_0^1 obj\_fun\_scaling * \boldsymbol{f}(\boldsymbol{\tau})\, d\tau \tag{3.46}$$

The process for the computation of the nonlinear inequality constraints, outlined in Figure 3.5, is the following:

1. Compute the position, velocity, and acceleration components of the follower satellite by means of the Bézier series and its derivatives.
2. Compute the thrust acceleration components and the total thrust acceleration.
3. Compute the nonlinear inequality constraints as:

$$\begin{aligned} \left(\boldsymbol{f}\left(\boldsymbol{\tau}\right)\right)^2 - f_{max}^2 \leq \mathbf{0} \\ d_{min}^2 - \left(\boldsymbol{d}\left(\boldsymbol{\tau}\right)\right)^2 \leq \mathbf{0} \end{aligned} \tag{3.47}$$

Note that the quantities are all in dimensionless form.

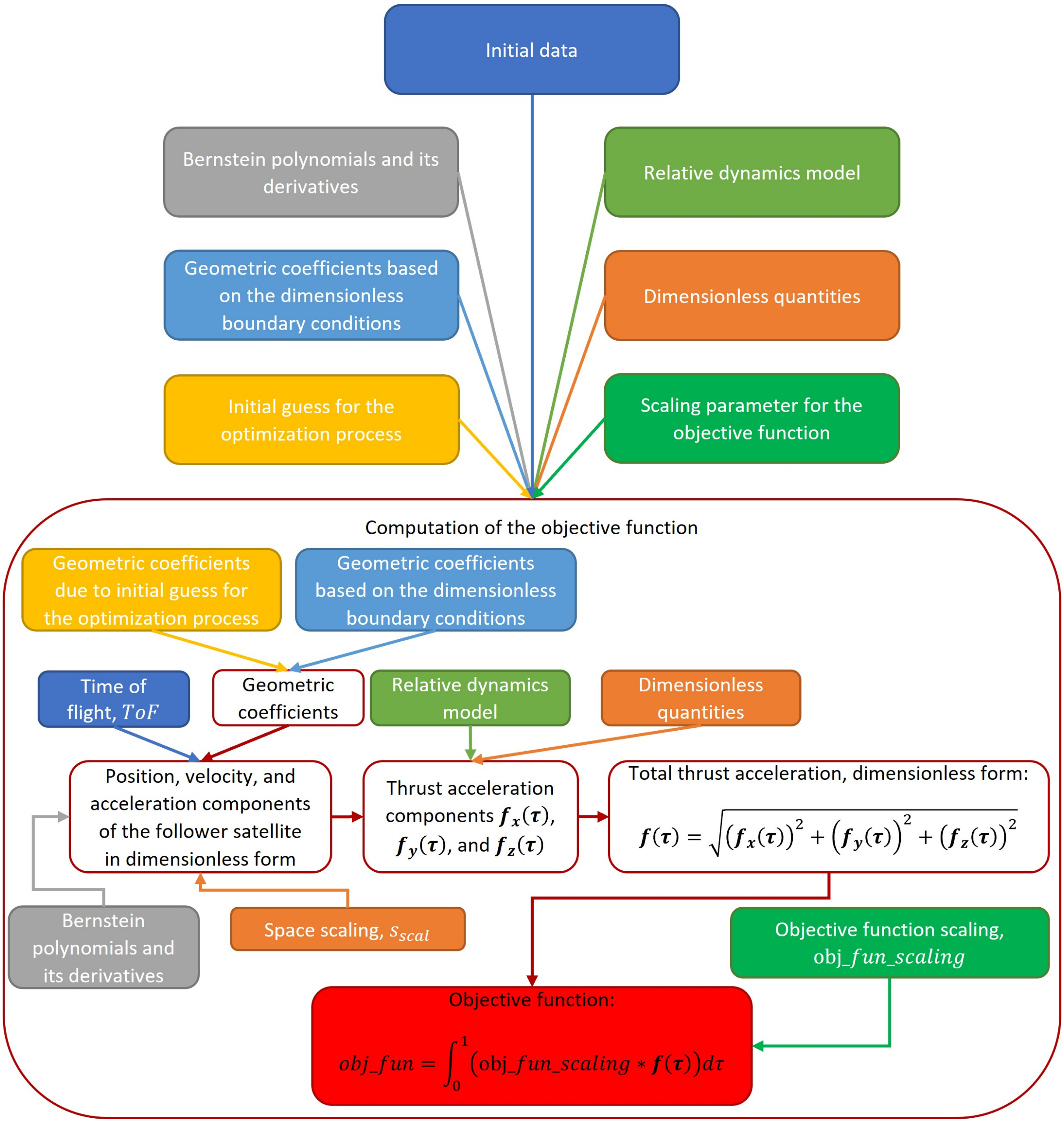
Initial data
Bernstein polynomials and its derivatives
Relative dynamics model
Geometric coefficients based on the dimensionless boundary conditions
Dimensionless quantities
Initial guess for the optimization process
Scaling parameter for the objective function
Computation of the objective function
Geometric coefficients due to initial guess for the optimization process
Geometric coefficients based on the dimensionless boundary conditions
Time of flight, ToF
Geometric coefficients
Relative dynamics model
Dimensionless quantities
Position, velocity, and acceleration components of the follower satellite in dimensionless form
Thrust acceleration components $\boldsymbol{f_x}(\boldsymbol{\tau})$, $\boldsymbol{f_y}(\boldsymbol{\tau})$, and $\boldsymbol{f_z}(\boldsymbol{\tau})$
Total thrust acceleration, dimensionless form:
$\boldsymbol{f}(\boldsymbol{\tau}) = \sqrt{(\boldsymbol{f_x}(\boldsymbol{\tau}))^2 + (\boldsymbol{f_y}(\boldsymbol{\tau}))^2 + (\boldsymbol{f_z}(\boldsymbol{\tau}))^2}$
Bernstein polynomials and its derivatives
Space scaling, s_{scal}
Objective function scaling, obj_$fun_scaling$
Objective function:
$obj\_fun = \int_0^1 (\text{obj}\_fun\_scaling * \boldsymbol{f}(\boldsymbol{\tau}))d\tau$

Figure 3.4: Workflow of the Computation of the Objective Function.

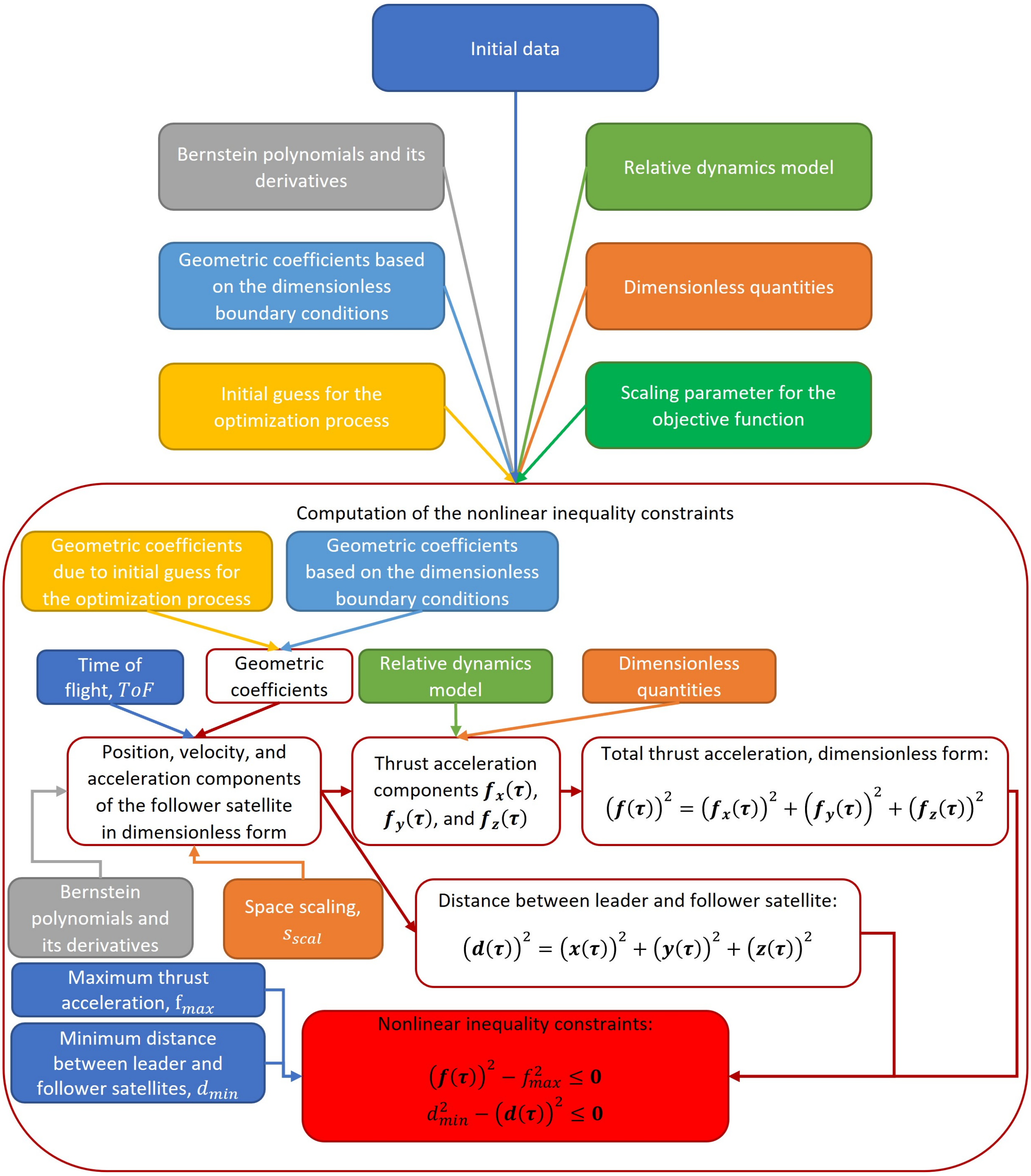
Initial data
Bernstein polynomials and its derivatives
Relative dynamics model
Geometric coefficients based on the dimensionless boundary conditions
Dimensionless quantities
Initial guess for the optimization process
Scaling parameter for the objective function
Computation of the nonlinear inequality constraints
Geometric coefficients due to initial guess for the optimization process
Geometric coefficients based on the dimensionless boundary conditions
Time of flight, ToF
Geometric coefficients
Relative dynamics model
Dimensionless quantities
Position, velocity, and acceleration components of the follower satellite in dimensionless form
Thrust acceleration components $\boldsymbol{f}_{\boldsymbol{x}}(\boldsymbol{\tau})$, $\boldsymbol{f}_{\boldsymbol{y}}(\boldsymbol{\tau})$, and $\boldsymbol{f}_{\boldsymbol{z}}(\boldsymbol{\tau})$
Total thrust acceleration, dimensionless form:
$(\boldsymbol{f}(\boldsymbol{\tau}))^2 = (\boldsymbol{f}_{\boldsymbol{x}}(\boldsymbol{\tau}))^2 + (\boldsymbol{f}_{\boldsymbol{y}}(\boldsymbol{\tau}))^2 + (\boldsymbol{f}_{\boldsymbol{z}}(\boldsymbol{\tau}))^2$
Bernstein polynomials and its derivatives
Space scaling, s_{scal}
Distance between leader and follower satellite:
$(\boldsymbol{d}(\boldsymbol{\tau}))^2 = (\boldsymbol{x}(\boldsymbol{\tau}))^2 + (\boldsymbol{y}(\boldsymbol{\tau}))^2 + (\boldsymbol{z}(\boldsymbol{\tau}))^2$
Maximum thrust acceleration, f_{max}
Minimum distance between leader and follower satellites, d_{min}
Nonlinear inequality constraints:
$(\boldsymbol{f}(\boldsymbol{\tau}))^2 - f_{max}^2 \leq \mathbf{0}$
$d_{min}^2 - (\boldsymbol{d}(\boldsymbol{\tau}))^2 \leq \mathbf{0}$

Figure 3.5: Workflow of the Computation of the Nonlinear Inequality Constraints.

4 | Testing of the Optimizer and a Study on the Bézier Series

This chapter presents the testing of the developed model and an analysis on how the results of the optimization process are affected by the order of the Bézier series.

4.1. Testing of the Developed Models

The developed models and code have been tested by comparing the results of the optimization with data available from literature [15].
The considered scenario as well as the optimization parameters are the following:

- The leader satellite is on a geostationary orbit.
- In literature, the dynamical model for the follower satellites is the HCW one.
- The minimum distance between the leader and the follower satellite is 1.6 m.
- The maximum thrust acceleration is 6×10^{-3} m/s^2.
- The time of flight is 300 s.
- The order of the Bézier series for each coordinate is $\boldsymbol{n} = [12 \quad 12 \quad 16]$.

Six different set of boundary conditions for the follower satellite have been considered and are reported in Table 4.1.
There are three differences between the implemented code and the one found in literature:

- In the implemented code 100 discretization points have been considered while in literature this number is equal to 120.
- In the implemented code the time of flight is not an optimization variable while in the literature implementation it is.
- In the implemented code each case is considered independently while in literature the first three cases are associated to three follower satellites that have to perform

the reconfiguration maneuver simultaneously; the same is done for the last three cases. Note that, for this reason in literature an additional condition, in the form of nonlinear inequality constraint, about the minimum distance between follower satellites is implemented.

	Initial Conditions - [m, m/s]	Final Conditions - [m, m/s]
Case 1	$[0 \quad 2 \quad 0 \quad 0 \quad 0 \quad 0]$	$[0 \quad -4 \quad 0 \quad 0 \quad 0 \quad 0]$
Case 2	$[0 \quad 4 \quad 0 \quad 0 \quad 0 \quad 0]$	$[0 \quad -2 \quad 0 \quad 0 \quad 0 \quad 0]$
Case 3	$[0 \quad 2 \quad 3 \quad 0 \quad 0 \quad 0]$	$[0 \quad -2 \quad -3 \quad 0 \quad 0 \quad 0]$
Case 4	$[2 \quad -3 \quad 0 \quad 0 \quad 0 \quad 0]$	$[-2 \quad 2 \quad 0 \quad 0 \quad 0 \quad 0]$
Case 5	$[-1.5 \quad -3 \quad 3 \quad 0 \quad 0 \quad 0]$	$[1 \quad 2 \quad -1.5 \quad 0 \quad 0 \quad 0]$
Case 6	$[-1.5 \quad -3 \quad -3 \quad 0 \quad 0 \quad 0]$	$[1 \quad 2 \quad 1.5 \quad 0 \quad 0 \quad 0]$

Table 4.1: Boundary Conditions for the Test Cases.

The ΔV obtained after the optimization process for each case and for both the HCW and the TH model is reported in Table 4.2.

It can be noted that the obtained results for both the HCW and TH model are equal or better than the ones obtained in literature. It is worth noting that the results obtained in literature could be higher than the ones obtained by means of the developed code due to the fact that in literature three satellites are being considered at the same time and a minimum distance between the follower satellites is enforced.

Method	Case 1	Case 2	Case 3	Case 4	Case 5	Case 6
	[m/s]	[m/s]	[m/s]	[m/s]	[m/s]	[m/s]
Literature	0.081	0.081	0.083	0.076	0.079	0.083
HCW Model	0.080	0.080	0.082	0.076	0.078	0.078
TH Model	0.080	0.080	0.082	0.076	0.078	0.078

Table 4.2: ΔV for the Different Cases with the Different Methods.

The computation time is reported in Table 4.3. It is worth noting that all the tests were carried out on a i7-4720HQ 2.60 GHz processor, 16.0 GB RAM, with Windows 10 and run on MATLAB R2022b.

It can be noted that the computation time is quite low in all the considered cases, with the TH model being slightly faster than the HCW one. It is also worth noting that the computation times are in agreement with the ones found in literature [15] in which for the first three cases a computation time of 138.848 s is obtained, while for the last three, a value of 233.338 s is obtained.

Method	**Case 1**	**Case 2**	**Case 3**	**Case 4**	**Case 5**	**Case 6**
	[s]	**[s]**	**[s]**	**[s]**	**[s]**	**[s]**
HCW Model	26.579	25.451	29.364	17.189	15.712	13.204
TH Model	25.962	21.751	27.047	14.999	13.605	16.333

Table 4.3: Computation Time for the Different Cases with the Different Methods.

Figure 4.1 and Figure 4.2 show the thrust acceleration history and the projection of the trajectory for the first case obtained with the HCW model. Figure 4.3 shows the distance between the leader and the follower satellite as a function of time, the trajectory in the LVLH reference frame, and the velocity and the position components for the first case obtained with the HCW model. Worth noting that the obtained trajectory is an in-plane one with the z component remaining equal to zero for the entire time.

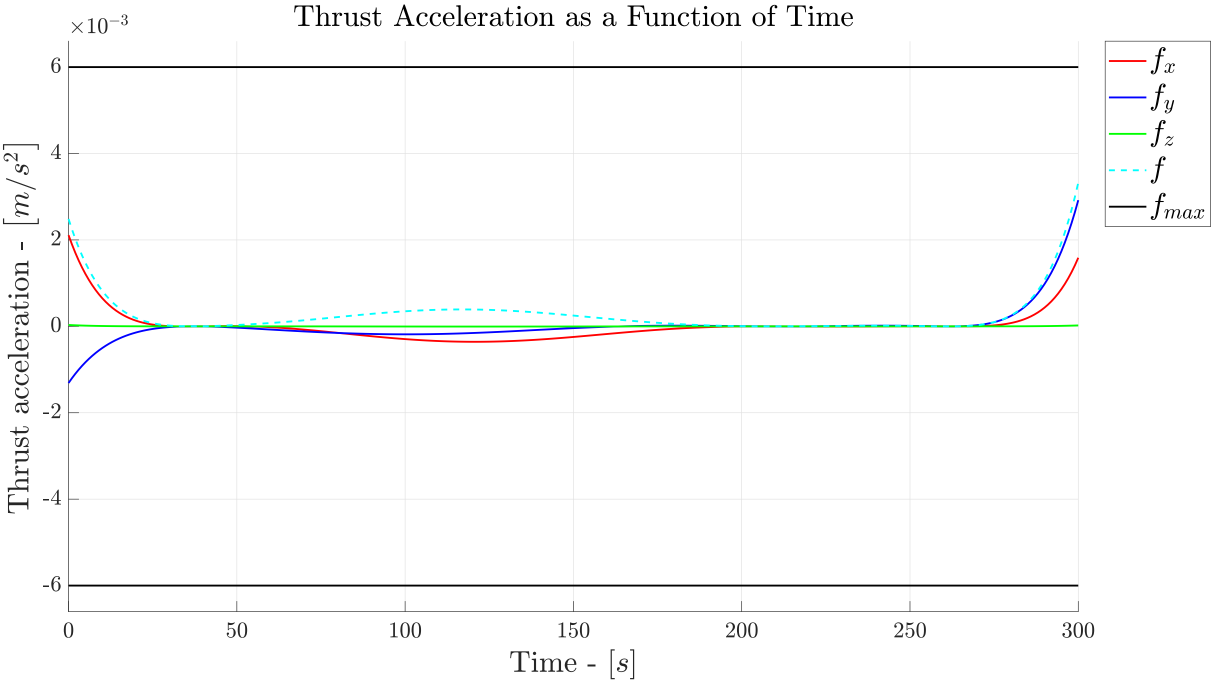

Figure 4.1: Total Thrust Acceleration and Thrust Acceleration Components as a Function of Time - Case 1.

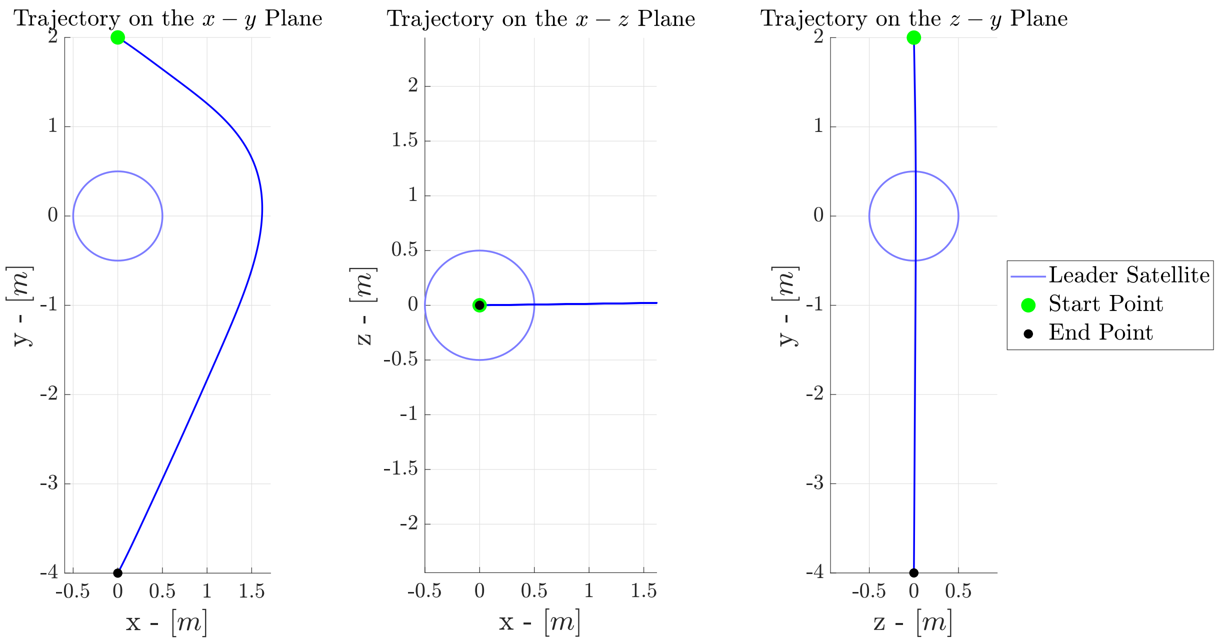

Figure 4.2: Projection of the Trajectory - Case 1.

