

Research paper

Delta-V-optimal centralized guidance strategy for under-actuated N-satellite formations

Ahmed Mahfouz^{a,*}, Gabriella Gaias^b, Florio Dalla Vedova^c, Holger Voos^{a,d}^a SnT, University of Luxembourg, 29, Avenue J.F Kennedy, Luxembourg, 1855, Luxembourg^b Department of Aerospace Science and Technology, Politecnico di Milano, 34, via La Masa, Milan, 20156, Italy^c LuxSpace, 9, Rue Pierre Werner, Betzdorf, 6832, Luxembourg^d Faculty of Science, Technology and Medicine, University of Luxembourg, 2, place de l'Université, Esch-sur-Alzette, 4365, Luxembourg

ARTICLE INFO

Keywords:

Formation flying
Relative Orbital Elements
Relative trajectory optimization
Convexification
Convex optimization
Sequential convex programming

ABSTRACT

This paper addresses the computation of Delta-V-optimal, safe, relative orbit reconfigurations for satellite formations in a centralized fashion. The formations under consideration comprise an uncontrolled chief spacecraft flying with an arbitrary number, N , of deputy satellites, where each deputy is equipped with a single electric thruster. Indeed, this represents a technological solution that is becoming widely employed by the producers of small-satellite platforms. While adopting a single electric thruster does reduce the required power, weight, and size of the orbit control system, it comes at the cost of rendering the satellite under-actuated. In this setting, the satellite can provide a desired thrust vector only after an attitude maneuver is carried out to redirect the thruster nozzle opposite to the desired thrust direction. In order to further extend the applicability range of such under-actuated platforms, guidance strategies are developed to support different reconfiguration scenarios for N -satellite formations. This paper starts from a classical non-convex quadratically constrained trajectory optimization formulation, which passes through multiple simplifications and approximations to arrive to two novel convex formulations, namely a second-order cone programming formulation, and a linear programming one. Out of five guidance formulations proposed in this article, the most promising three were compared through an extensive benchmark analysis that is applied to fifteen of the most widely-used solvers. This benchmark experiment provides information about the key distinctions between the different problem formulations, and under which conditions each one of them can be recommended.

1. Introduction

As the world entered the new space era, it has witnessed a shift in the design philosophy of satellites. A modern space mission commonly comprises one or more simpler, smaller, lighter, lower-powered, and cheaper satellites, in contrast to a traditional mission, which typically relied on a single, sophisticated, large, heavy, power demanding, and expensive spacecraft [1]. By deploying multiple small satellites in a coordinated formation, these missions can cover larger areas, provide redundancy, and offer more frequent data updates.

As the satellites gradually became smaller in size, it was no wonder that Electric Propulsion (EP) systems were increasingly used to support their maneuverability in orbit. At the cost of lower thrust levels, electric thrusters typically offer higher specific impulse, and hence better fuel efficiency than their chemical counterparts [2]. The low fuel and propellant storage requirements is mainly what makes EP an attractive option for small satellites [3]. Often motivated by the need

to reduce weight, complexity, and hence cost, some satellite manufacturers started opting to incorporate a single electric thruster onboard their spacecraft. Examples of such satellites include the PLATiNO platform [4] and Triton-X [5]. A satellite that employs a single electric thruster is, in nature, under-actuated, since a desired thrust vector is only achievable after an attitude maneuver which redirects the thruster nozzle into the desired direction. This paper focuses on the problem of optimizing the trajectories necessary for a reconfiguration of an arbitrary number of such under-actuated satellites flying together in a formation.

The problem of formation reconfiguration has been a subject of extensive research. While guidance and control strategies have been proposed for formations that leverage impulsive thrusters [6–9], they are not applicable to formations that employ electric ones. Existing low-thrust guidance schemes assume omnidirectional thrusting [10, 11], rendering them inapplicable to single-thruster satellite formations. Guidance and control policies were developed for the AVANTI

* Corresponding author.

E-mail address: Ahmed.mahfouz@uni.lu (A. Mahfouz).

mission [12,7,13], which incorporates a satellite equipped with one impulsive thruster flying in a formation with a non-cooperative object, yet again, these guidance and control methods are not suitable for low-thrust formations. On a related note, a satellite incorporating a single impulsive thruster cannot be considered under-actuated, unlike a spacecraft with a sole low-thruster. An impulsive thruster is idle for the most part of the maneuver due to the fact that it is able to provide large Delta-V changes, typically in a matter of seconds. A thruster-idle period is ample for an attitude redirection maneuver to take place before the following thruster firing. To address the problem of relative orbit corrections for mono-electric-thruster satellites, Model Predictive Control (MPC) schemes have been proposed in [14,15], yet operational constraints, such as the necessary thruster-off-periods during ground contact, during payload operations, or during eclipse, were not considered in these works.