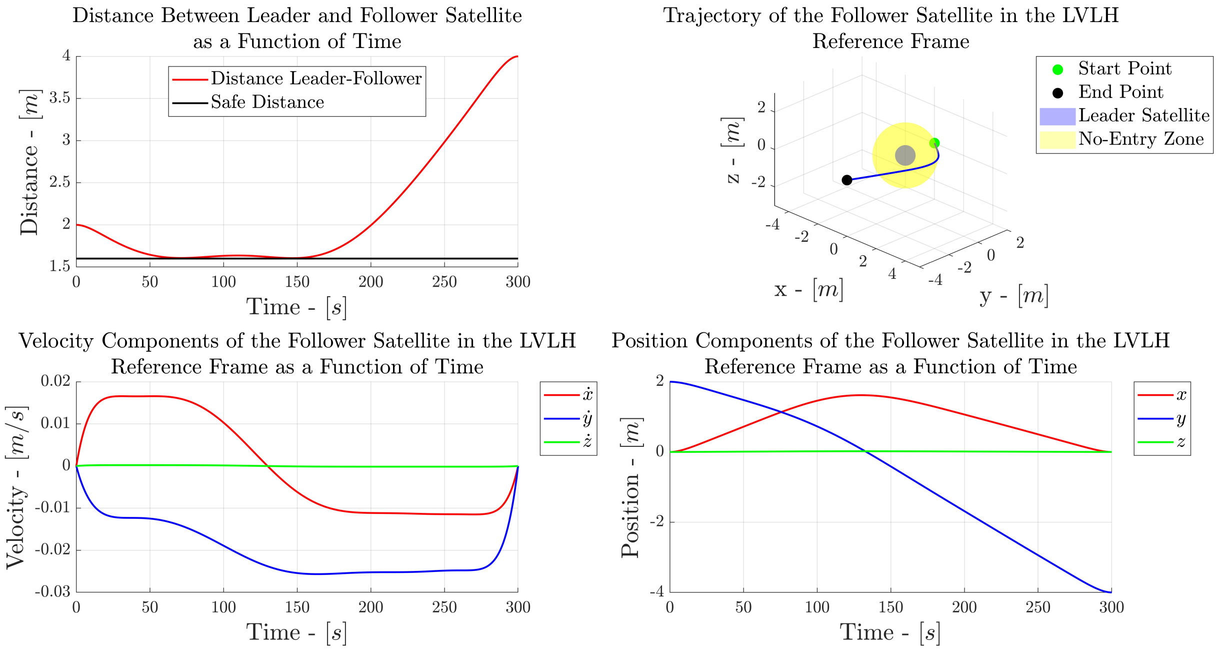

Figure 4.3: Distance Between Leader and Follower Satellites as a Function of Time (Top Left), Trajectory on the LVLH Reference Frame (Top Right), Velocity Components as a Function of Time (Bottom Left), Position Components as a Function of Time (Bottom Right) - Case 1.

Figure 4.4 and Figure 4.5 show the thrust acceleration history and the projection of the trajectory for the last case obtained with the HCW model. Figure 4.6 shows the distance between the leader and the follower satellite as a function of time, the trajectory in the LVLH reference frame, and the velocity and the position components for the last case obtained with the HCW model. Worth noting that in this case the obtained trajectory is not an in-plane one.

It can be noted that in both cases the obtained total thrust acceleration, f, tries to resemble a bang-bang solution, which is the optimal one when dealing with the fuel-optimal problem; such solution cannot be obtained by means of the implemented method due to the mathematical description of the shape function. It is also worth noting a common feature of all the obtained solutions: the follower satellite tends to get closer to the leader one, up until the safe distance is reached, and then reaches the final position imposed by the boundary conditions. Finally note that the same results are obtained when using the TH model.

The figures for the remaining cases show a similar behavior and can be found in Appendix A.

Overall, the developed tool for trajectory optimization can be considered successful, based on the obtained results and their comparison with data available from literature.

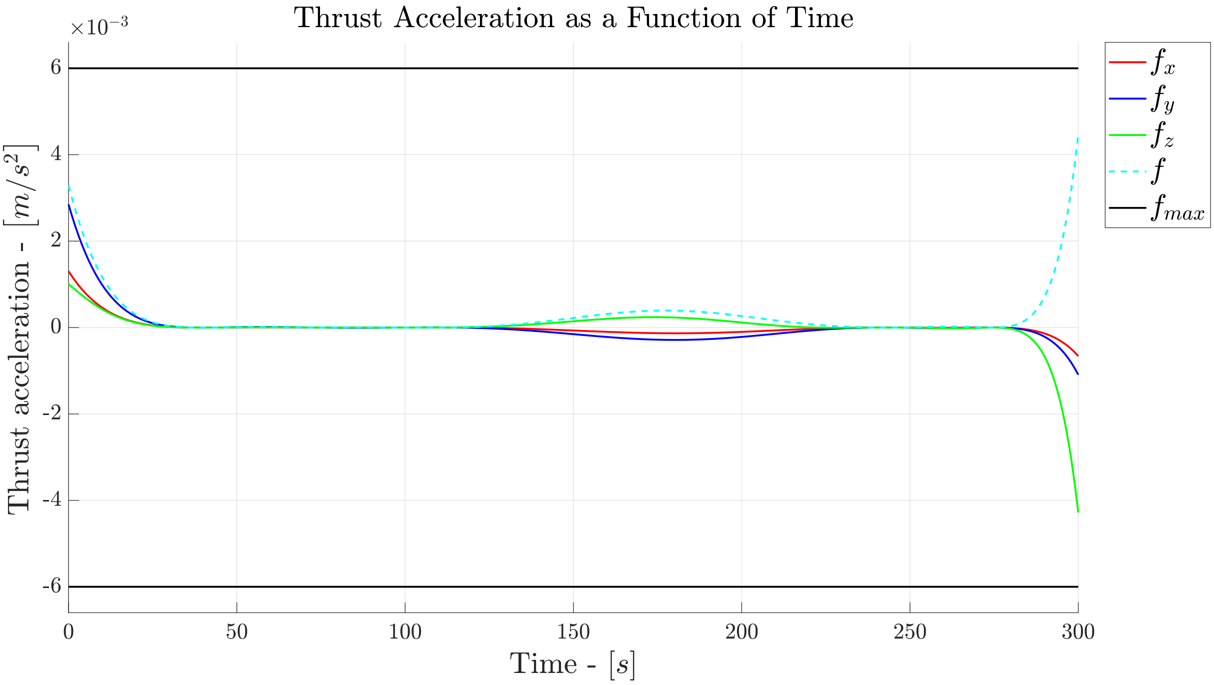

Figure 4.4: Total Thrust Acceleration and Thrust Acceleration Components as a Function of Time - Case 6.

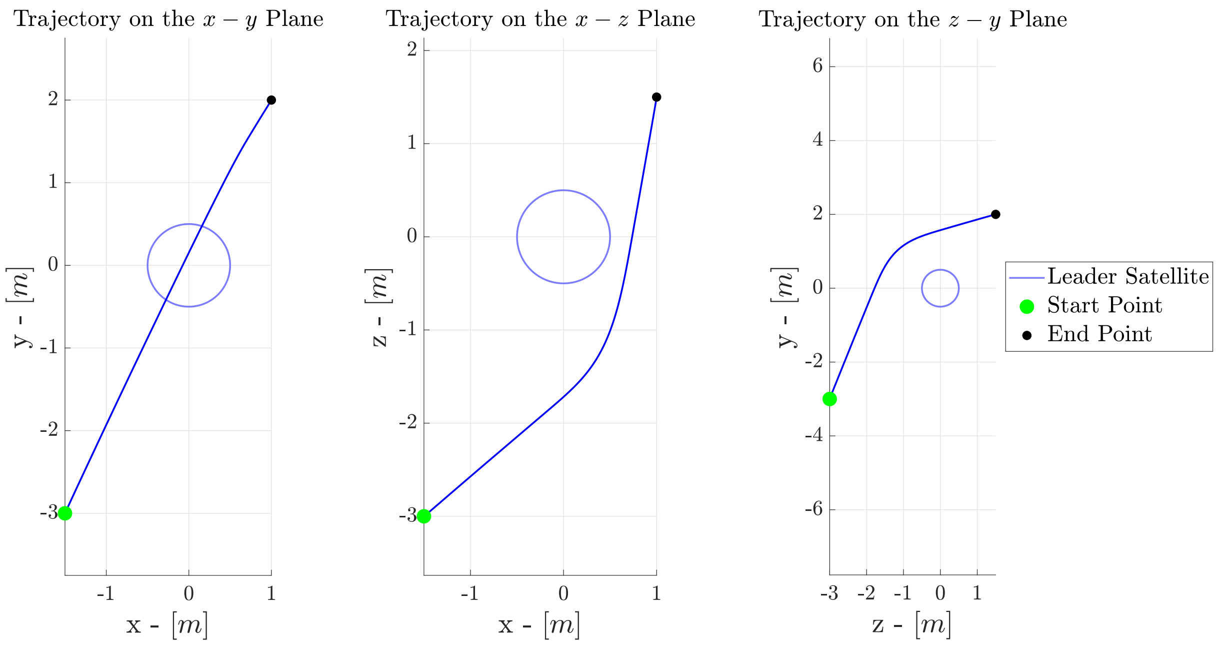

Figure 4.5: Projection of the Trajectory - Case 6.

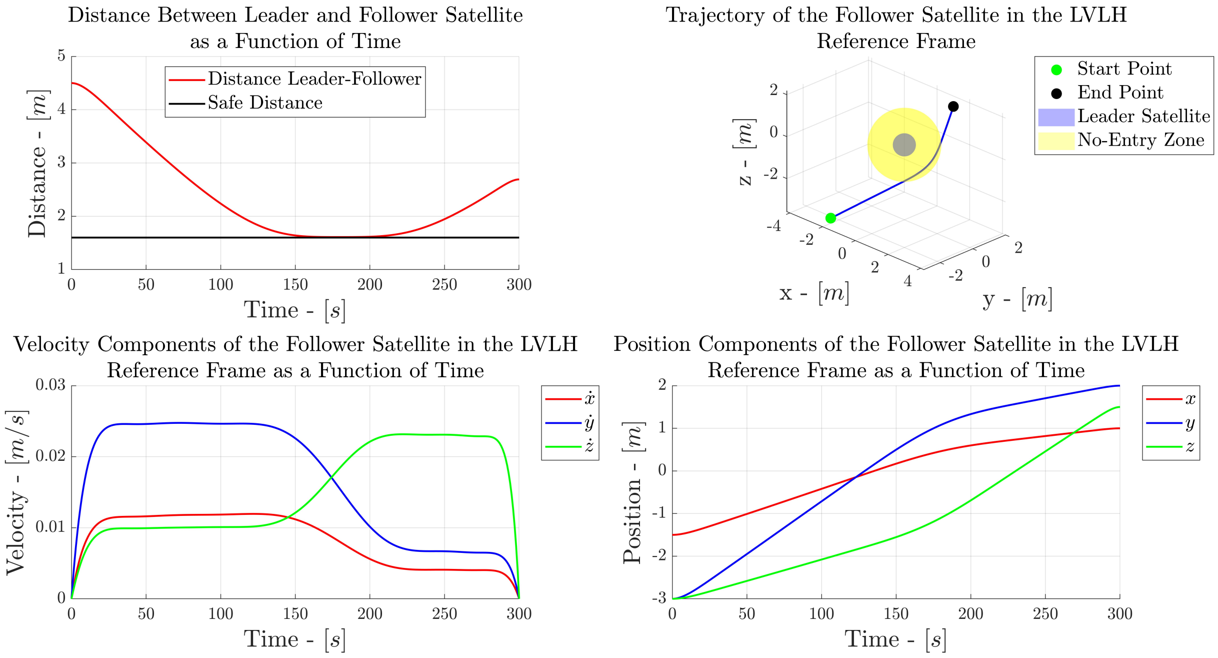

Figure 4.6: Distance Between Leader and Follower Satellites as a Function of Time (Top Left), Trajectory on the LVLH Reference Frame (Top Right), Velocity Components as a Function of Time (Bottom Left), Position Components as a Function of Time (Bottom Right) - Case 6.

4.2. An Analysis on the Order of the Bézier Series

This section presents an analysis on how the order of the Bézier series affects the results of the trajectory optimization in terms of obtained ΔV, computation time, and thrust acceleration.
Additionally, it has been studied the difference between the trajectory, obtained by means of the Bézier series, and the one obtained after applying the control to the dynamics of the follower satellite. To make this part clearer, the following process has been performed:

1. Perform trajectory optimization, as presented in Chapter 3.
2. With the geometric coefficients obtained at the previous step, compute the trajectory and the thrust acceleration components:

$$\boldsymbol{x}(t) = \begin{bmatrix} x(t) & y(t) & z(t) \end{bmatrix} = \boldsymbol{x_{dim}}(\tau) * s_{scal} = ...$$

$$... = \begin{bmatrix} \boldsymbol{B_x}(\tau) & \boldsymbol{B_y}(\tau) & \boldsymbol{B_z}(\tau) \end{bmatrix} \begin{bmatrix} \boldsymbol{P_{x,dim}} \\ \boldsymbol{P_{y,dim}} \\ \boldsymbol{P_{z,dim}} \end{bmatrix} * s_{scal} \tag{4.1}$$

$$\boldsymbol{f}(t) = \begin{bmatrix} f_x(t) & f_y(t) & f_z(t) \end{bmatrix} = \boldsymbol{f_{dim}}(\tau) * \frac{s_{scal}}{ToF^2} = ...$$

$$... = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \boldsymbol{B''_x}(\tau)\boldsymbol{P_{x,dim}} \\ \boldsymbol{B''_y}(\tau)\boldsymbol{P_{y,dim}} \\ \boldsymbol{B''_z}(\tau)\boldsymbol{P_{z,dim}} \end{bmatrix} * \frac{s_{scal}}{ToF^2} + ...$$

$$... + \begin{bmatrix} 0 & -2N & 0 \\ 2N & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{B'_x}(\tau)\boldsymbol{P_{x,dim}} \\ \boldsymbol{B'_y}(\tau)\boldsymbol{P_{y,dim}} \\ \boldsymbol{B'_z}(\tau)\boldsymbol{P_{z,dim}} \end{bmatrix} * \frac{s_{scal}}{ToF} + ...$$

$$... + \begin{bmatrix} -3N^2 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & N^2 \end{bmatrix} \begin{bmatrix} \boldsymbol{B_x}(\tau)\boldsymbol{P_{x,dim}} \\ \boldsymbol{B_y}(\tau)\boldsymbol{P_{y,dim}} \\ \boldsymbol{B_z}(\tau)\boldsymbol{P_{z,dim}} \end{bmatrix} * s_{scal} \tag{4.2}$$

3. Substitute the thrust acceleration components in the HCW dynamical model, presented in Sec. 2.2.1, and perform the numerical integration to obtain the actual trajectory of the satellite, $\boldsymbol{x_{control}}(t)$.

4. Compute the absolute value of the difference between the two:

$$|\boldsymbol{x}_{\boldsymbol{control}}(t) - \boldsymbol{x}_{\boldsymbol{Bézier}}(t)| \tag{4.3}$$

In which $\boldsymbol{x}_{\boldsymbol{control}}(t)$ is the trajectory obtained after numerical integration, while $\boldsymbol{x}_{\boldsymbol{Bézier}}(t)$ is the one obtained by means of Eq. (4.1).

Figure 4.7 shows the ΔV as a function of the order of the Bézier series for all the six cases presented in Sec. 4.1. It can be noted that in all the cases the ΔV decreases for increasing orders of the Bézier series up to $\boldsymbol{n} = 10$; instead, when the order is $\boldsymbol{n} \geq 12$ the ΔV remains almost constant. Also note that in a few instances, when $\boldsymbol{n} \geq 12$, the obtained ΔV is higher than the one obtained by adjacent orders of the Bézier series, this is probably due to some issues in the optimization process.

Figure 4.8, instead, shows the computation time as a function of the order of the Bézier series for all six cases presented in Sec. 4.1. It can be noted that the computation time increases for increasing order of the Bézier series; this is expected since, as the order of the Bézier series increases, the number of optimization variables increases as well.

Finally note that by analyzing these two figures it is possible to conclude that a Bèzier series of order 10 is the best trade-off between ΔV and computation time. Indeed, for such order of the Bèzier series the computation time is still limited; on the other hand, the ΔV does not decrease significantly for higher orders of the series.

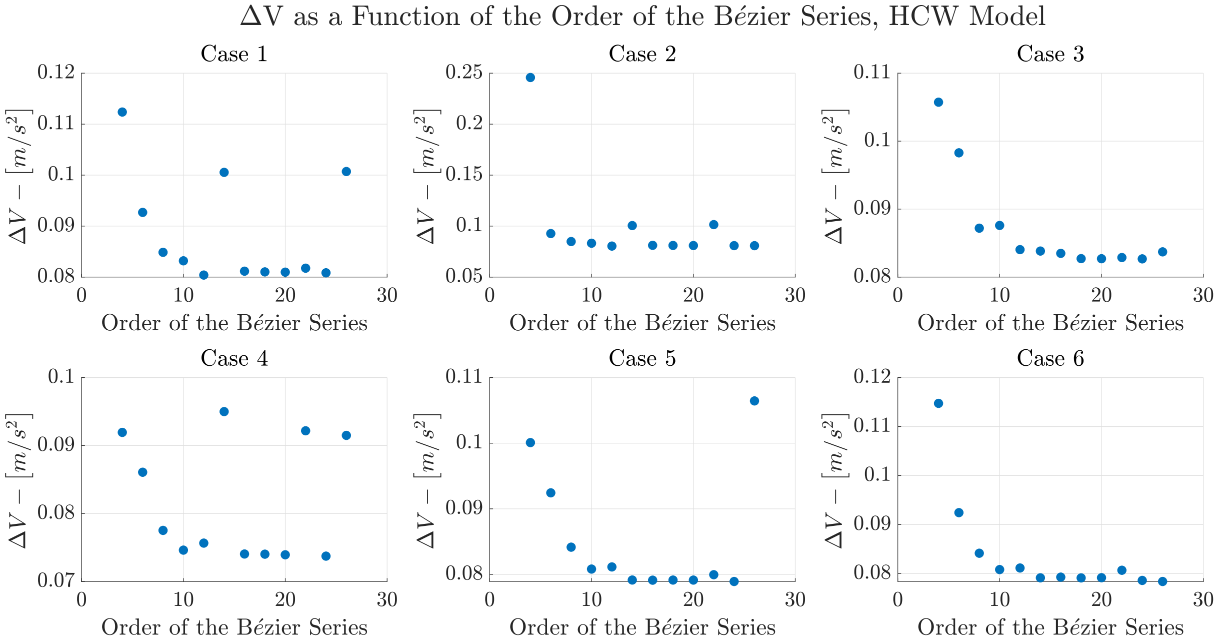

Figure 4.7: ΔV as a Function of the Order of the Bézier Series.

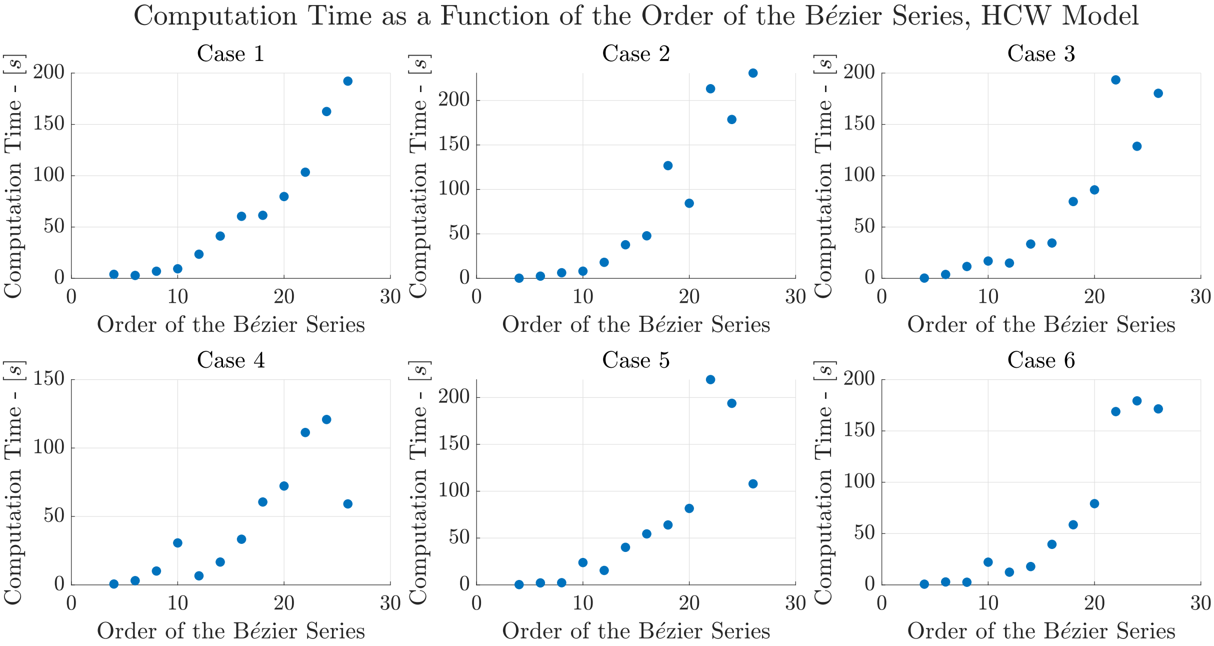

Figure 4.8: Computation Time as a Function of the Order of the Bézier Series.

Figure 4.9 shows the thrust acceleration components and the total thrust acceleration as a function of time, for different orders of the Bézier series for case 2. Figure 4.10, instead, shows the same quantities but for case 4. It can be noted that in both cases, for $\boldsymbol{n} \leq 10$, the obtained thrust acceleration profile varies with the order of the Bézier series; on the other hand, for $\boldsymbol{n} \geq 12$ the thrust acceleration profile remains almost unchanged. It is worth noting that for lower orders of the Bézier series the thrust acceleration profiles are quite simple and resemble a second or third order polynomial; this is expected since Bézier series of lower orders are limited in terms of the possible functions that they can represent and, therefore, only simple functions can be obtained. For higher orders of the Bézier series, instead, the thrust acceleration profiles remain almost unchanged since an optimal solution has been reached and a different profile would lead to a higher ΔV. Finally, as already mentioned in Sec. 4.1, it is possible to note that, as the order of the Bézier series increases, the obtained thrust profile tends to resemble as much as possible a bang-bang solution. Obtaining such solution is limited by the mathematical nature of the Bézier series but, more in general, by the use of a shape-based method.

The figures of the thrust acceleration components and the total thrust acceleration as a function of time, for different orders of the Bézier series, for the remaining cases show a similar behavior and can be found in Appendix B.

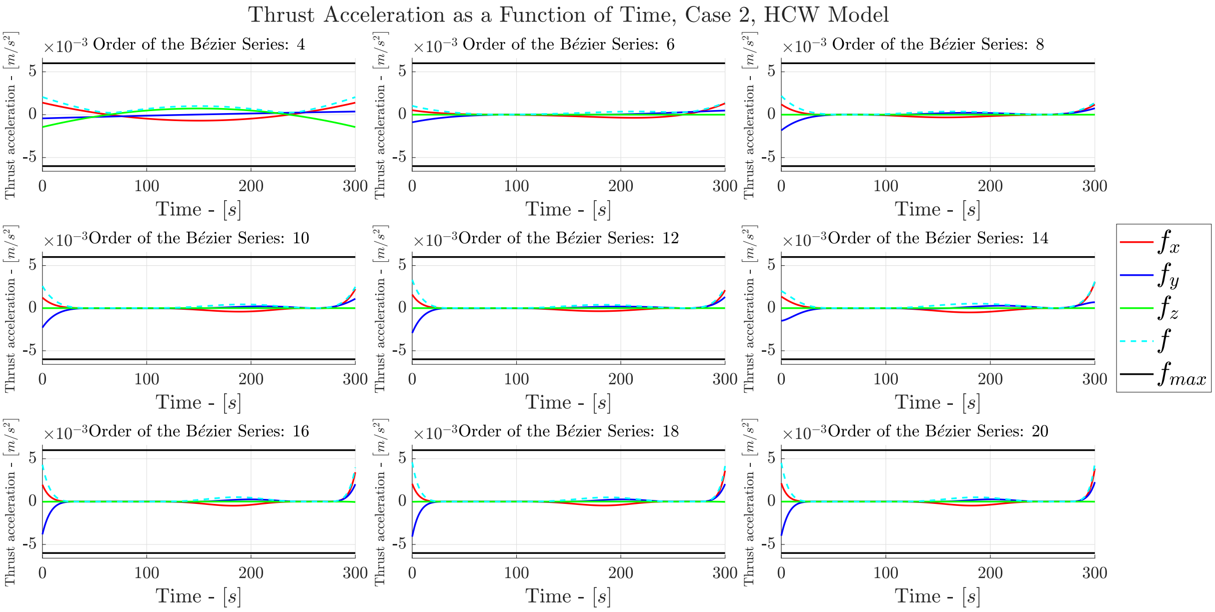

Figure 4.9: Thrust Acceleration as a Function of Time for Different Orders of the Bézier Series - Case 2.

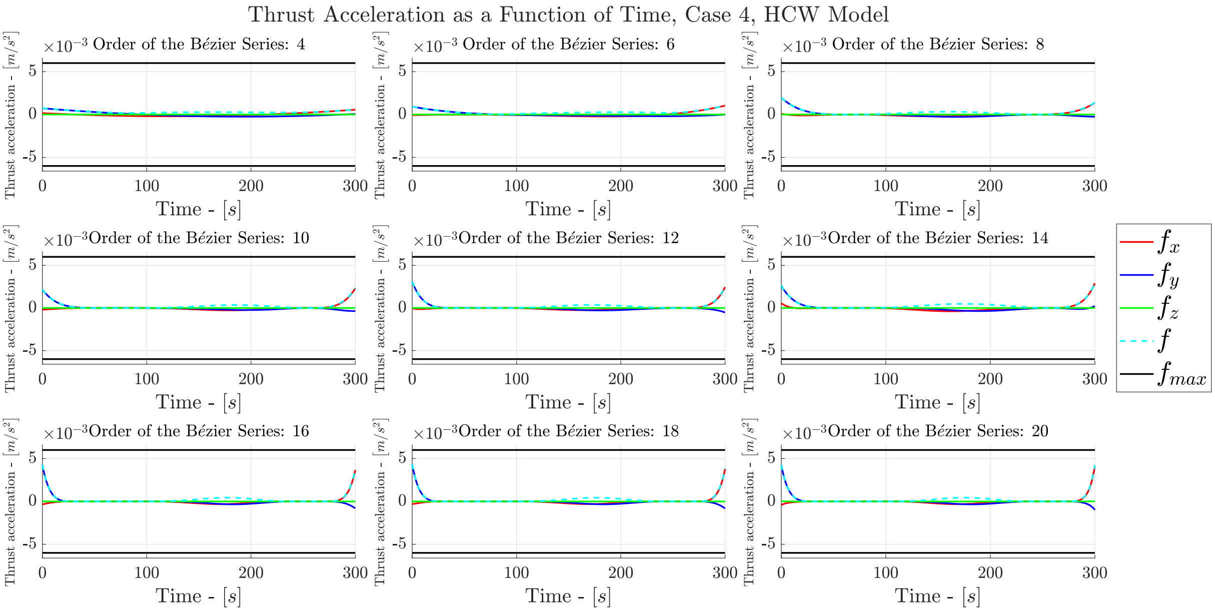

Figure 4.10: Thrust Acceleration as a Function of Time for Different Orders of the Bézier Series - Case 4.

Figure 4.11 shows the plot of Eq. (4.3) for different orders of the Bézier series for case 2. Figure 4.12, instead, shows the same quantities but for case 5. It can be noted that in both cases the difference between the trajectory, computed by means of the Bézier series, and the one obtained by applying the control is in the order of 10^{-14} m, therefore it is negligible. Additionally, when the order of the Bézier series is ≤ 8 the obtained behavior is typical of when the values are close to the machine epsilon; when the order of the Bézier series is ≥ 8 such behavior is still present, but it is more difficult to grasp due to the values being one order of magnitude bigger with respect to the former cases. Another trend that can be noted in most cases is that the difference between the computed trajectory and the one obtained by applying the control is increasing in time; this is expected in all of the dynamical problems of such kind. However, it must be pointed out that such trend is not always present, an example can be seen in Figure 4.11 for the x coordinate when the order of the Bézier series is equal to 12. A possible explanation of this unusual trend could be the fact that, in the considered case, the x component is initially equal to 0, then it increases with time until a maximum is reached, and finally it decreases until 0 is reached again; the decreasing in the x coordinate might be related to the decrease in the difference between the computed trajectory and the one obtained by applying the control. However, this explanation can justify this unusual trend only partially, indeed for the same case different trends are obtained for different orders of the Bézier series. The figures for the remaining cases show a similar behavior and can be found in Appendix B.

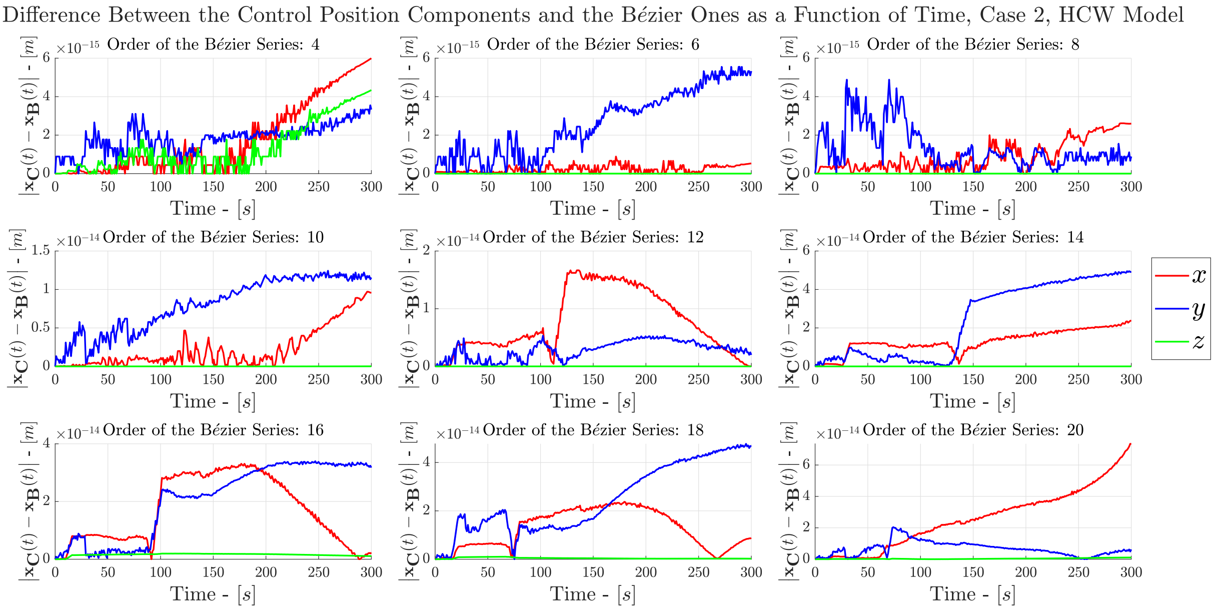

Figure 4.11: $|\boldsymbol{x}_{control}(t) - \boldsymbol{x}_{Bézier}(t)|$ for Different Orders of the Bézier Series - Case 2.

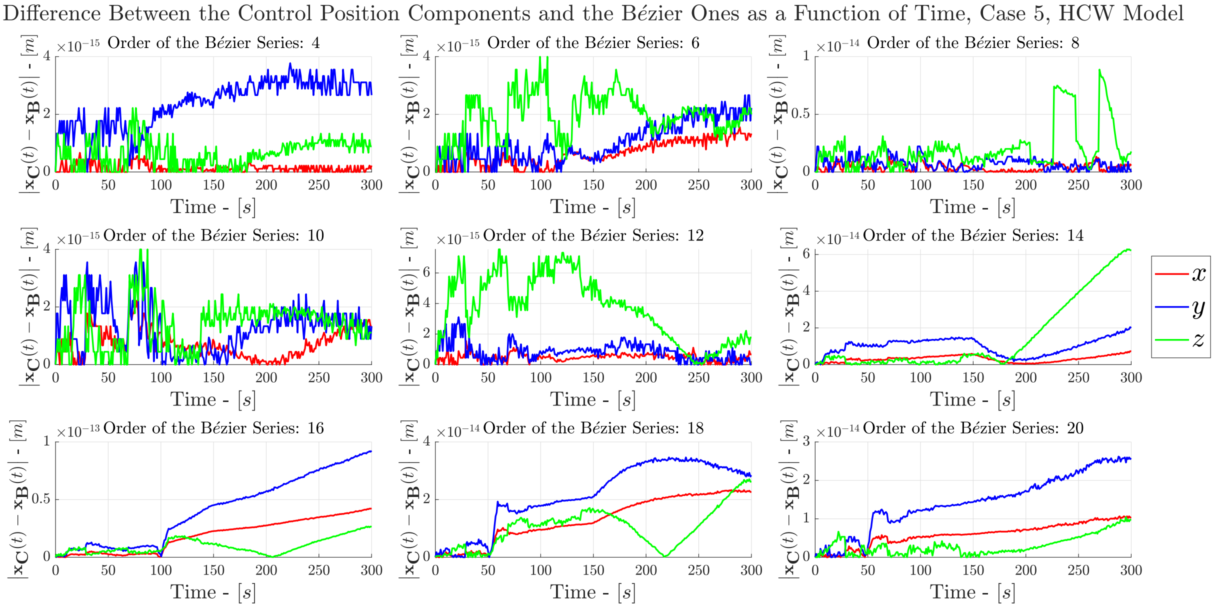

Figure 4.12: $|\boldsymbol{x_{control}}(t) - \boldsymbol{x_{Bézier}}(t)|$ for Different Orders of the Bézier Series - Case 5.

5 | Formation Reconfiguration

This chapter presents the developed strategy for formation reconfiguration. This chapter is structured as follows: the first part presents the general idea that has been implemented; the second part provides the MATLAB implementation highlighting the criticalities of the approach; the third part proposes a solution for such criticalities. Finally, the last part presents the results for some test cases.

5.1. Introduction

The developed tool for formation reconfiguration has one important constraint: it can be applied when the involved satellites are flying on a circular orbit. Even if this is a quite limiting constraint, it is worth noting that most of the applications for formation flying are coupled with circular orbits.

The general idea for the developed strategy for formation reconfiguration is the following: once the trajectory for a follower satellite is computed, such result is valid at any point along the orbit; this is true when dealing with circular orbits. Due to the fact that the computed trajectory is valid at any time, collision avoidance between the follower satellites can be enforced by time-shifting the starting maneuvering time of each satellite.

5.2. Formation Reconfiguration, a Simple Approach to the Problem

The workflow of the implemented tool for formation reconfiguration, shown in Figure 5.1, is the following:

1. Each satellite performs its own trajectory optimization, as presented in Chapter 3.
2. The minimum distance between the involved satellites is computed: if such distance is higher than the minimum prescribed value, it is possible to perform formation reconfiguration without a shift in maneuvering times of the involved satellites, and the algorithm ends.

3. If the minimum distance between the involved satellites is lower than the minimum prescribed value, then formation reconfiguration is performed by shifting the maneuvering times of the involved satellites by means of a genetic algorithm.

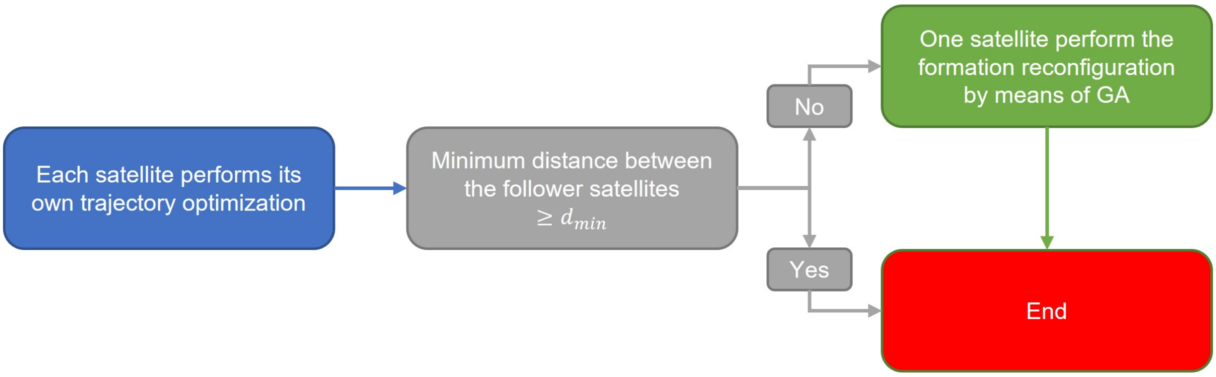
Each satellite performs its own trajectory optimization
Minimum distance between the follower satellites $\geq d_{min}$
No
Yes
One satellite perform the formation reconfiguration by means of GA
End

Figure 5.1: Workflow of Formation Reconfiguration, a Simple Algorithm.

Note that trajectory optimization is performed individually by each satellite; on the other hand formation reconfiguration by means of a genetic algorithm is performed by a devoted satellite. In this way formation reconfiguration is performed by means of a decentralized approach and exploiting the computational power of all the involved satellites; it is worth noting that to do so a strong inter-satellite communication is necessary.
Concerning the formation reconfiguration by means of a genetic algorithm, it is an optimization problem in which the objective function is to minimize the time required to complete the formation reconfiguration:

$$\begin{aligned} \min_{\boldsymbol{t_{start}}} \quad & max\left(\boldsymbol{t_{start}}\right) \\ s.t. \quad & d_{min} - \boldsymbol{d_{FF}}(t) \leq \mathbf{0} \end{aligned} \tag{5.1}$$

In which:

- $\boldsymbol{t_{start}}$ is the optimization variable; it is a vector that contains the starting maneuvering times for each satellite.
- $\boldsymbol{d_{FF}}(t)$ is a vector that contains the distances between all the follower satellites taken two at a time.
- d_{min} is the minimum distance between the follower satellites.

It can be noted that the fitness function of the genetic algorithm is to reduce the maximum of the starting times, while the nonlinear inequality constraints enforce a minimum distance between all the follower satellites in the form of $d_{min} - \boldsymbol{d_{FF}}(t) \leq \mathbf{0}$.

Some remarks: the optimization variables can range between 0 and $ToF * (n_{sats} - 1)$ with n_{sats} being the number of follower satellites; the reason for the upper boundary is that, due to the fact that the leader satellite is on a circular orbit, all the possible reconfiguration strategies are found within this maximum value and outside of this range the same results will be found. From the MATLAB implementation point of view, the vector of optimization variables is made dimensionless as follows:

$$\boldsymbol{\tau_{start}} = \frac{\boldsymbol{t_{start}}}{ToF * (n_{sats} - 1)} \tag{5.2}$$

In this way all the optimization variables can range between 0 and 1 with 1 corresponding to a time of $ToF * (n_{sats} - 1)$.

Finally, note that there is no need to impose constraints on the maximum total thrust acceleration and minimum distance between the follower and leader satellites since such constraints are already enforced in the trajectory optimization for each satellite.

After having introduced the dimensionless time, the problem described in Eq. (5.1) can be rewritten as follows:

$$\begin{aligned} \min_{\boldsymbol{\tau_{start}}} \quad & max\left(\boldsymbol{\tau_{start}}\right) \\ s.t. \quad & d_{min} - \boldsymbol{d_{FF}}(\tau) \leq \mathbf{0} \end{aligned} \tag{5.3}$$

Two remarks: the reason behind the choice of a genetic algorithm is that, unlike for trajectory optimization, for such optimization problem there is not a mathematical meaning in terms of derivative of the fitness function with respect to the optimization variables; therefore, a stochastic, population-based algorithm that searches randomly by mutation and crossover among population members [21] is more suitable for the solution of such problem rather than a gradient-based approach, which also requires an initial guess. Finally, to solve this optimization problem the MATLAB's built-in *ga* function has been exploited.

5.2.1. Limits of This Approach

With the developed algorithm there is one big criticality: there are several instances in which a solution to the formation reconfiguration cannot be found.

An example in which it is not possible to find a solution is shown in Figure 5.2 and Figure 5.3. Figure 5.2 shows the distances between the follower satellites and the distances between the leader and the follower satellites in two cases: in the top part such distances are obtained in case all the satellites perform the reconfiguration maneuvers at the same time; in the bottom part, instead, are the results after the formation reconfiguration algorithm. In Figure 5.3, instead, are reported the final trajectories for all the satellites.

This happens when the involved satellites get close to each other in the proximity of the initial or final positions. Indeed, in such cases, even delaying the starting maneuvering time as much as possible would lead to an unfeasible trajectory.
To solve this issue a more complex algorithm is needed and will be presented in Sec. 5.3.