In this article, the Delta-V-optimal, collision-free, formation reconfiguration problem is considered for an arbitrary number, N , of deputy satellites flying in a formation with an uncontrolled chief. To allow for such reconfiguration to take place, each deputy is equipped with one electric thruster. The proposed trajectory optimization procedures are open-loop control strategies which can be incorporated in the closed feedback loop in many ways; one of which is a shrinking-horizon MPC such as the one presented in [16]. The first attempt to formulate the guidance problem is presented as a non-convex Quadratically Constrained Quadratic Programming (QCQP) problem, which is later relaxed to a convex QCQP that can be solved using Sequential Convex Programming (SCP). Due to the fact that the optimal solutions to the QCQP formulations require unnecessarily large Delta-V changes, the convex QCQP problem is reformulated into a Second-Order Cone Programming (SOCP) problem which makes better use of the available fuel. A final relaxation is applied to the SOCP formulation to transform it into the simplest form of convex programming problems; Linear Programming (LP). The main contributions of this paper are as follows:

- The mathematical formulations of the SOCP and the LP problems, which produce more Delta-V efficient solutions in comparison to the classical QCQP problems, while at the same time being easier to implement and faster to solve by most solvers;
- A benchmark experiment which is carried out on more than a dozen solvers over the convex QCQP, the SOCP, and the LP formulations;
- Guidelines for embedding the guidance algorithms for space-borne applications, drawn considering typical reconfigurations problems tackled in remote sensing applications.

The guidance plans presented in this paper are implemented as part of the AuFoSat toolbox; a Guidance, Navigation and Control library, developed to support the future missions of Triton-X; the multi-mission Low-Earth Orbit (LEO) microsatellite platform developed by LuxSpace. Previous AuFoSat research discussed orbit design [17], relative navigation [18,19], absolute orbit keeping [15], and relative orbit corrections for two-satellite formations [16].

The article is organized as follows: Section 2 introduces the mathematical model used in the development of the trajectory optimization formulations, specifically a linearized model of the quasi-non-singular Relative Orbital Elements (ROE). In Section 3, the various formulations of the formation reconfiguration problem are presented. A sensitivity analysis of the trajectory optimization schemes to different parameters is discussed in Section 5. Section 6 details the implications of employing the final guidance formulation and presents the benchmark experiment, along with a discussion on the limitations of the proposed guidance schemes. Finally, the paper is concluded in Section 7.

2. Dynamical model

In this section, a dynamical model of the quasi-non-singular Relative Orbital Elements (ROE), which is necessary for the development of

the trajectory optimization strategies, is introduced. Multiple reference frames are utilized in the development of the ROE dynamics. The ones used in this work are; an Earth-Centered-Inertial frame (ECI), denoted as $\mathbb{F}^i$. The True of Date (TOD) Earth Equator frame is the one referred to by ECI in this article; a Satellite-body-fixed frame, denoted by $\mathbb{F}^b$, with axes along the principal axes of inertia of the satellite; and the Radial-Transversal-Normal frame (RTN), denoted as $\mathbb{F}^r$, with its x -axis along the position vector of the chief pointing away from the Earth, with its z -axis along the normal direction to the chief's orbital plane, and with its y -axis completing the right-handed triad. Vectors expressed in $\mathbb{F}^i$, $\mathbb{F}^b$, or $\mathbb{F}^r$ are signified by the superscripts $(\cdot)^i$, $(\cdot)^b$, or $(\cdot)^r$ respectively.

The following set of orbital elements can be used to describe the orbital motion of a satellite, which is under the gravitational influence of a large celestial body such as the Earth, in a planet centered reference frame,

$$\alpha := [a \quad \theta \quad e_x \quad e_y \quad i \quad \Omega]^T, \quad (1)$$