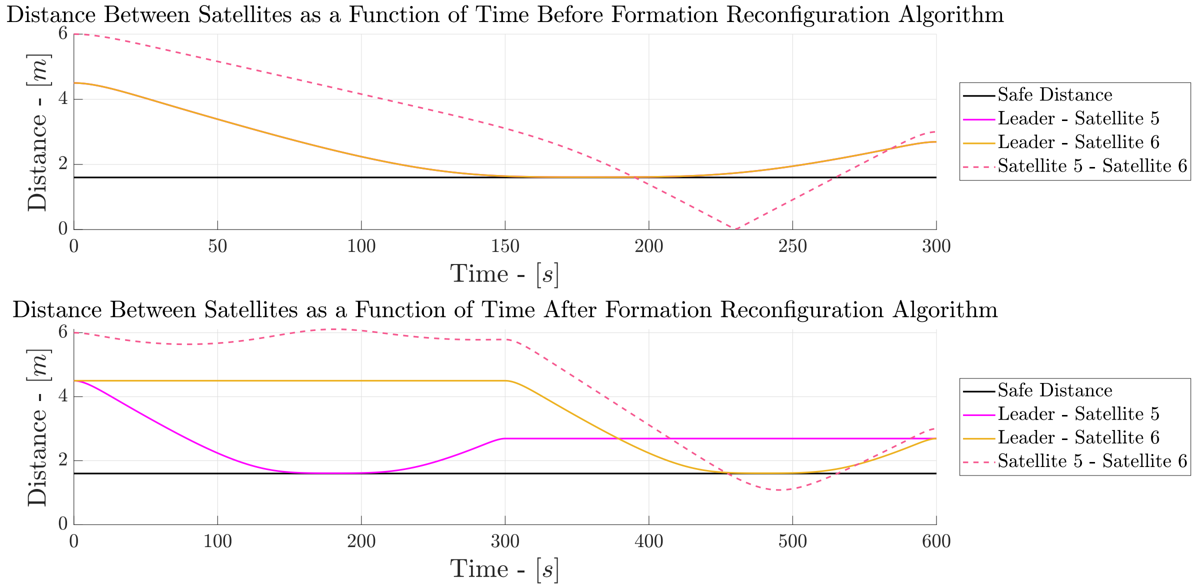

Figure 5.2: Distance Between Satellites as a Function of Time When Reconfiguration Maneuvers are Performed at the Same Time (Top) and After Formation Reconfiguration Algorithm (Bottom).

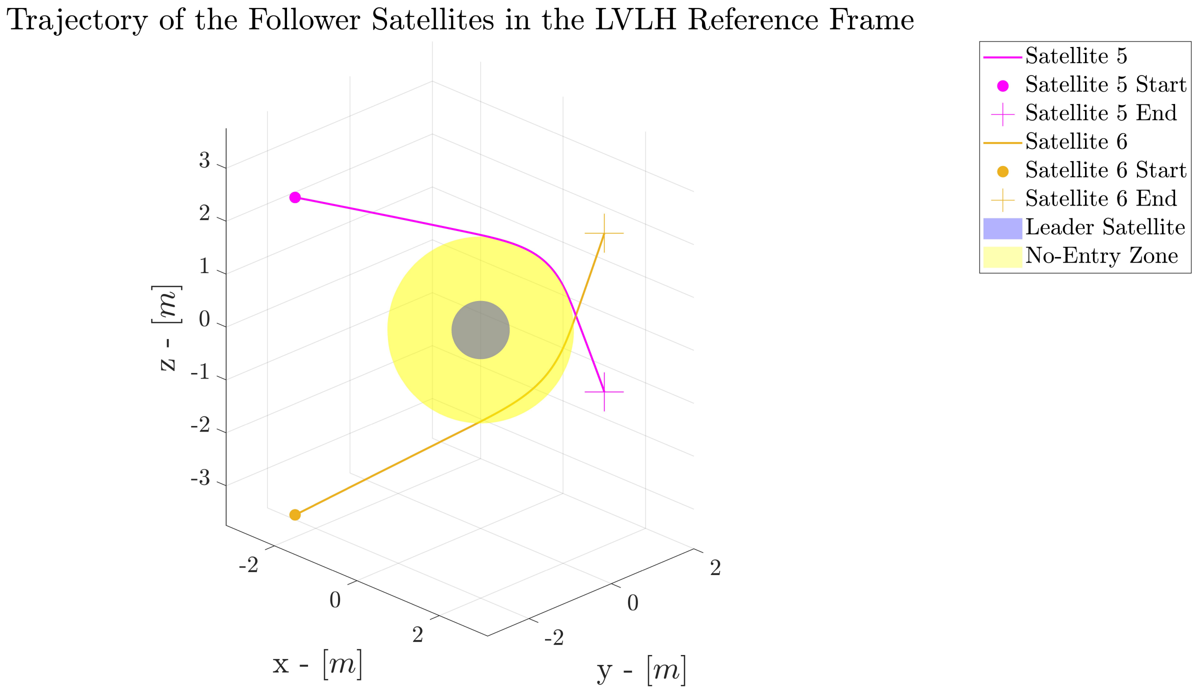

Figure 5.3: Trajectory of the Satellites in the LVLH Reference Frame.

5.3. Advanced Formation Reconfiguration

The idea of the advanced formation reconfiguration algorithm is to add nonlinear inequality constraints in the trajectory optimization problem to avoid collision with the other follower satellites while keeping the process as much decentralized as possible.

5.3.1. Introduction

In order to make all the parts of the algorithm clear, a brief overview of the whole process could be useful. The workflow of the algorithm, represented in Figure 5.4, is the following:

1. The first part of the algorithm is the same as the one found in Sec. 5.2.
2. If such algorithm fails in finding a possible reconfiguration strategy, some computations, necessary for the next part of the process, are performed.
3. In the last part of the algorithm a *while* loop, in which a possible reconfiguration strategy is searched, is performed.

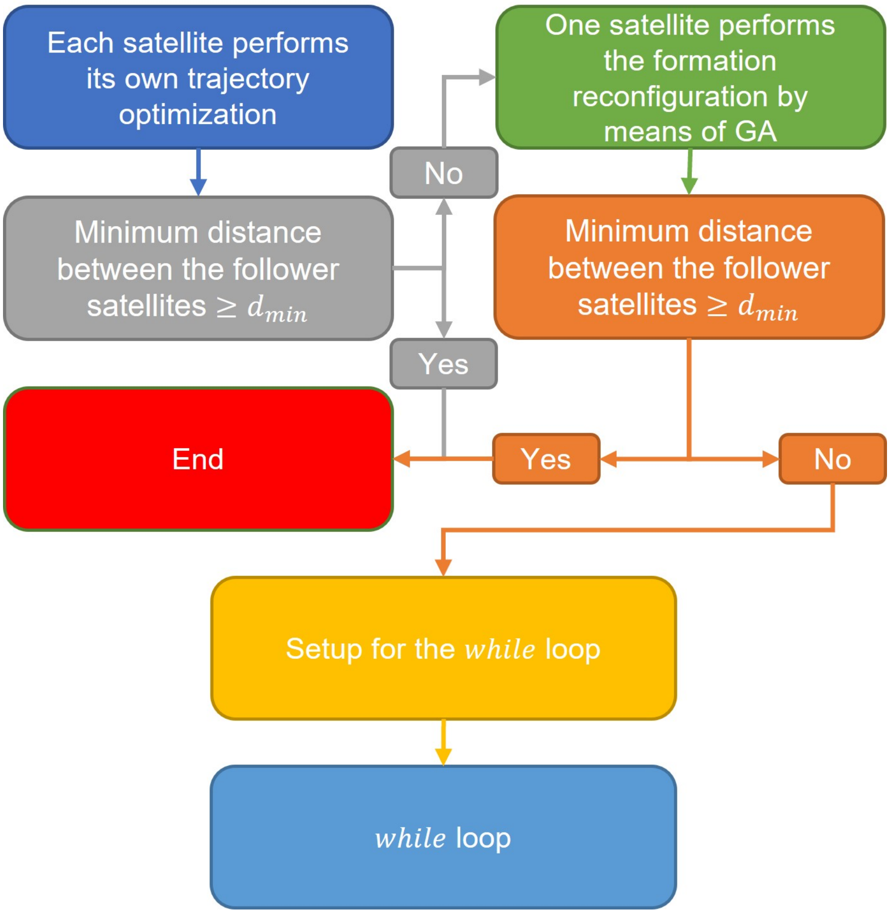
Each satellite performs its own trajectory optimization
One satellite performs the formation reconfiguration by means of GA
No
Minimum distance between the follower satellites $\geq d_{min}$
Minimum distance between the follower satellites $\geq d_{min}$
Yes
End
Yes
No
Setup for the *while* loop
while loop

Figure 5.4: Workflow of the Advanced Formation Reconfiguration Algorithm.

In the setup for the *while* loop two operations are performed:

1. The distances between the initial and the final positions of all the involved satellites are computed. Those values will be necessary when defining the additional nonlinear inequality constraints in the trajectory optimization process.
2. All possible pairs of follower satellites are considered, and, by means of a genetic algorithm, it is checked if a formation reconfiguration, between the two considered satellites, is possible. If formation reconfiguration is not possible, the involved satellites will compute a new trajectory by adding the additional nonlinear inequality constraints. Otherwise, if a satellite does not have problems with all the other satellites, its trajectory will not be computed again but the same results obtained in the first part of the algorithm will be kept valid.

The reason for the second step of the setup process is to identify which are the satellites that do not allow the reconfiguration and so to modify their trajectory.

5.3.2. Additional Nonlinear Inequality Constraints

The additional nonlinear inequality constraints are in the form of a minimum distance between the trajectory of the follower satellites with respect to some prescribed positions; in particular, such prescribed positions are the boundary conditions of the other follower satellites. Such nonlinear inequality constraints are added so that each satellite will try to avoid the initial and the final positions of all the other follower satellites, and a feasible formation reconfiguration could be obtained.

To make this clearer, consider the following example: a formation reconfiguration is required between the satellites 1, 2, and 3. By performing the genetic algorithm between satellite 1 and satellite 2, and between satellite 1 and satellite 3, it is found that formation reconfiguration is not possible. Therefore, the additional nonlinear inequality constraints added to satellite 1 are the following:

$$\begin{aligned} &d_{min,\boldsymbol{Sat_1}-\boldsymbol{x_2}(0)} - \boldsymbol{d}_{\boldsymbol{Sat_1}-\boldsymbol{x_2}(0)}(t) \leq \mathbf{0} \\ &d_{min,\boldsymbol{Sat_1}-\boldsymbol{x_2}(ToF)} - \boldsymbol{d}_{\boldsymbol{Sat_1}-\boldsymbol{x_2}(ToF)}(t) \leq \mathbf{0} \\ &d_{min,\boldsymbol{Sat_1}-\boldsymbol{x_3}(0)} - \boldsymbol{d}_{\boldsymbol{Sat_1}-\boldsymbol{x_3}(0)}(t) \leq \mathbf{0} \\ &d_{min,\boldsymbol{Sat_1}-\boldsymbol{x_3}(ToF)} - \boldsymbol{d}_{\boldsymbol{Sat_1}-\boldsymbol{x_3}(ToF)}(t) \leq \mathbf{0} \end{aligned} \tag{5.4}$$

In which:

- $\boldsymbol{d}_{\boldsymbol{Sat_1}-\boldsymbol{x_2}(0)}(t)$ is the distance between satellite 1 and the initial position of satellite 2, while $d_{min,\boldsymbol{Sat_1}-\boldsymbol{x_2}(0)}$ is the prescribed minimum distance.

- $\boldsymbol{d_{Sat_1-x_2(ToF)}}(t)$ is the distance between satellite 1 and the final position of satellite 2, while $d_{min,\boldsymbol{Sat_1-x_2(ToF)}}$ is the prescribed minimum distance.
- $\boldsymbol{d_{Sat_1-x_3(0)}}(t)$ is the distance between satellite 1 and the initial position of satellite 3, while $d_{min,\boldsymbol{Sat_1-x_3(0)}}$ is the prescribed minimum distance.
- $\boldsymbol{d_{Sat_1-x_3(ToF)}}(t)$ is the distance between satellite 1 and the final position of satellite 3, while $d_{min,\boldsymbol{Sat_1-x_3(ToF)}}$ is the prescribed minimum distance.

Generalizing Eq. (5.4), for each satellite i the following nonlinear inequality constraints are added to the trajectory optimization problem:

$$\begin{aligned} &d_{min,\boldsymbol{Sat_i-x_j(0)}} - \boldsymbol{d_{Sat_i-x_j(0)}}(t) \leq \mathbf{0} \\ &d_{min,\boldsymbol{Sat_i-x_j(ToF)}} - \boldsymbol{d_{Sat_i-x_j(ToF)}}(t) \leq \mathbf{0} \end{aligned} \qquad j = 1, 2, ..., n_{sats},\ j \neq i \tag{5.5}$$

Concerning the prescribed minimum distances, they are obtained considering two satellites at a time. In particular, for each possible satellite pair, such minimum distance is the minimum among the following quantities:

- The minimum allowed distance between the satellites, d_{min}, as already presented in Sec. 3.1.3.
- The distance between the initial positions of the considered pair.
- The distance between the final positions of the considered pair.
- The distance between the initial position of one satellite and the final one of the other.
- The distance between the final position of one satellite and the initial one of the other.

Considering again the example with satellites 1, 2, and 3, the following is valid:

$$\begin{aligned} d_{min,\boldsymbol{Sat_1-x_2(0)}} = d_{min,\boldsymbol{Sat_1-x_2(ToF)}} &= min(d_{min}, \left|\boldsymbol{x_2}(0) - \boldsymbol{x_1}(0)\right|, ... \\ &... \left|\boldsymbol{x_2}(0) - \boldsymbol{x_1}(ToF)\right|, \left|\boldsymbol{x_2}(ToF) - \boldsymbol{x_1}(0)\right|, \left|\boldsymbol{x_2}(ToF) - \boldsymbol{x_1}(ToF)\right|) \\ d_{min,\boldsymbol{Sat_1-x_3(0)}} = d_{min,\boldsymbol{Sat_1-x_3(ToF)}} &= min(d_{min}, \left|\boldsymbol{x_3}(0) - \boldsymbol{x_1}(0)\right|, ... \\ &... \left|\boldsymbol{x_3}(0) - \boldsymbol{x_1}(ToF)\right|, \left|\boldsymbol{x_3}(ToF) - \boldsymbol{x_1}(0)\right|, \left|\boldsymbol{x_3}(ToF) - \boldsymbol{x_1}(ToF)\right|) \end{aligned} \tag{5.6}$$

Generalizing Eq. (5.6), for each satellite i the following is obtained:

$$\begin{aligned} d_{min,\boldsymbol{Sat_i}-\boldsymbol{x_j}(0)} = d_{min,\boldsymbol{Sat_i}-\boldsymbol{x_j}(ToF)} = min(d_{min}, ... \\ \left|\boldsymbol{x_j}(0) - \boldsymbol{x_i}(0)\right|, \left|\boldsymbol{x_j}(0) - \boldsymbol{x_i}(ToF)\right|, ... \\ \left|\boldsymbol{x_j}(ToF) - \boldsymbol{x_i}(0)\right|, \left|\boldsymbol{x_j}(ToF) - \boldsymbol{x_i}(ToF)\right| \end{aligned} \qquad j = 1, 2, ..., n_{sats},\ j \neq i \tag{5.7}$$

Due to the fact that $d_{min,\boldsymbol{Sat_i}-\boldsymbol{x_j}(0)} = d_{min,\boldsymbol{Sat_i}-\boldsymbol{x_j}(ToF)}$ the following notation will be used:

$$d_{min,\boldsymbol{Sat_i}-\boldsymbol{x_j}(0)} = d_{min,\boldsymbol{Sat_i}-\boldsymbol{x_j}(ToF)} = d_{min,\boldsymbol{Sat_i}-\boldsymbol{x_j}} \qquad j = 1, 2, ..., n_{sats},\ j \neq i \tag{5.8}$$

Such minimum distance is computed in this manner for two reasons: the first one is to avoid enforcing a too high constraint if the initial and final positions of the considered satellites are far from each other; in this case d_{min} would be used. The second one is to avoid an unfeasible trajectory optimization problem, that is when d_{min} is not the lowest computed value.

5.3.3. Advanced Formation Reconfiguration Algorithm

It is now possible to tackle the developed algorithm with additional details. Figure 5.5 shows the workflow of the developed algorithm; as already mentioned, the first part of the algorithm is exactly the same as the one presented in Sec. 5.2, and in case such algorithm fails in finding a feasible reconfiguration, a new process is performed.

As already mentioned, the first step of this new process is the setup of the *while* loop in which the minimum distances, $d_{min,\boldsymbol{Sat_i}-\boldsymbol{x_j}}$, are computed as already explained in Sec. 5.3.2. After that, all possible pairs of follower satellites are considered, and, by means of a genetic algorithm, it is checked if a formation reconfiguration, between the two considered satellites, is possible. If a satellite does not have problems with all the other satellites for formation reconfiguration, its trajectory will not be computed again but the same results obtained in the first part of the algorithm will be kept valid. Otherwise, the involved satellites will compute a new trajectory by adding the additional nonlinear inequality constraints presented in Sec. 5.3.2. Note that these additional constraints are only with respect to the satellites that have formation reconfiguration issues between each other; an example will be provided later in this section in order to make this concept clearer.

The advanced formation reconfiguration algorithm is based on a *while* loop. Before explaining the processes that are performed in that part of the algorithm, it is necessary to define the exit conditions from the *while* loop; there are two possible exit conditions:

- At a certain iteration no collisions are detected, this can be either before or after having performed the formation reconfiguration by means of the genetic algorithm.
- The maximum number of iterations is reached.

The difference between one iteration and the following one is in the imposed minimum distance: indeed, the minimum distance, $d_{min,\boldsymbol{Sat_i}-\boldsymbol{x_j}}$, is not set from the beginning of the *while* loop, but rather is reached at the last iteration; additionally, a total of three iterations are performed. To make this clearer, consider the following example: given two satellites, satellite 1 and satellite 2, the following has been computed for the minimum distances:

$$d_{min,\boldsymbol{Sat_1}-\boldsymbol{x_2}} = d_{min,\boldsymbol{Sat_2}-\boldsymbol{x_1}} = 1.2\ m \tag{5.9}$$

Therefore, the following values will be enforced at each iteration k:

$$\begin{aligned} k=1 \quad & d_{min,\boldsymbol{Sat_1}-\boldsymbol{x_2}} = d_{min,\boldsymbol{Sat_2}-\boldsymbol{x_1}} = 0.1\ m \\ k=2 \quad & d_{min,\boldsymbol{Sat_1}-\boldsymbol{x_2}} = d_{min,\boldsymbol{Sat_2}-\boldsymbol{x_1}} = 0.65\ m \\ k=3 \quad & d_{min,\boldsymbol{Sat_1}-\boldsymbol{x_2}} = d_{min,\boldsymbol{Sat_2}-\boldsymbol{x_1}} = 1.2\ m \end{aligned} \tag{5.10}$$

It is worth noting that 0.1 m is the lowest value, and in case a lower value is obtained in Eq. (5.7) the constraint is dropped since it becomes useless in the trajectory optimization process. Also note that the the value used at iteration 2 is the average between the values at the first and the last iterations.

It is now possible to tackle the *while* loop itself, in this part of the algorithm the following operations are performed:

1. Satellites that require a new trajectory, with the additional nonlinear inequality constraints, perform trajectory optimization.
2. The minimum distance among the involved satellites is computed, if such distance is higher than the minimum prescribed value, it is possible to perform formation reconfiguration without a shift in maneuvering times of the involved satellites and the *while* loop ends.
3. If the minimum distance among the involved satellites is lower than the minimum prescribed value, then formation reconfiguration is performed by shifting the maneuvering times of the involved satellites by means of a genetic algorithm.

4. The minimum distance among the involved satellites is computed once again, if such distance is higher than the minimum prescribed value, it is possible to perform formation reconfiguration with a shift in maneuvering times of the involved satellites and the *while* loop ends. Note that this process is embedded in the genetic algorithm, as shown in Eq. (5.3).

5. If the minimum distance among the involved satellites is lower than the minimum prescribed value but the maximum number of iterations, which has been set to 3, has been reached, the *while* loop ends.

6. If the maximum number of iterations has not been reached, all the follower satellites are considered two at a time, and, by means of a genetic algorithm, it is checked if a formation reconfiguration between the two considered satellites is possible. If formation reconfiguration is not possible, the involved satellites will compute a new trajectory in the following iteration with an increased value of $d_{min,\boldsymbol{Sat_i}-\boldsymbol{x_j}}$.

Note that the last step of the *while* loop is performed, since computing new trajectories for some satellites might give rise to issues that with previous trajectories would not come up.

It is now possible to present an example that clarifies the algorithm: four satellites are considered, satellite 1, satellite 2, satellite 3, and satellite 4. The first part of the algorithm fails in finding a possible solution to the formation reconfiguration problem, therefore the advanced formation reconfiguration is performed. The following result is obtained when all the satellites are considered two at a time, and it is checked, by means of a genetic algorithm, if a formation reconfiguration between the two considered satellites is possible:

	Satellite 1	**Satellite 2**	**Satellite 3**	**Satellite 4**
Satellite 1	-	IMPOSSIBLE	IMPOSSIBLE	POSSIBLE
Satellite 2	IMPOSSIBLE	-	POSSIBLE	POSSIBLE
Satellite 3	IMPOSSIBLE	POSSIBLE	-	IMPOSSIBLE
Satellite 4	POSSIBLE	POSSIBLE	IMPOSSIBLE	-

Table 5.1: Example, Iteration 0.

This means that all the satellites require a new trajectory, in particular:

- Satellite 1 requires additional constraints with respect to satellite 2 and satellite 3; there is no need to add constraints with respect to satellite 4.