where a is the semi-major axis, θ is the mean argument of latitude, $\mathbf{e} := [e_x \quad e_y]^T = [e \cos \omega \quad e \sin \omega]^T$, is the eccentricity vector with e being the eccentricity of the orbit and ω being the argument of periapsis. Furthermore, i is the orbital inclination and Ω is the Right Ascension of the Ascending Node (RAAN). A Cartesian state vector, $\mathbf{x}^i := [(\mathbf{r}^i)^T \quad (\mathbf{v}^i)^T]^T$, can equally represent the motion of the satellite, where $\mathbf{r}^i$ and $\mathbf{v}^i$ are the absolute position and velocity vectors expressed in $\mathbb{F}^i$. Note that the Cartesian state vector can be mapped through a nonlinear transformation to the classical orbital elements, and vice versa [20]. When this mapping is applied to the precise Cartesian state vector of the satellite, the instantaneous osculating orbital elements are produced, which, in the discussions to follow, are denoted by $\tilde{\alpha}$. Conversely, the mean orbital elements, denoted in the rest of the paper by α , are the one-orbit averaged elements, where the short- and long-term oscillations generated by the J_2 harmonic of the Earth gravitational potential are removed. The transformations in [21] are utilized to perform the mean/osculating elements mapping.

The motion of the i th deputy satellite with respect to the chief spacecraft is parameterized in this research by the dimensionless quasi-non-singular ROE vector, which is defined, using the orbital elements of both, the deputy and the chief, as follows,

$$\delta_i \alpha := \begin{bmatrix} \delta_i a \\ \delta_i \lambda \\ \delta_i e_x \\ \delta_i e_y \\ \delta_i i_x \\ \delta_i i_y \end{bmatrix} = \begin{bmatrix} \Delta_i a / a_c \\ \Delta_i \theta + \Delta_i \Omega \cos i_c \\ \Delta_i e_x \\ \Delta_i e_y \\ \Delta_i i \\ \Delta_i \Omega \sin i_c \end{bmatrix}, \quad (2)$$

where $\delta_i \alpha$ is the dimensionless ROE vector of the i th deputy, $\delta_i a$ is its relative semi-major axis, $\delta_i \lambda$ is the relative mean longitude, $\delta_i \mathbf{e} := [\delta_i e_x \quad \delta_i e_y]^T$ is the relative eccentricity vector, and $\delta_i \mathbf{i} := [\delta_i i_x \quad \delta_i i_y]^T$ is the relative inclination vector. It is to be noted that in this paper, the subscript $(\cdot)_i \forall i \in \{1, 2, \dots, N\}$, with N being the number of deputies, denotes a quantity related to the i th deputy satellite, while the subscript $(\cdot)_c$ is used for chief-related quantities. Furthermore, $\delta_i(\cdot)$ signifies a relative quantity between the i th deputy and the chief, which may or may not be the arithmetic difference between the two quantities, while $\Delta_i(\cdot)$ signifies the arithmetic difference between $(\cdot)_i$ and $(\cdot)_c$, i.e., $\Delta_i(\cdot) := (\cdot)_i - (\cdot)_c$. As in the case of absolute orbital elements, the osculating ROE vector is denoted by $\delta_i \tilde{\alpha}$, whereas the mean ROE vector is referred to as $\delta_i \alpha$. A dimensional ROE vector is obtained by multiplying the dimensionless ROE vector by the semi-major axis of the chief,

$$\mathbf{y}_i = a_c \delta_i \alpha, \quad (3)$$

where $\mathbf{y}_i$ is the dimensional mean ROE vector of the i th deputy, with units of length.

Assuming neighboring orbits of the chief and the deputies, and a near-circular orbit of the chief, the dynamics of the ROE can be

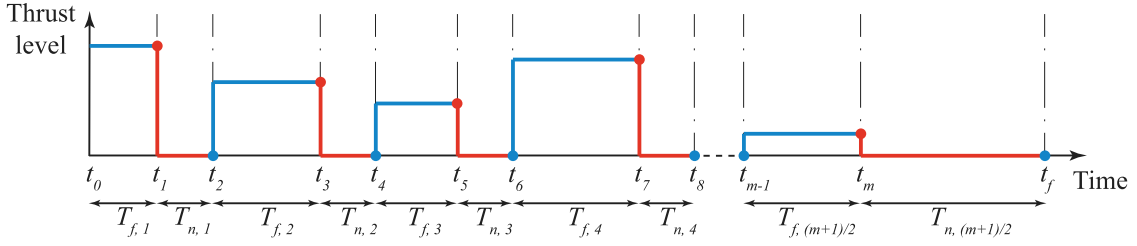


Fig. 1. Graphical representation of the low-thrust guidance strategy.