- Satellite 2 requires additional constraints only with respect to satellite 1; there is no need to add constraints with respect to satellite 3 and satellite 4.
- Satellite 3 requires additional constraints with respect to satellite 1 and satellite 4; there is no need to add constraints with respect to satellite 2.
- Satellite 4 requires additional constraints only with respect to satellite 3; there is no need to add constraints with respect to satellite 1 and satellite 2.

After computing the minimum distances, $d_{min,\boldsymbol{Sat_i - x_j}}$, the while loop is performed; in the first iteration a new trajectory is computed for all the satellites, but formation reconfiguration is still not possible. At the end of the first iteration all the satellites are considered two at a time and it is checked, by means of a genetic algorithm, if a formation reconfiguration between the two considered satellites is possible. The following result is obtained:

	Satellite 1	**Satellite 2**	**Satellite 3**	**Satellite 4**
Satellite 1	-	POSSIBLE	POSSIBLE	POSSIBLE
Satellite 2	POSSIBLE	-	POSSIBLE	POSSIBLE
Satellite 3	POSSIBLE	POSSIBLE	-	IMPOSSIBLE
Satellite 4	POSSIBLE	POSSIBLE	IMPOSSIBLE	-

Table 5.2: Example, Iteration 1.

This means that satellite 1 and satellite 2 solved their issues and no issues arose with the other satellites; it is not necessary to compute a new trajectory for such satellites in the following iteration. Satellite 3 solved the issue with satellite 1 but not with satellite 4. Instead, nothing changed for satellite 4.

After the end of the second iteration, still no formation reconfiguration is possible and only satellite 3 and 4 require a new trajectory. Finally, at the end of the third, and last, iteration a possible formation reconfiguration strategy is found.

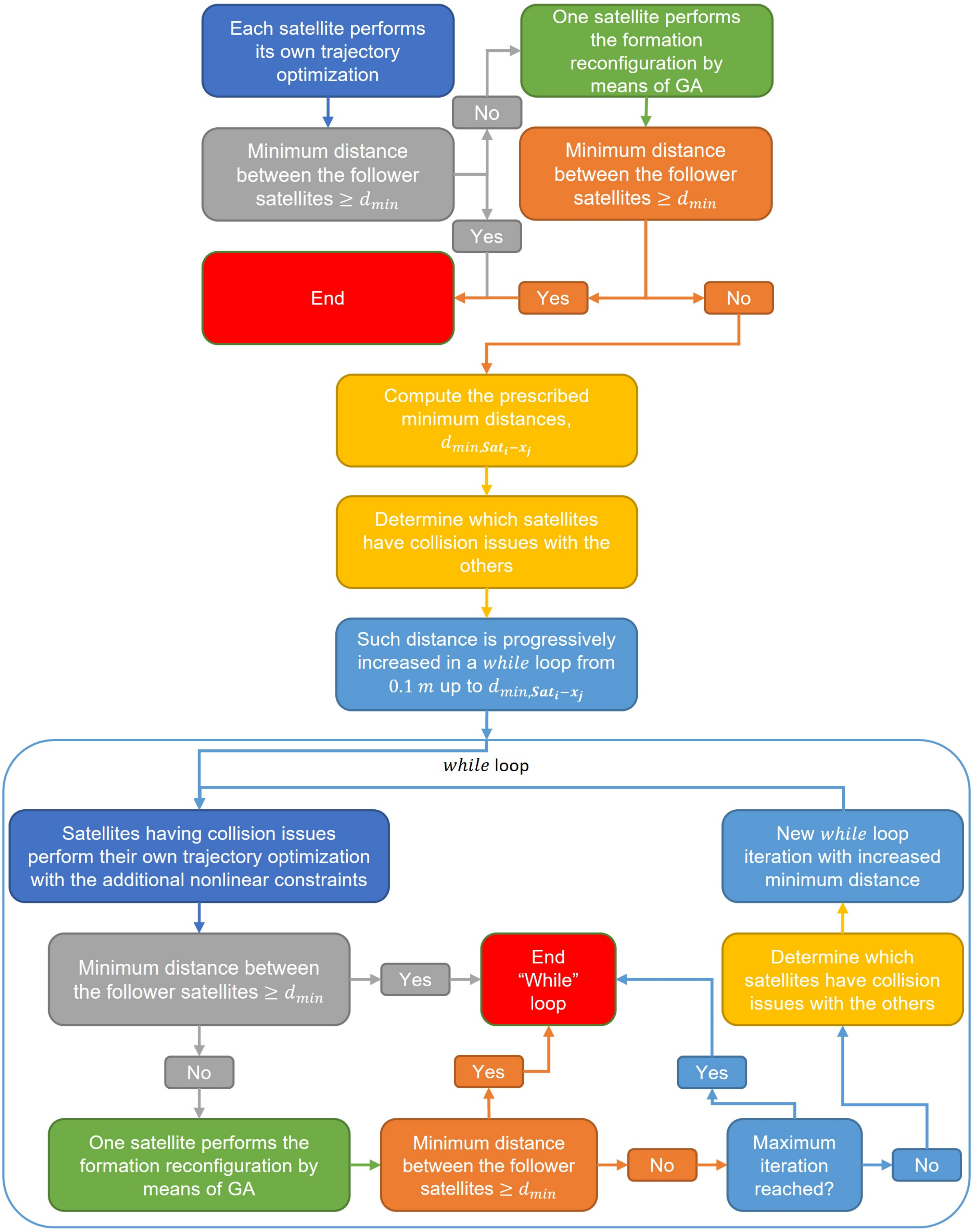
Each satellite performs its own trajectory optimization
One satellite performs the formation reconfiguration by means of GA
No
Minimum distance between the follower satellites ≥ d_{min}
Minimum distance between the follower satellites ≥ d_{min}
Yes
End
Yes
No
Compute the prescribed minimum distances, $d_{min,Sat_i - x_j}$
Determine which satellites have collision issues with the others
Such distance is progressively increased in a *while* loop from $0.1\ m$ up to $d_{min,Sat_i - x_j}$
while loop
Satellites having collision issues perform their own trajectory optimization with the additional nonlinear constraints
New *while* loop iteration with increased minimum distance
Minimum distance between the follower satellites ≥ d_{min}
Yes
End "While" loop
Determine which satellites have collision issues with the others
No
Yes
Yes
One satellite performs the formation reconfiguration by means of GA
Minimum distance between the follower satellites ≥ d_{min}
No
Maximum iteration reached?
No

Figure 5.5: Extended Workflow of the Advanced Formation Reconfiguration Algorithm.

5.4. Testing of the Developed Algorithm

The developed algorithm has been tested in two different scenarios: the first one considers the satellites reported in Table 5.3, while the second one the ones reported in Table 5.4. The considered satellites are the ones that have already been used in Sec. 4.1, while the two scenarios are the ones found in [15]. In particular, in the first scenario the follower satellites move from a planar configuration to another planar one; instead, in the second scenario the follower satellites move from a tetrahedral configuration to another tetrahedral one.

	Initial Conditions - $[m,\ m/s]$	**Final Conditions - $[m,\ m/s]$**
Satellite 1	$[0\quad 2\quad 0\quad 0\quad 0\quad 0]$	$[0\quad -4\quad 0\quad 0\quad 0\quad 0]$
Satellite 2	$[0\quad 4\quad 0\quad 0\quad 0\quad 0]$	$[0\quad -2\quad 0\quad 0\quad 0\quad 0]$
Satellite 3	$[0\quad 2\quad 3\quad 0\quad 0\quad 0]$	$[0\quad -2\quad -3\quad 0\quad 0\quad 0]$

Table 5.3: Boundary Conditions for the Satellites of Scenario 1.

	Initial Conditions - $[m,\ m/s]$	**Final Conditions - $[m,\ m/s]$**
Satellite 4	$[2\quad -3\quad 0\quad 0\quad 0\quad 0]$	$[-2\quad 2\quad 0\quad 0\quad 0\quad 0]$
Satellite 5	$[-1.5\quad -3\quad 3\quad 0\quad 0\quad 0]$	$[1\quad 2\quad -1.5\quad 0\quad 0\quad 0]$
Satellite 6	$[-1.5\quad -3\quad -3\quad 0\quad 0\quad 0]$	$[1\quad 2\quad 1.5\quad 0\quad 0\quad 0]$

Table 5.4: Boundary Conditions for the Satellites of Scenario 2.

The conditions for the formation reconfiguration are the same as the ones used in Sec. 4.1; following the most relevant ones:

- The leader satellite is on a geostationary orbit.
- The minimum distance between the leader and the follower satellite is 1.6 m.
- The maximum thrust acceleration is $6 \times 10^{-3}\ m/s^2$.
- The time of flight is 300 s.
- The order of the Bézier series for each coordinate is $\boldsymbol{n} = [12\quad 12\quad 16]$.

5.4.1. Planar Configuration to Planar Configuration

In Table 5.5 are reported the results of this reconfiguration and are compared with the data available from literature [15].

Note that the computation time is obtained by assuming that the trajectory optimization is simultaneously performed by each satellite.

It can be noted that the obtained ΔV is slightly lower than the one obtained in literature while the computation time is approximately half than the one obtained in literature. The total time of flight, instead, is higher due to the idea behind the implemented algorithm. It is worth noting that the main reason for such a low computation time is that for this scenario the advanced formation reconfiguration algorithm was not necessary; indeed, by shifting the initial maneuvering times it was already possible to obtain a feasible formation reconfiguration with the trajectories already computed in Sec. 4.1.

	Literature	Implemented Algorithm
ΔV Sat 1 - $[m/s]$	0.081	0.080
ΔV Sat 2 - $[m/s]$	0.081	0.080
ΔV Sat 3 - $[m/s]$	0.083	0.082
Total ΔV - $[m/s]$	0.246	0.242
Time of Flight - $[s]$	299.952	407.828
Computation Time - $[s]$	138.848	68.045

Table 5.5: Results of Scenario 1.

Figure 5.6 shows the distances between the follower satellites and the distances between the leader and the follower satellites in two cases: in the top part such distances are the ones obtained in case all the satellites perform the reconfiguration maneuvers at the same time; in the bottom part, instead, are the results after the formation reconfiguration algorithm. Finally, Figure 5.7 shows the final trajectories for all the involved satellites.

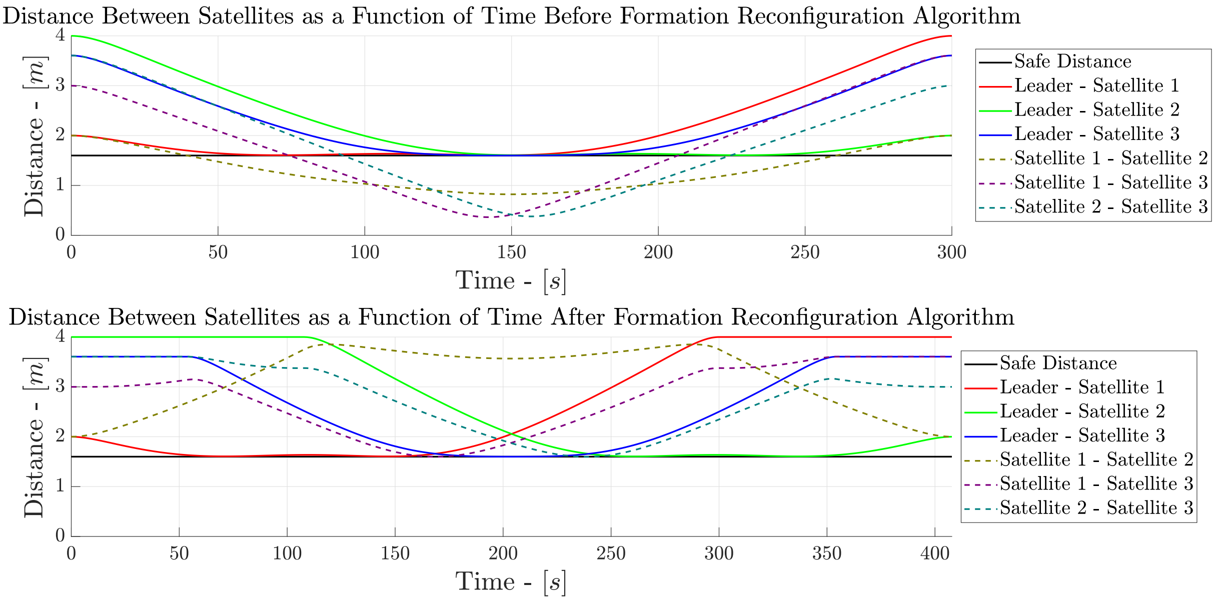

Figure 5.6: Distance Between Satellites as a Function of Time When Reconfiguration Maneuvers are Performed at the Same Time (Top) and After Formation Reconfiguration Algorithm (Bottom) for the Scenario 1.

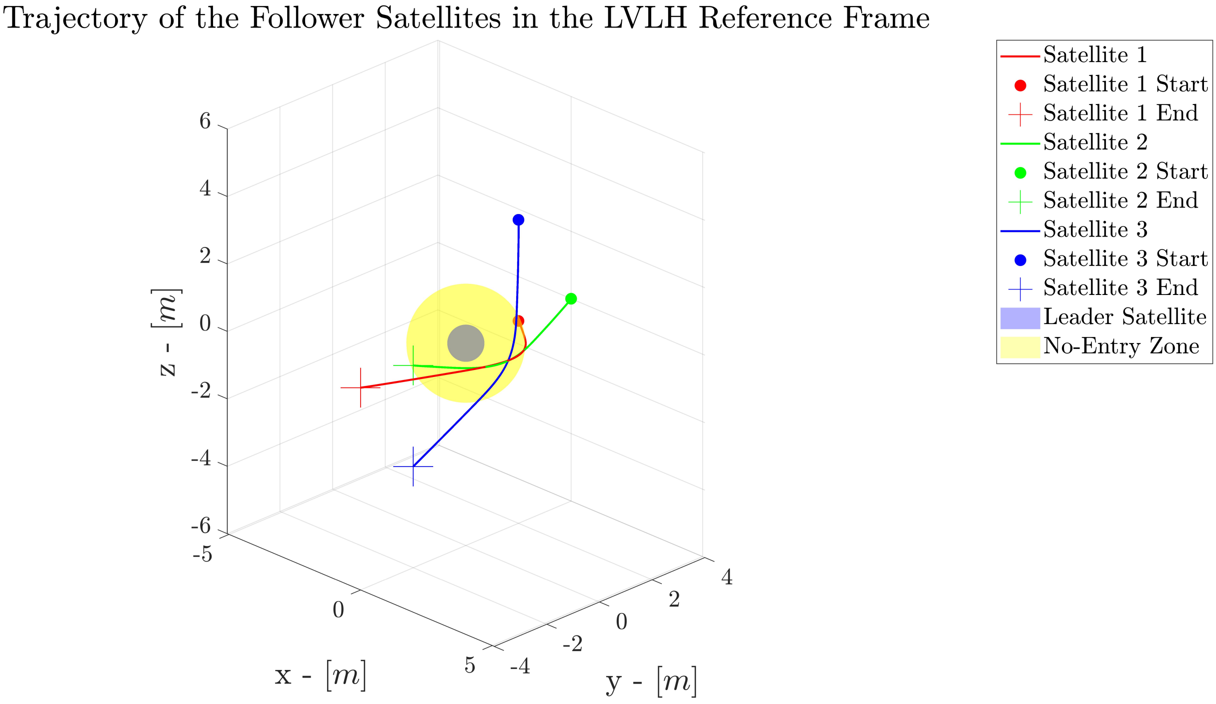

Figure 5.7: Trajectory of the Satellites in the LVLH Reference Frame for the Scenario 1.

5.4.2. Tetrahedral to Tetrahedral Configuration

In Table 5.6 are reported the results of this reconfiguration and are compared with the data available from literature [15].

Note that the computation time is obtained by assuming that the trajectory optimization is simultaneously performed by each satellite.

It can be noted that, similarly to the previous scenario, also in this case the obtained ΔV is slightly lower than the one obtained in literature and the total time of flight is higher. On the other hand, the computation time is higher than the one obtained in literature. The main reason for the higher computation time is that for this scenario the advanced formation reconfiguration algorithm was necessary; indeed, shifting the initial maneuvering times was not enough to obtain a feasible formation reconfiguration and it was necessary to compute new trajectories by including additional constraints.

	Literature	**Implemented Algorithm**
$\boldsymbol{\Delta V}$ **Sat 4 - [$\boldsymbol{m/s}$]**	0.076	0.076
$\boldsymbol{\Delta V}$ **Sat 5 - [$\boldsymbol{m/s}$]**	0.079	0.079
$\boldsymbol{\Delta V}$ **Sat 6 - [$\boldsymbol{m/s}$]**	0.083	0.079
Total $\boldsymbol{\Delta V}$ **- [$\boldsymbol{m/s}$]**	0.238	0.234
Time of Flight - [$\boldsymbol{s}$]	299.992	525.711
Computation Time - [$\boldsymbol{s}$]	233.338	406.101

Table 5.6: Results of Scenario 2.

Figure 5.8 shows the distances between the follower satellites and the distances between the leader and the follower satellites in two cases: in the top part such distances are the ones obtained in case all the satellites perform the reconfiguration maneuvers at the same time; in the bottom part, instead, are the results after the advanced formation reconfiguration algorithm. Finally, Figure 5.9 shows the trajectories for all the involved satellites, in particular: the initial trajectories are on the left side, they are obtained without the introduction of the additional constraints; while on the right side are the final trajectories, obtained after the execution of the advanced formation reconfiguration algorithm. In particular, it can be noted that the trajectories of satellites 5 and 6 have been changed in order to make the formation reconfiguration possible.

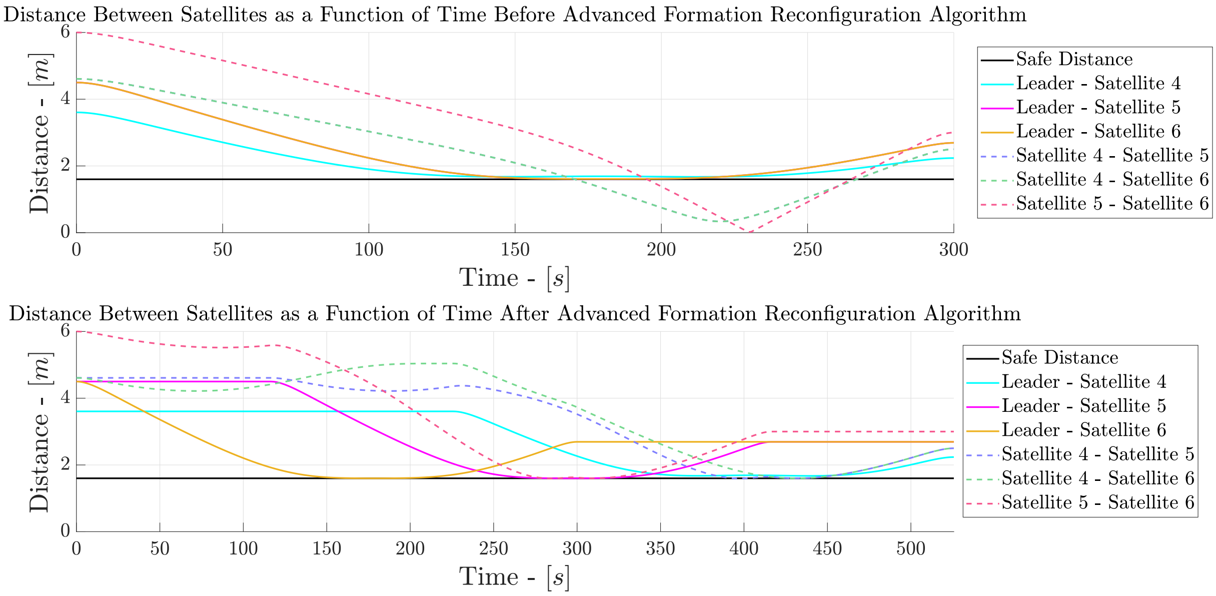

Figure 5.8: Distance Between Satellites as a Function of Time When Reconfiguration Maneuvers are Performed at the Same Time (Top) and After Advanced Formation Reconfiguration Algorithm (Bottom) for the Scenario 2.