linearized to the first order considering the mean effect of the J_2 zonal harmonic. In fact, a closed form solution of the linearized dynamics can be obtained for piece-wise constant input acceleration as discussed in [8]. The system evolution is expressed in the following form,

$$\mathbf{y}_i(t_{k+1}) = \Phi(t_k, t_{k+1}) \mathbf{y}_i(t_k) + \Psi(t_k, t_{k+1}) \bar{\mathbf{u}}_i^r(t_k, t_{k+1}), \quad (4)$$

where $\Phi(t_k, t_{k+1})$ is the State Transition Matrix (STM) between the two time instants, t_k and t_{k+1} , $\Psi(t_k, t_{k+1})$ is the convolution matrix between the same two time instants, and $\bar{\mathbf{u}}_i^r(t_k, t_{k+1}) = \mathbf{a}_c \mathbf{u}_i^r(t_k, t_{k+1})$, with $\mathbf{u}_i^r(t_k, t_{k+1}) = [u_{i,R}(t_k, t_{k+1}) \quad u_{i,T}(t_k, t_{k+1}) \quad u_{i,N}(t_k, t_{k+1})]^\top$ being the input acceleration vector provided by the i th deputy's thruster, expressed in $\mathbb{R}^3$, and constant over the period $[t_k, t_{k+1})$. In the rest of the text, and in order to simplify the representation of equations, the following notations are used, $\Phi_k \equiv \Phi(t_k, t_{k+1})$, $\Psi_k \equiv \Psi(t_k, t_{k+1})$, $\mathbf{y}_k \equiv \mathbf{y}(t_k)$, $\bar{\mathbf{u}}_{i,k} \equiv \bar{\mathbf{u}}_i^r(t_k, t_{k+1})$, and $\mathbf{u}_{i,k} \equiv \mathbf{u}_i^r(t_k, t_{k+1})$. Note that Φ_k and Ψ_k do not relate to the i th deputy since they only depend on chief-related quantities [8].

3. Guidance

In this section, a guidance scheme is developed for a formation that comprises an arbitrary number, N , of deputy satellites together with a chief spacecraft. The ultimate goal of the guidance module is to drive the state of each deputy from its initial values, $\mathbf{y}_{i,0}$ at the initial time, t_0 , to the user-defined reference state, $\mathbf{y}_{i,f}$ at the final time, t_f . The N deputies are assumed to be each equipped with a single throttleable electric thruster dedicated for (relative) orbit reconfiguration, while the chief is uncontrolled and is assumed to be the spacecraft onboard of which the guidance calculations take place. In this under-actuated setting, each deputy has to undergo repeating attitude maneuvers in order to redirect its thruster in the desired direction before every thruster firing. The time required for these redirection maneuvers was taken into consideration in the design phase of the guidance procedure, and the thruster of each deputy is thus dictated to operate on an alternating on/off mode, where the slew maneuvers are allocated in the thruster off-periods. On top of the natural relative orbit changes during the thrusters off-periods, further forced ROE corrections are simultaneously induced for all the deputies through $\frac{m+1}{2}$ finite burns, with m being an odd number. Fig. 1 illustrates the alternation between the on and off states of the electric thruster onboard of one of the deputies which shall drive its state from the initial to the reference value.

It is important to emphasize that the time instances at which the thruster alternates between the on and off states, $\mathbf{t} = [t_0 \quad t_1 \quad \dots \quad t_f]$ (see Fig. 1), are assumed to be identical for all the deputies. It is, in fact, a common practice to unify the discretization steps for all the deputies in centralized trajectory optimization schemes [10,22]. This is particularly relevant for close formations, where all satellites undergo eclipse and establish ground contact at nearly the same times. Additionally, in our context where the time intervals between each two successive time steps in $\mathbf{t}$ are typically in the order of hundreds of seconds, synchronization offsets are not expected to be a notable source of error. The problem of specifying a separate time vector for each deputy is out of the scope of this paper, and will be the focus of our

future research. In fact, the problems of computing and broadcasting this time vector are also out of the scope of this study. The common time vector is left for the mission operator to determine, in order to enhance the mission predictability, which is a very important factor in the operation of real missions. It appears that leaving $\mathbf{t}$ as a user-input not only helps predict the behavior of the formation ahead of time, but also helps accommodate any operational constraints within the thrusters idle intervals. Operational constraints in our context might include not being able to use the thruster or the attitude control system for relative orbit correction during specific times, e.g., during ground contact, payload operations, or during times when the satellite is in the shadow. The time vector, $\mathbf{t}$, can be defined either through the time instances at which the thrusters switch between the on and off states, or through defining the initial time, t_0 , and the periods during which the thrusters are turned on or off. As a result, by letting $\mathcal{L} = \{1, 2, \dots, (m+1)/2\}$, these periods are divided into two categories;

- The forced motion time periods, $T_{f,l}$, $l \in \mathcal{L}$;
- The natural motion periods (coast arcs), $T_{n,l}$, $l \in \mathcal{L}$.