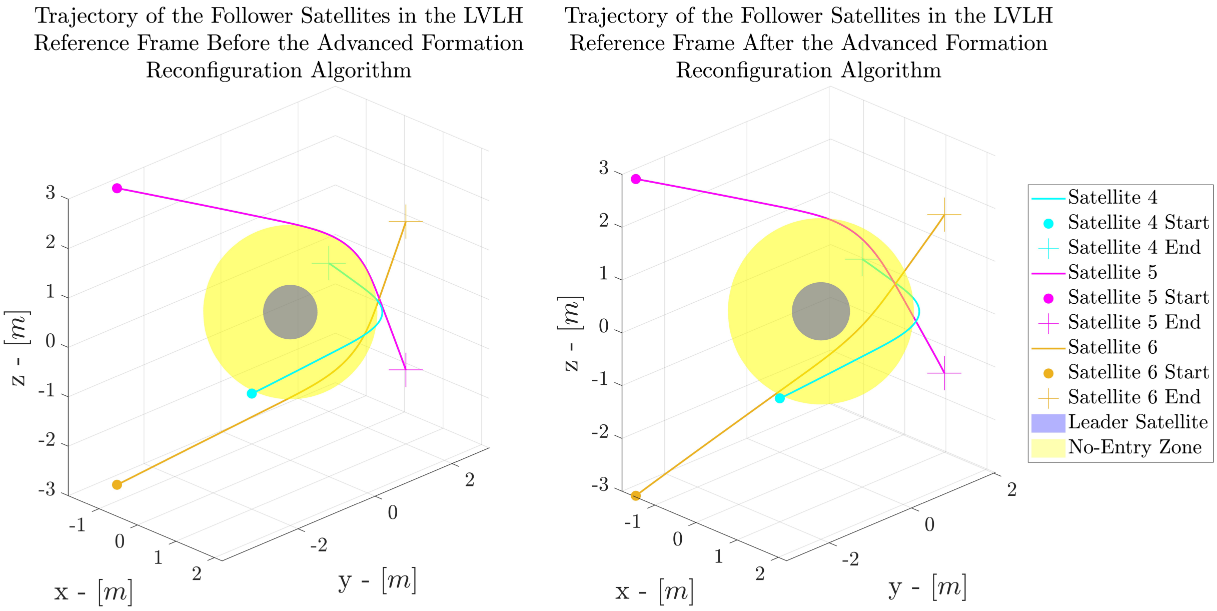

Figure 5.9: Trajectory of the Satellites in the LVLH Reference Frame Before (Left) and After (Right) Advanced Formation Reconfiguration Algorithm for the Scenario 2.

5.5. Final Comments About the Developed Algorithm

It is worth spending a few more words about the developed algorithm. The strengths of such algorithm are that the computation time is quite low, especially when there is no need to compute the trajectories multiple times, the total ΔV is still optimal since each satellite has an optimized trajectory, and it has a decentralized approach, therefore the computational power of all the platforms can be exploited. Additionally, with the developed algorithm it could also be possible to deal with more complex problems in terms of number of involved satellites; indeed, it was also possible to solve the formation reconfiguration with all six satellites at the same time. The downside of increasing too much the number of involved satellites is that the algorithm's performance, in terms of computation time, decreases rapidly due to the complexity of the problem. On the other hand, it is worth to note that this algorithm has been developed for satellites flying in close formation, that is, for satellites flying within 15 m from the leader one, and, in such scenario, it is unlikely that the number of involved satellites will be greater than four.

There are three additional important considerations to make: the first one is that the comparison, in terms of ΔV, that has been performed in Sec. 5.4 with respect to the literature findings must be correctly contextualized; such comparison is only related to the quality, in terms of ΔV, of the reconfiguration maneuver but the two results cannot quite be compared directly. Indeed, to correctly compare the two results an equal time window would be required. The second and third considerations relate to the conditions of the satellites before and after having performed the reconfiguration maneuver; as it can be seen in Figure 5.6 and in Figure 5.8, the satellites remain fixed in either the initial and/or the final positions depending on their starting maneuvering times. To do that an additional control is necessary; such control is negligible in the considered scenarios due to the fact that the initial and final velocities, in the LVLH reference frame, are equal to zero, but this might not be the case when such velocities are different from zero. This leads to the final remark: the developed tool works well when a zero velocity at the boundaries is assumed and, in case this condition cannot be imposed, not only a poor solution might be obtained, but also errors could arise.

6 | Conclusions and Future Developments

There are two main results from this work: the first one concerns trajectory optimization while the second is related to formation reconfiguration.

Considering the former, trajectory optimization has been performed by means of a shape-based method; in particular, the Bèzier series has been exploited for the solution of such problem. The developed algorithm has been compared to results from literature and can be considered successful. Additionally, a study on how the Bézier series affects the obtained results has been performed. The are two main lessons that have been learned from such an analysis: the first one is that, even increasing the order of the Bézier series, the obtained result is limited by the mathematical nature of the shape function; indeed, from the mathematical point of view there are limitations in terms of the possible shapes that such function can achieve. The second lesson is that a good trade-off between the obtained ΔV and computation time is achieved when the order of the Bézier series is equal to 10. Concerning shape-based methods for the fuel-optimal trajectory optimization problem, possible improvements could be achieved by exploiting a different shape function. Ideal shape functions for such problems would be those that allow to resemble a bang-bang solution as close as possible; therefore, when considering the control, the obtained solution must have two features: two peaks in the initial and final parts of the trajectory and, if needed, a lower control in the central part in order to correct the trajectory. Additionally, the shape function shall be capable of abruptly vary from a maximum value to zero in order to make the control as efficient as possible.

Considering now formation reconfiguration, a number of observations can be done. Firstly, this optimization problem has been tackled by means of a decentralized approach; this choice is dictated by the goal of achieving a higher autonomy with respect to ground stations for satellites flying in formation. Indeed, as the number of missions that exploits formation flying increases, a higher autonomy will not only be desirable, but also mandatory. Concerning the developed algorithm, it has been proved to be working correctly when tested with data available from literature. Nonetheless it has some limitations from the application point of view, the two major ones are related to the conditions of the satellites: the leader satellite must be on a circular orbit and the velocities of the follower satellites, in the LVLH reference frame, must be equal to zero at the boundaries. For these reasons, future works could aim at improving such algorithm to be used in a broader range of scenarios such as different conditions of the leader and follower satellites or different relative distances between the involved satellites.

Bibliography

[1] Jonas Radtke, Enrico Stoll, Hugh Lewis, and Benjamin Bastida Virgili. The impact of the increase in small satellite launch traffic on the long-term evolution of the space debris environment. In T. Flohrer and F. Schmitz, editors, *Proceedings of the 7th European Conference on Space Debris*, volume 7. The European Space Agency (ESA), 2017.

[2] Owen Brown, Paul Eremenko, and Paul Collopy. Value-centric design methodologies for fractionated spacecraft: Progress summary from phase i of the darpa system f6 program. In *AIAA SPACE 2009 Conference & Exposition*, Pasadena, California, Sep 2009. American Institute of Aeronautics and Astronautics.

[3] Jesse Leitner. Formation flying-the future of remote sensing from space. In *18th International Symposium on Space Flight Dynamics*, volume 548, page 621, 2004.

[4] Saptarshi Bandyopadhyay, Rebecca Foust, Giri P. Subramanian, Soon-Jo Chung, and Fred Y. Hadaegh. Review of formation flying and constellation missions using nanosatellites. *Journal of Spacecraft and Rockets*, 53(3):567–578, May 2016.

[5] Francesca Scala, Gabriella Gaias, Camilla Colombo, and Manuel Martín-Neira. Design of optimal low-thrust manoeuvres for remote sensing multi-satellite formation flying in low earth orbit. *Advances in Space Research*, 68(11):4359–4378, 2021.

[6] Forrester T. Johnson. Approximate finite-thrust trajectory optimization. *AIAA Journal*, 7(6):993–997, 1969.

[7] J. Vlassenbroeck and R. Van Dooren. A chebyshev technique for solving nonlinear optimal control problems. *IEEE Transactions on Automatic Control*, 33(4):333–340, Apr 1988.

[8] Anastassios E. Petropoulos and James M. Longuski. Shape-based algorithm for the automated design of low-thrust, gravity assist trajectories. *Journal of Spacecraft and Rockets*, 41(5):787–796, 2004.

[9] Massimiliano Vasile. Preliminary design of low-thrust multiple gravity assist trajectories. *Journal of Spacecraft and Rockets*, 43, 09 2006.

[10] Bradley Wall. *Shape-Based Approximation Method for Low-Thrust Trajectory Optimization.* American Institute of Aeronautics and Astronautics, Jun 2008.

[11] Ehsan Taheri and Ossama Abdelkhalik. Shape based approximation of constrained low-thrust space trajectories using fourier series. *Journal of Spacecraft and Rockets*, 49(3):535–546, 2012.

[12] Ehsan Taheri and Ossama Abdelkhalik. Fast initial trajectory design for low-thrust restricted-three-body problems. *Journal of Guidance, Control, and Dynamics*, 38(11):2146–2160, 2015.

[13] Ehsan Taheri and Ossama Abdelkhalik. Initial three-dimensional low-thrust trajectory design. *Advances in Space Research*, 57(3):889–903, 2016.

[14] Mingying Huo, Giovanni Mengali, Alessandro A. Quarta, and Naiming Qi. Electric sail trajectory design with bezier curve-based shaping approach. *Aerospace Science and Technology*, 88:126–135, 2019.

[15] Zichen Fan, Mingying Huo, Song Xu, Jun Zhao, and Naiming Qi. Fast cooperative trajectory optimization for close-range satellite formation using bezier shape-based method. *IEEE Access*, 8:30918–30927, 2020.

[16] Naiming Qi, Zichen Fan, Mingying Huo, Desong Du, and Ce Zhao. Fast trajectory generation and asteroid sequence selection in multispacecraft for multiasteroid exploration. *IEEE Transactions on Cybernetics*, 52(7):6071–6082, 2022.

[17] Roberto Armellin, Mauro Massari, and Amalia Ercoli Finzi. Optimal formation flying reconfiguration and station keeping maneuvers using low thrust propulsion. In *Proceedings of the 18th International Symposium on Space Flight Dynamics (ESA SP-548)*, pages 429–434, 2004.

[18] Marco Sabatini and Giovanni B. Palmerini. Collective control of spacecraft swarms for space exploration. *Celestial Mechanics and Dynamical Astronomy*, 105(1–3):229–244, Mar 2009.

[19] Yue Hang, Guangyan Xu, Danwei Wang, and Eng Kee Poh. Comparison study of relative dynamic models for satellite formation flying. In *2008 2nd International Symposium on Systems and Control in Aerospace and Astronautics*, pages 1–6, 2008.

[20] Hartmut Prautzsch, Wolfgang Boehm, and Marco Paluszny. *Bézier and B-Spline*

Techniques. Mathematics and Visualization. Springer Berlin Heidelberg, Berlin, Heidelberg, 2002.

[21] The MathWorks Inc. Genetic Algorithm, 2023.

A | Appendix A

This appendix shows the remaining results of the trajectory optimization presented in Sec. 4.1.

Figure A.1 and Figure A.2 show the thrust acceleration history and the projection of the trajectory for the second case obtained with the HCW model. Figure A.3 shows the distance between the leader and the follower satellite as a function of time, the trajectory in the LVLH reference frame, and the velocity and the position components for the second case obtained with the HCW model.

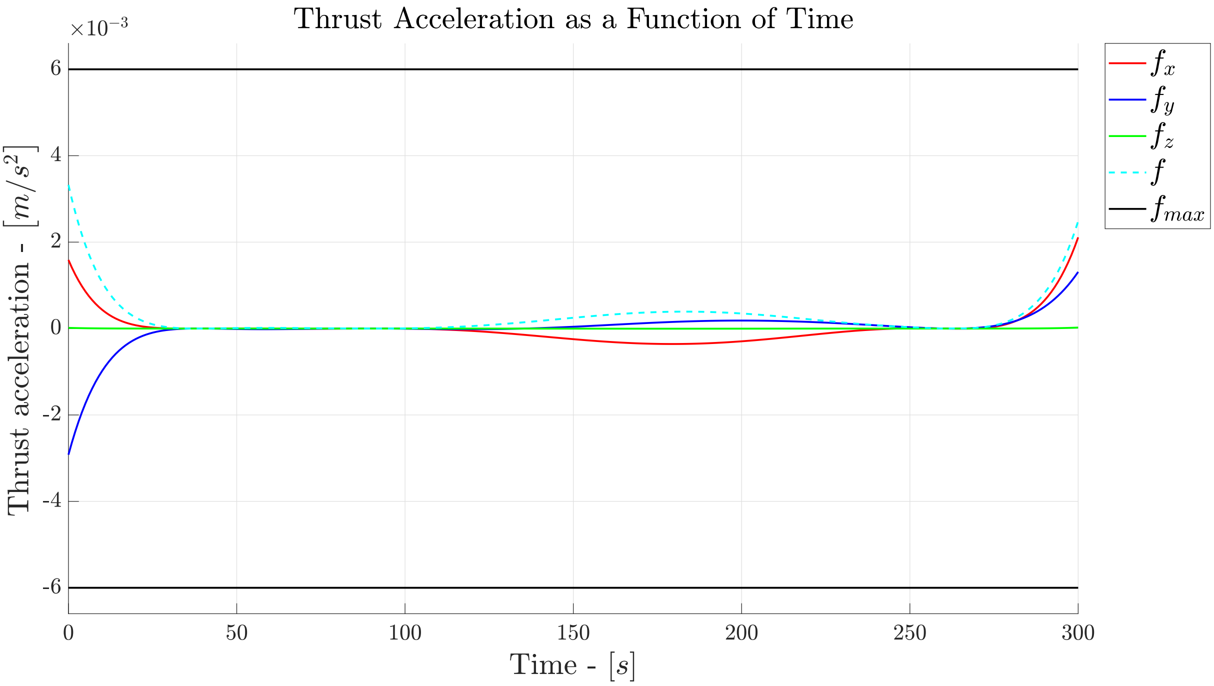

Figure A.1: Total Thrust Acceleration and Thrust Acceleration Components as a Function of Time - Case 2.

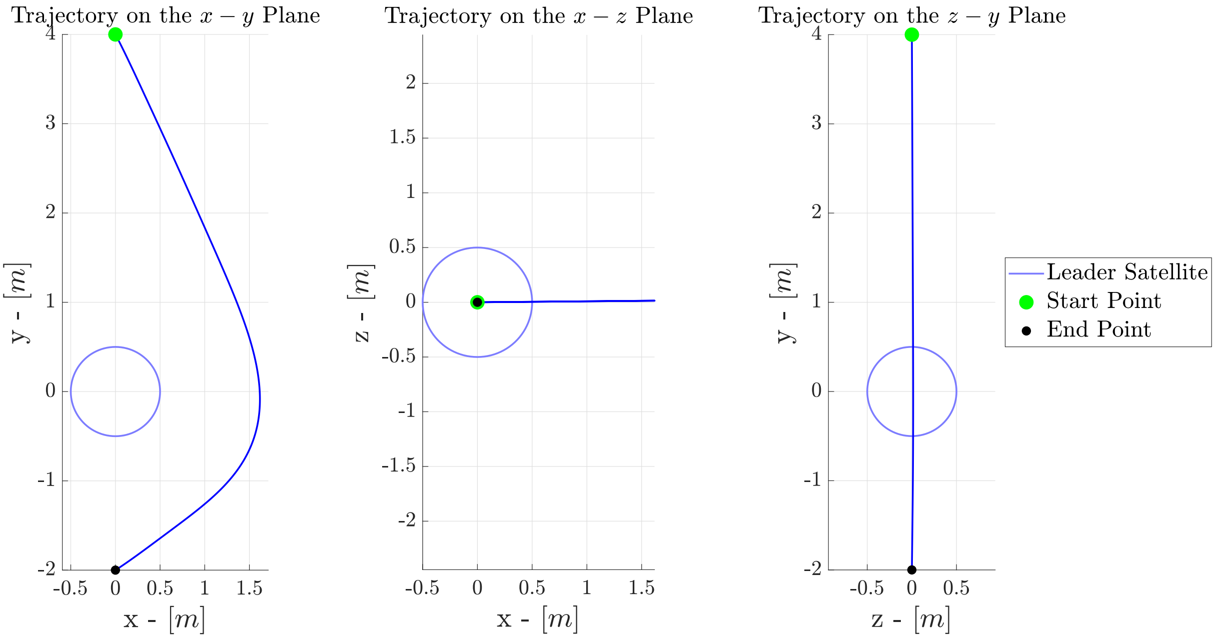

Figure A.2: Projection of the Trajectory - Case 2.

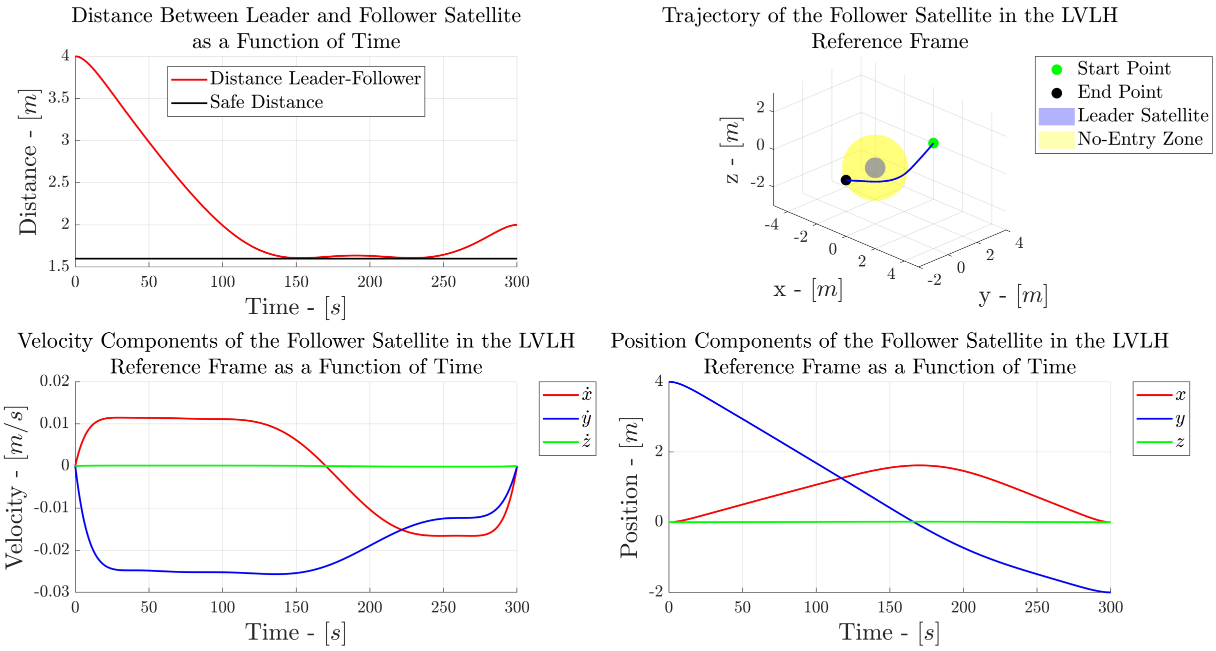

Figure A.3: Distance Between Leader and Follower Satellites as a Function of Time (Top Left), Trajectory on the LVLH Reference Frame (Top Right), Velocity Components as a Function of Time (Bottom Left), Position Components as a Function of Time (Bottom Right) - Case 2.

Figure A.4 and Figure A.5 show the thrust acceleration history and the projection of the trajectory for the third case obtained with the HCW model. Figure A.6 shows the distance between the leader and the follower satellite as a function of time, the trajectory in the LVLH reference frame, and the velocity and the position components for the third case obtained with the HCW model.

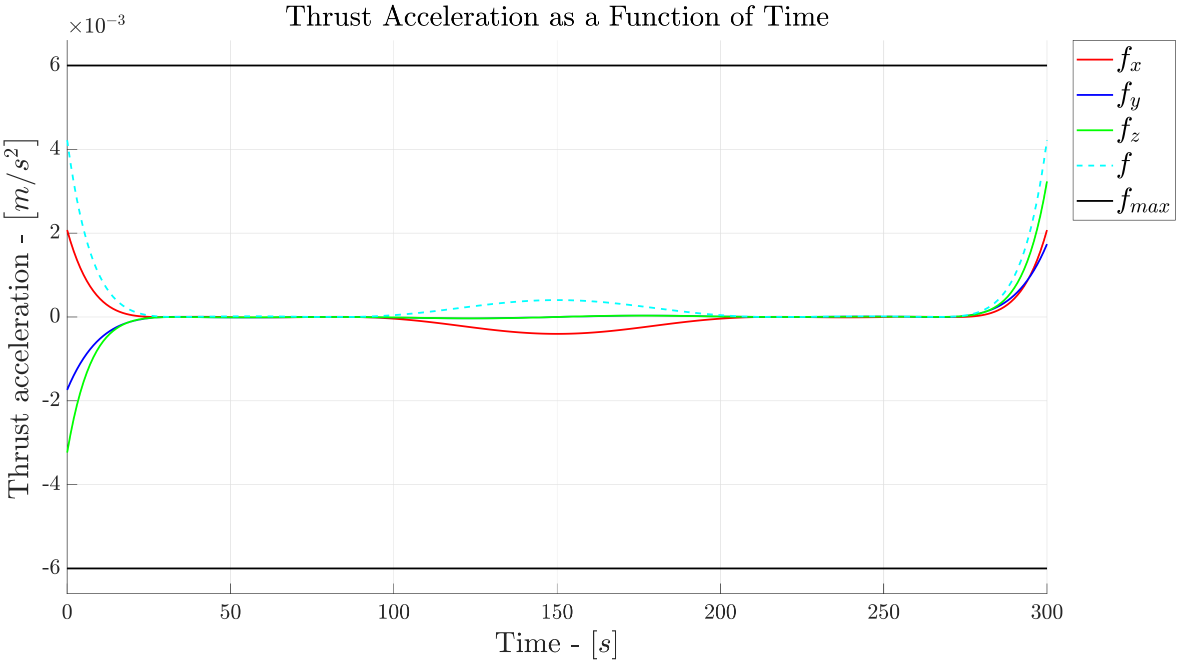

Figure A.4: Total Thrust Acceleration and Thrust Acceleration Components as a Function of Time - Case 3.

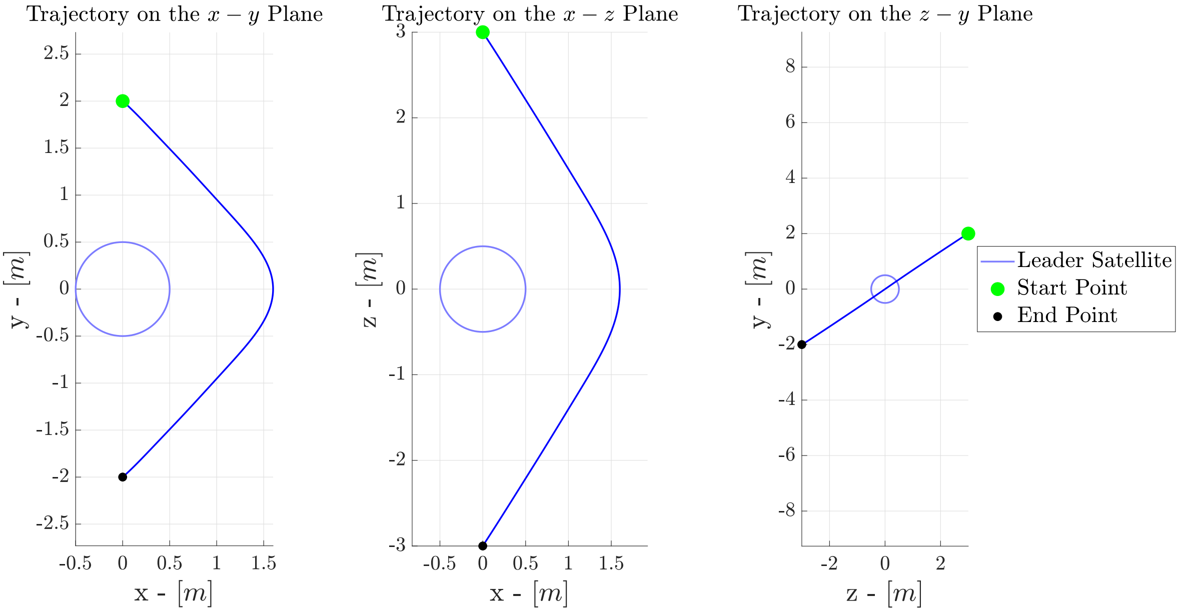

Figure A.5: Projection of the Trajectory - Case 3.

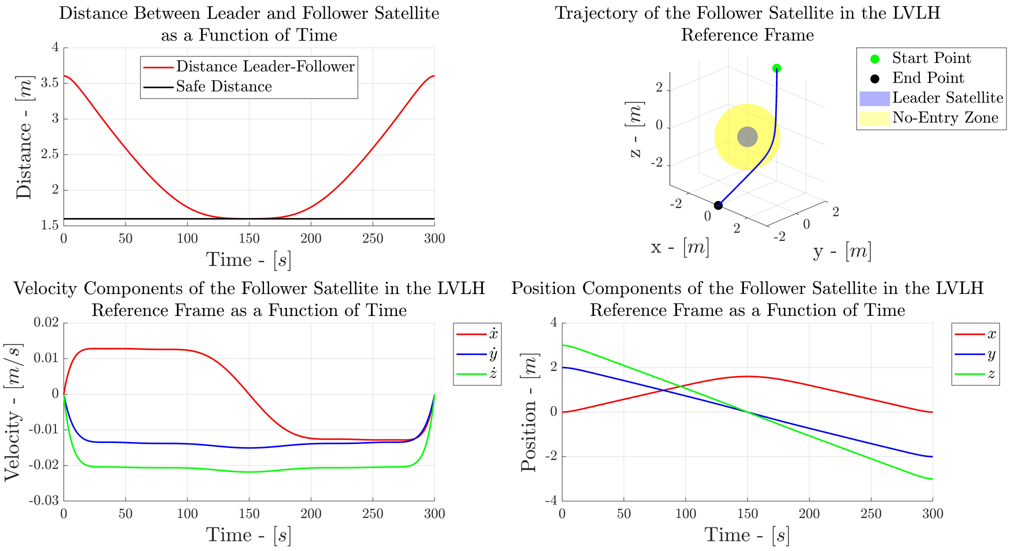

Figure A.6: Distance Between Leader and Follower Satellites as a Function of Time (Top Left), Trajectory on the LVLH Reference Frame (Top Right), Velocity Components as a Function of Time (Bottom Left), Position Components as a Function of Time (Bottom Right) - Case 3.

Figure A.7 and Figure A.8 show the thrust acceleration history and the projection of the trajectory for the fourth case obtained with the HCW model. Figure A.9 shows the distance between the leader and the follower satellite as a function of time, the trajectory in the LVLH reference frame, and the velocity and the position components for the fourth case obtained with the HCW model.

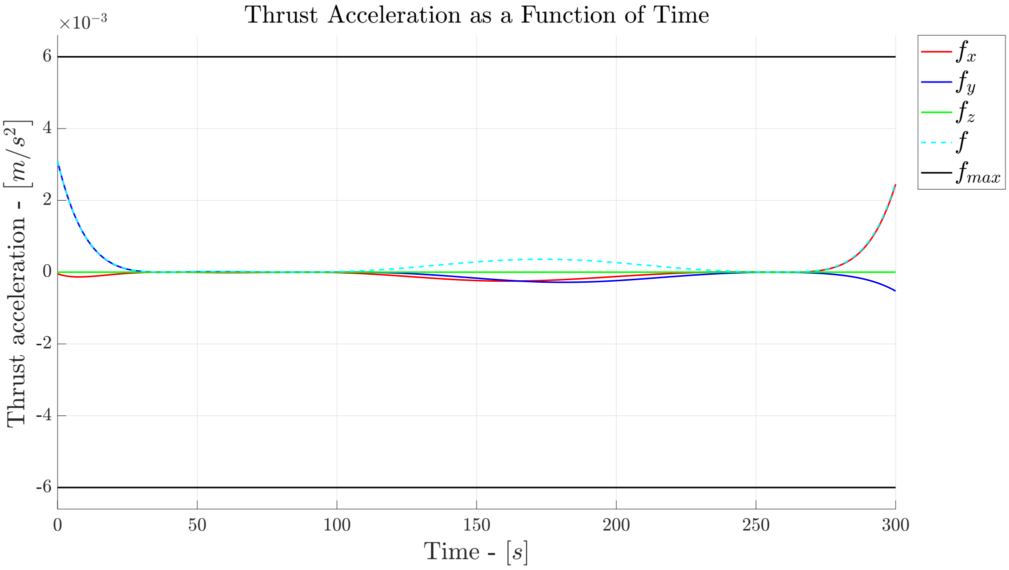

Figure A.7: Total Thrust Acceleration and Thrust Acceleration Components as a Function of Time - Case 4.

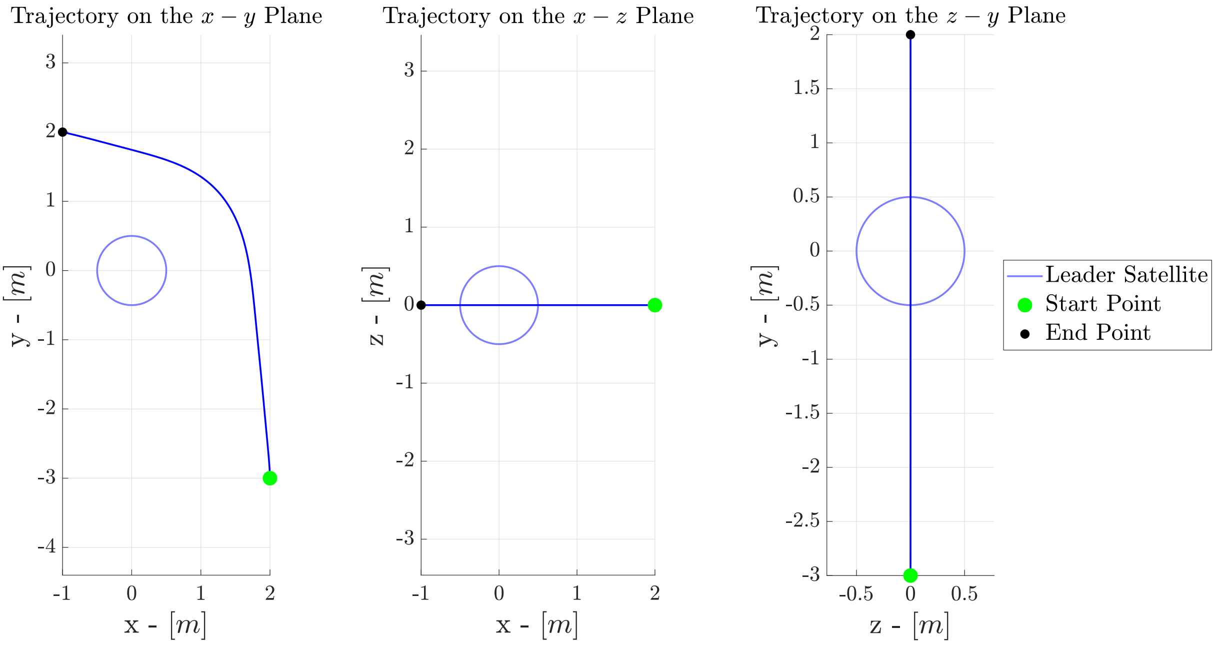

Figure A.8: Projection of the Trajectory - Case 4.

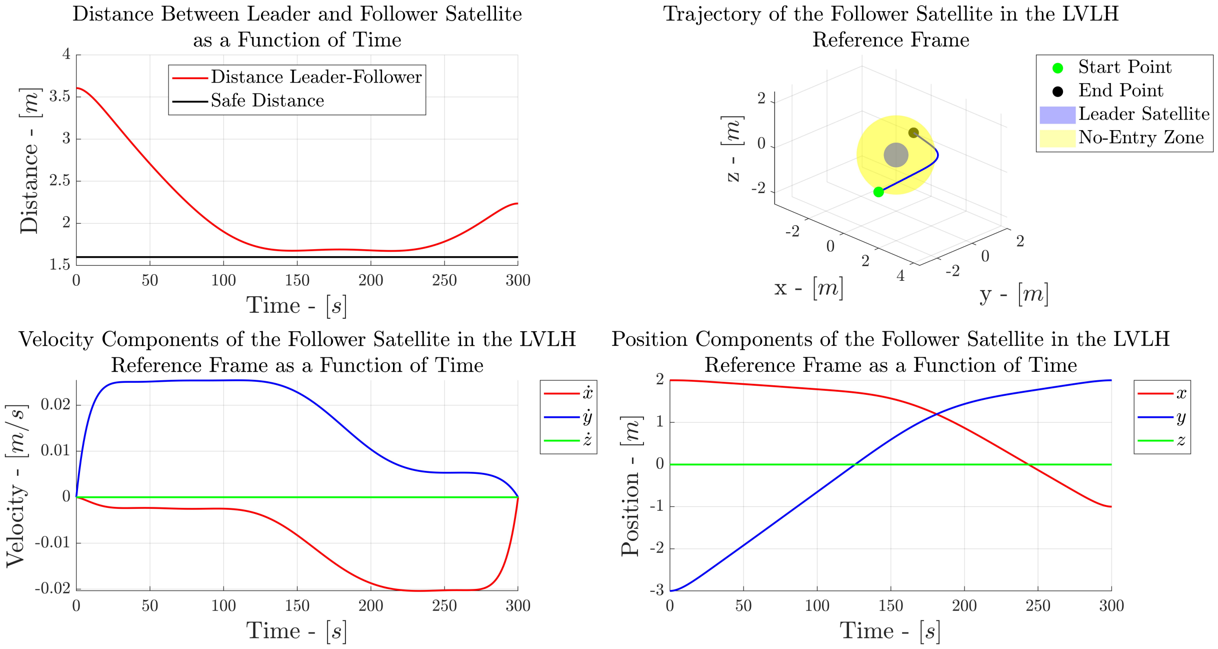

Figure A.9: Distance Between Leader and Follower Satellites as a Function of Time (Top Left), Trajectory on the LVLH Reference Frame (Top Right), Velocity Components as a Function of Time (Bottom Left), Position Components as a Function of Time (Bottom Right) - Case 4.

Figure A.10 and Figure A.11 show the thrust acceleration history and the projection of the trajectory for the fifth case obtained with the HCW model. Figure A.12 shows the distance between the leader and the follower satellite as a function of time, the trajectory in the LVLH reference frame, and the velocity and the position components for the fifth case obtained with the HCW model.

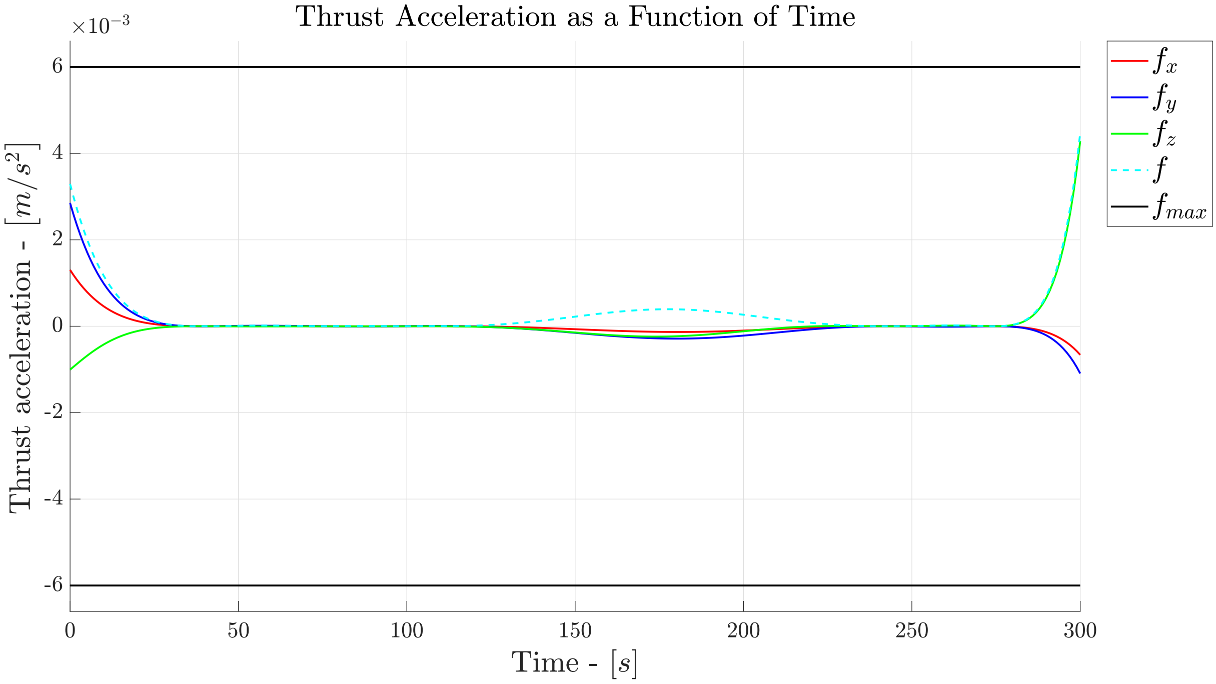

Figure A.10: Total Thrust Acceleration and Thrust Acceleration Components as a Function of Time - Case 5.

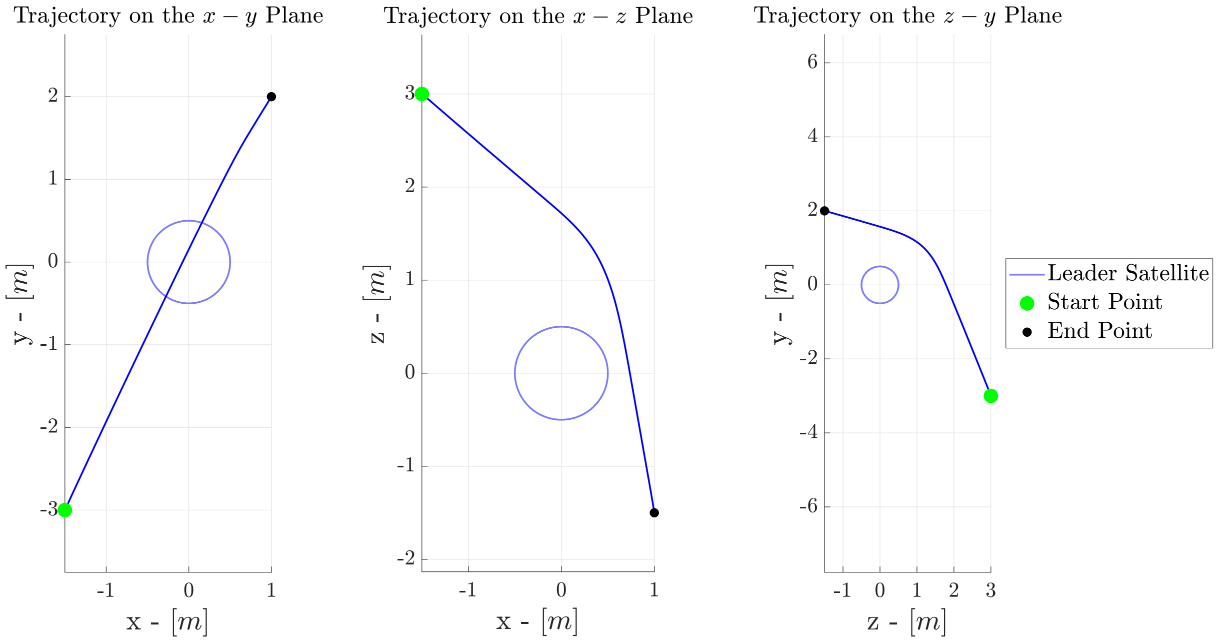

Figure A.11: Projection of the Trajectory - Case 5.

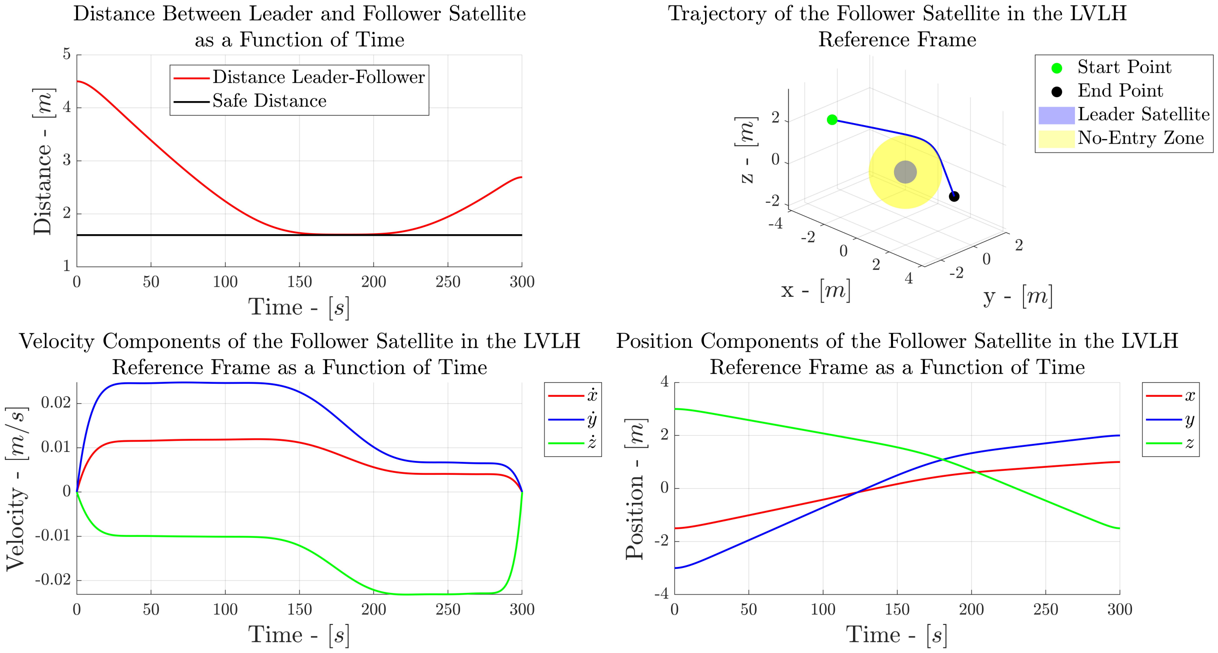

Figure A.12: Distance Between Leader and Follower Satellites as a Function of Time (Top Left), Trajectory on the LVLH Reference Frame (Top Right), Velocity Components as a Function of Time (Bottom Left), Position Components as a Function of Time (Bottom Right) - Case 5.

B | Appendix B

This appendix shows the remaining results of the study of how the trajectory optimization is affected by the order of the Bézier series presented in Sec. 4.2.

From Figure B.1 to Figure B.4 are reported the thrust acceleration components and the total thrust acceleration as a function of time, for different orders of the Bézier series for the cases 1, 3, 5, and 6.

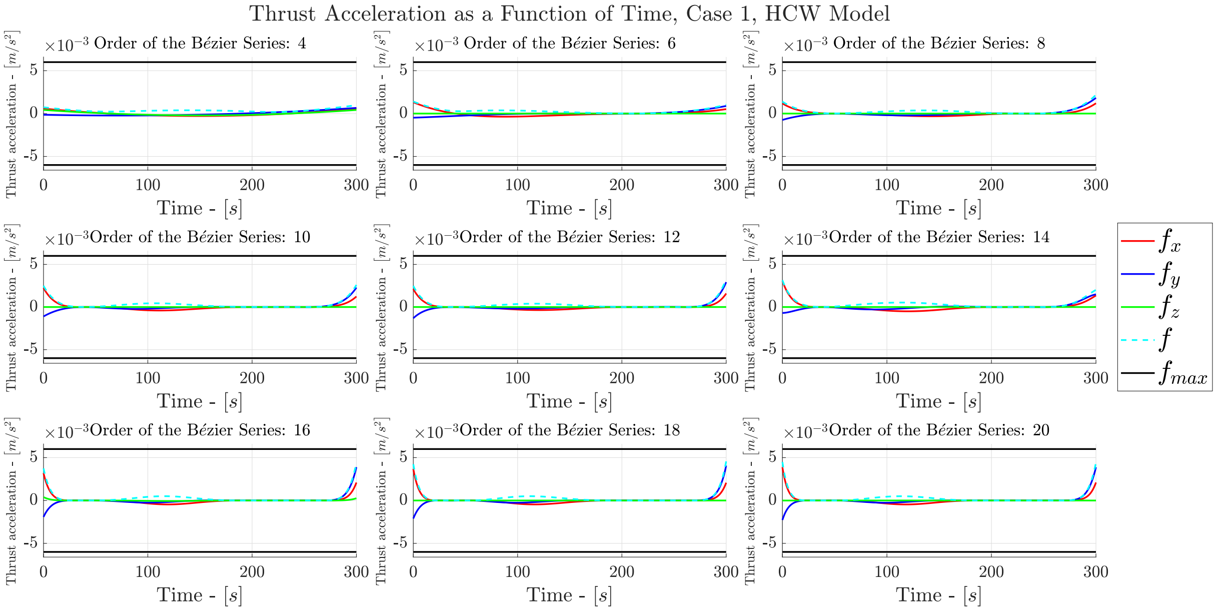

Figure B.1: Thrust Acceleration as a Function of Time for Different Orders of the Bézier Series - Case 1.

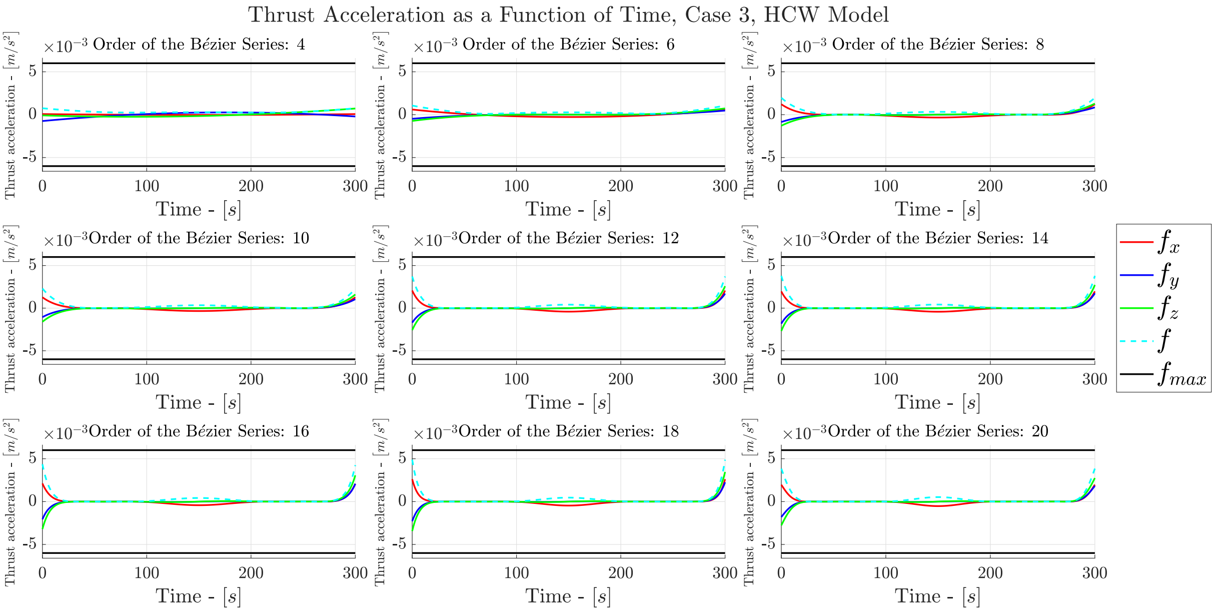

Figure B.2: Thrust Acceleration as a Function of Time for Different Orders of the Bézier Series - Case 3.

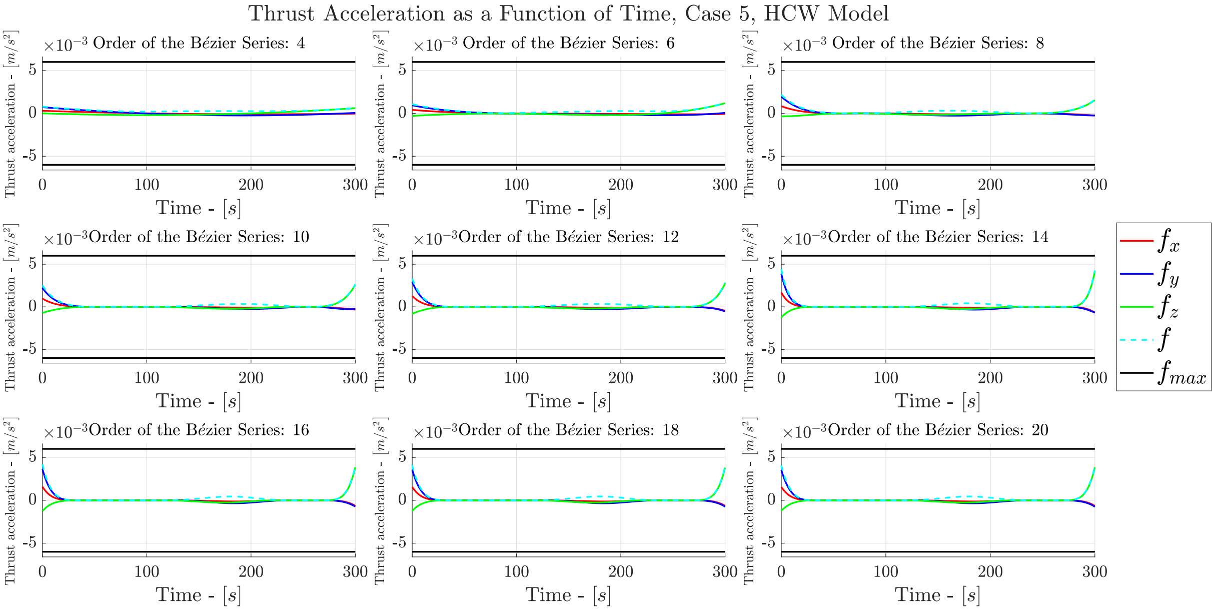

Figure B.3: Thrust Acceleration as a Function of Time for Different Orders of the Bézier Series - Case 5.

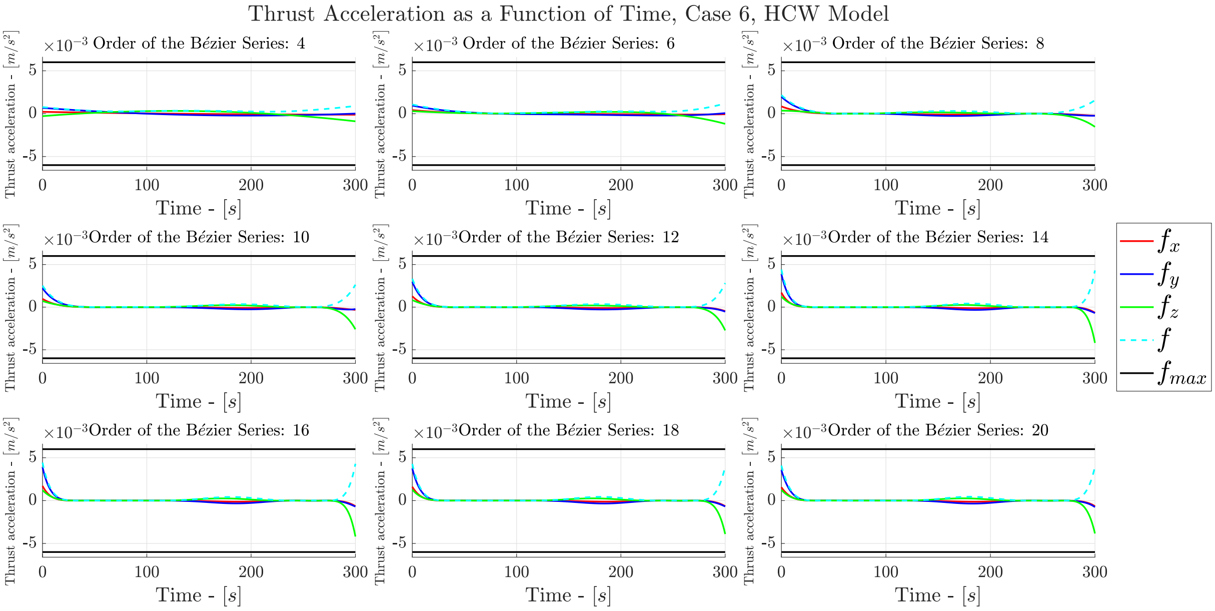
Thrust Acceleration as a Function of Time, Case 6, HCW Model

Figure B.4: Thrust Acceleration as a Function of Time for Different Orders of the Bézier Series - Case 6.

From Figure B.5 to Figure B.8 are reported the plots of Eq. 4.3 for different orders of the Bézier series for the cases 1, 3, 4, and 6.

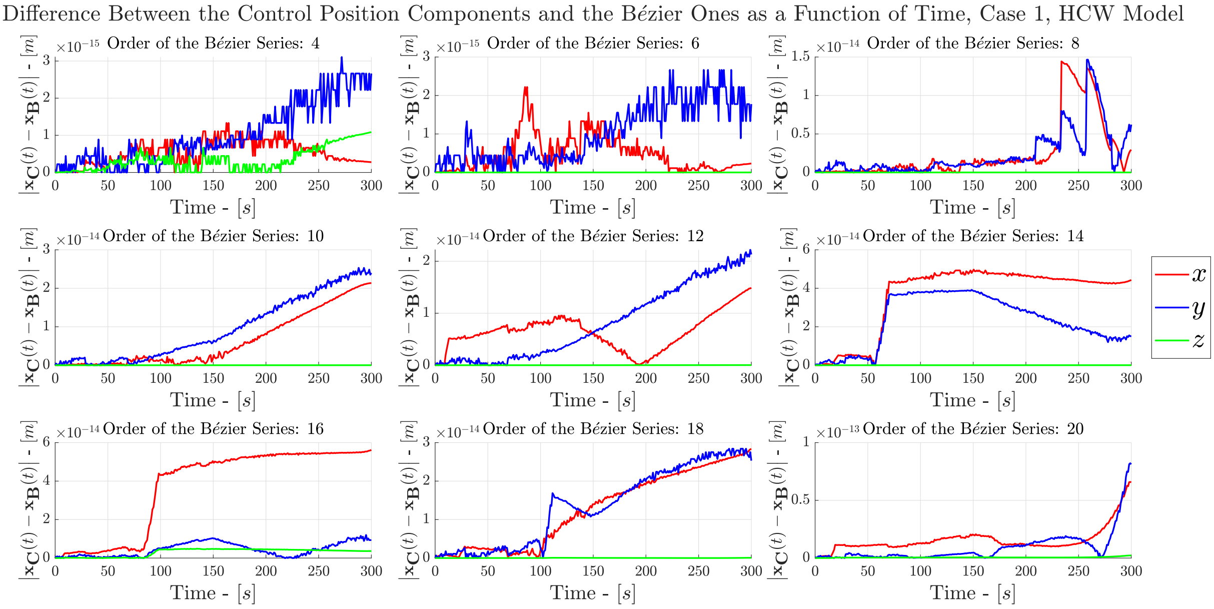
Difference Between the Control Position Components and the Bézier Ones as a Function of Time, Case 1, HCW Model

Figure B.5: $|\boldsymbol{x}_{\boldsymbol{control}}(t) - \boldsymbol{x}_{\boldsymbol{Bézier}}(t)|$ for Different Orders of the Bézier Series - Case 1.

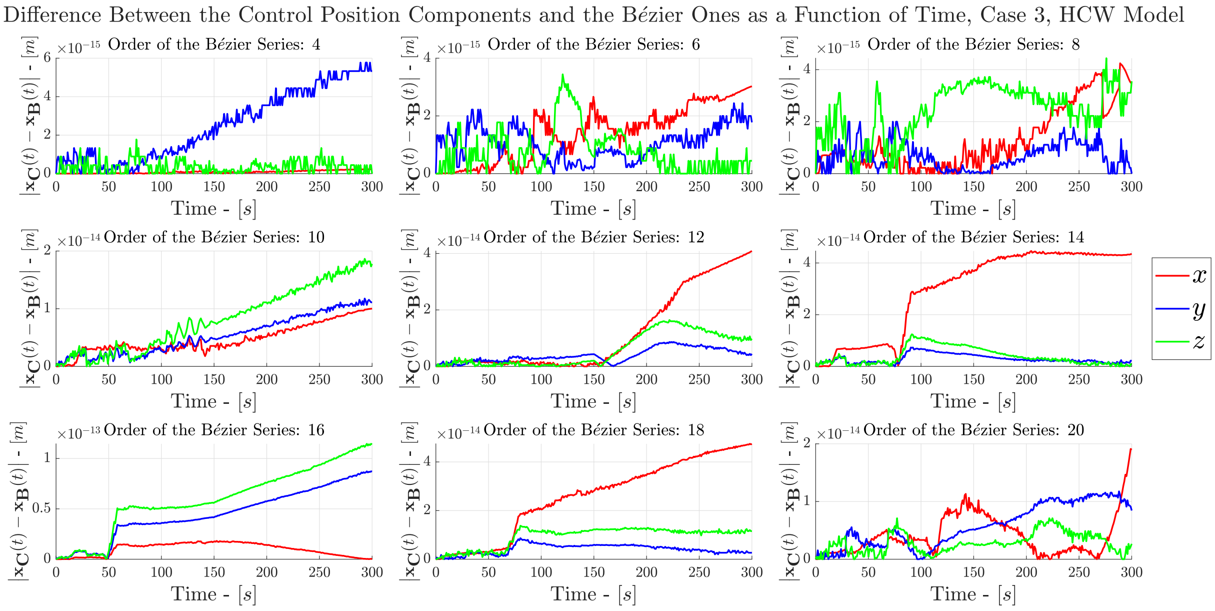

Figure B.6: $|\boldsymbol{x_{control}}(t) - \boldsymbol{x_{Bézier}}(t)|$ for Different Orders of the Bézier Series - Case 3.

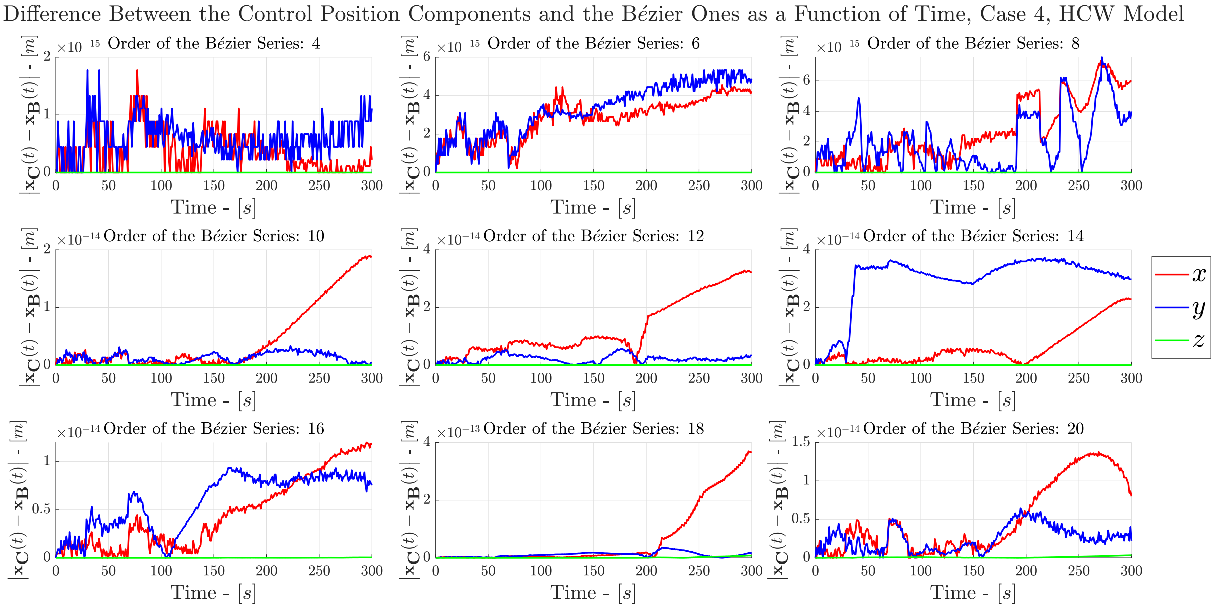

Figure B.7: $|\boldsymbol{x_{control}}(t) - \boldsymbol{x_{Bézier}}(t)|$ for Different Orders of the Bézier Series - Case 4.

Difference Between the Control Position Components and the Bézier Ones as a Function of Time, Case 6, HCW Model

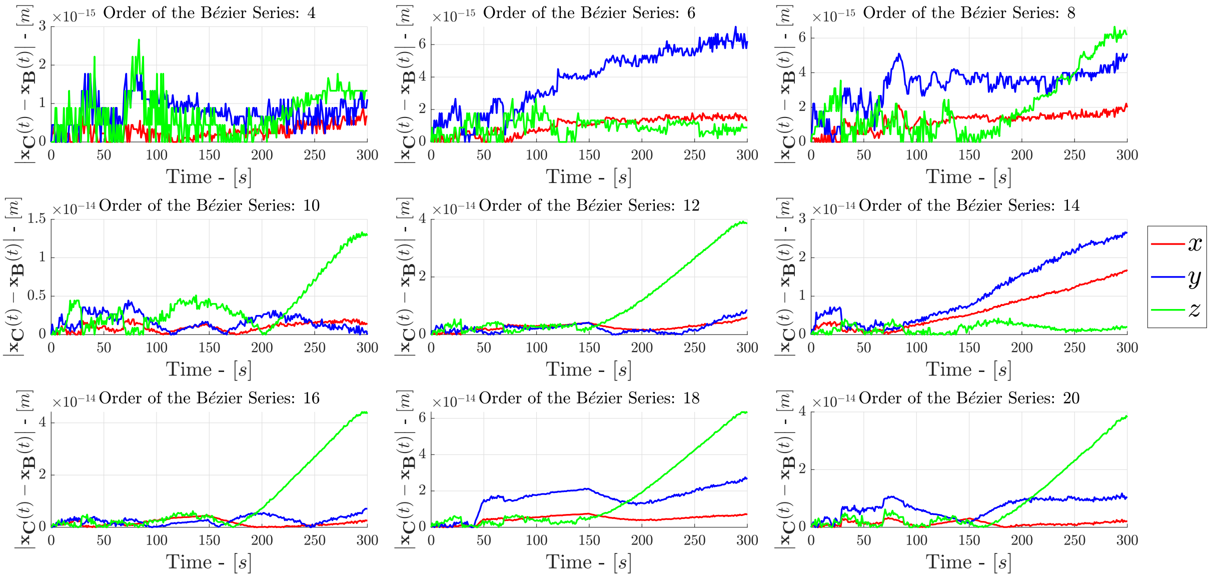

Figure B.8: $|\boldsymbol{x}_{\boldsymbol{control}}(t) - \boldsymbol{x}_{\boldsymbol{Bézier}}(t)|$ for Different Orders of the Bézier Series - Case 6.

Acknowledgements

Before ending the thesis I would like to spend a few words for all the people that helped and supported me not only in this work, but also during the journey of my university years.

I am extremely grateful to Professor Mauro Massari for giving me the opportunity to work on such an interesting topic, for always being available to answer all my questions, and for providing invaluable scientific guidance. Concerning the thesis, I would also like to express my deepest gratitude to my sister Federica that greatly helped me in the review of the written part.

Special thanks to my parents, my sister, and my girlfriend Alice for loving and supporting me, especially in times where I did not believe in myself. They are my pillars, and I could have not made it without them. My sincere thanks also goes to all my relatives for constantly encouraging me and showing interest in my studies.

There are three groups of friends that I would like to thank. My favorite group of engineers, Giacomo, Edoardo, and Andrea, one of the few friends that could actually understand the topic of this thesis and always ready to help when in need; special shoutout to Andrea for listening to me and providing useful material when I was dealing with the optimization problem. Next, I would like to thank my dear friends Matteo, Rachele, Federico, Edoardo, Claudia, Diego, Amanda, Mattia, Davide, and Nicolò whose support was beyond all my expectations. Finally, I would like to thank Josh and Piana who were the most incredible thing that could have happened to me during the epidemic, and kept my enthusiasm and motivation high during tough times.