Since the attitude redirection maneuvers are allocated during the natural motion periods, a lower bound for these coast arcs has to be imposed which is related to the maximum allowable angular speed, i.e., angular velocity's Euclidean norm, of the deputies, such that,

$$T_{n,l} \geq \frac{\pi}{\omega_{\max}} + T_{\text{safety}} \quad \forall l \in \mathcal{L}, \quad (5)$$

where $\omega_{\max} = \min(\omega_{i,\max})$ is the maximum angular speed of the deputy with slowest angular rate, with $\omega_{i,\max}$ being the maximum angular speed of the i th deputy, and $T_{\text{safety}} \geq 0$ is added to ensure that the coast arc can accommodate the most stringent attitude maneuver.

Letting,

$$\mathbf{Y} = \begin{bmatrix} \mathbf{y}_{1,0} & \mathbf{y}_{2,0} & \dots & \mathbf{y}_{N,0} \\ \mathbf{y}_{1,1} & \mathbf{y}_{2,1} & \dots & \mathbf{y}_{N,1} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{y}_{1,m+1} & \mathbf{y}_{2,m+1} & \dots & \mathbf{y}_{N,m+1} \end{bmatrix}, \quad (6)$$

$$\bar{\mathbf{U}} = \begin{bmatrix} \bar{\mathbf{u}}_{1,0} & \bar{\mathbf{u}}_{2,0} & \dots & \bar{\mathbf{u}}_{N,0} \\ \bar{\mathbf{u}}_{1,1} & \bar{\mathbf{u}}_{2,1} & \dots & \bar{\mathbf{u}}_{N,1} \\ \vdots & \vdots & \ddots & \vdots \\ \bar{\mathbf{u}}_{1,m} & \bar{\mathbf{u}}_{2,m} & \dots & \bar{\mathbf{u}}_{N,m} \end{bmatrix}, \quad (7)$$

the guidance problem can be formally written as an optimization problem with a quadratic objective function, as is classically done [23], as follows,

Problem 1 (Non-convex Formulation).

$$\min_{\mathbf{Y}, \bar{\mathbf{U}}} \frac{1}{a_c^2} \sum_{i \in \mathcal{I}} \sum_{k \in \mathcal{K}_f} \left(\Delta t_k^2 \bar{\mathbf{u}}_{i,k}^\top \bar{\mathbf{u}}_{i,k} \right)$$

subject to,

$$\mathbf{y}_{i,0} = \mathbf{y}_{i,0}, \quad \mathbf{y}_{i,m+1} = \mathbf{y}_{i,f} \quad \forall i \in \mathcal{I}, \quad (8)$$

$$\mathbf{y}_{i,k+1} = \Phi_k \mathbf{y}_{i,k} + \Psi_k \bar{\mathbf{u}}_{i,k} \quad \forall i \in \mathcal{I}, \quad \forall k \in \mathcal{K}, \quad (9)$$

$$\bar{\mathbf{u}}_{i,k} = \mathbf{0} \quad \forall i \in \mathcal{I}, \quad \forall k \in \mathcal{K}_n, \quad (10)$$

$$\bar{\mathbf{u}}_{i,k}^T \bar{\mathbf{u}}_{i,k} \leq a_c^2 u_{i,max}^2 \quad \forall i \in \mathcal{I}, \forall k \in \mathcal{K}_f, \quad (11)$$

$$(\mathbf{y}_{i,k} - \mathbf{y}_{j,k})^T \mathbf{T}_k^T \mathbf{T}_k (\mathbf{y}_{i,k} - \mathbf{y}_{j,k}) \geq R_{CA}^2 \quad \forall i, j \in \mathcal{I}, i \neq j, \forall k \in \mathcal{K}, \quad (12)$$

$$\mathbf{y}_{i,k}^T \mathbf{T}_k^T \mathbf{T}_k \mathbf{y}_{i,k} \geq R_{CA}^2 \quad \forall i \in \mathcal{I}, \forall k \in \mathcal{K}, \quad (13)$$

where $\Delta t_k = t_{k+1} - t_k$, $\mathbf{0}$ is a vector of zeros, $\mathcal{I} = \{1, 2, \dots, N\}$ is the set of deputies' indices, $\mathcal{K} = \mathcal{K}_f \cup \mathcal{K}_n$, with $\mathcal{K}_f = \{0, 2, 4, \dots, m-1\}$ being the set of time indices where the thruster of each deputy is turned on, i.e., forced motion periods, and $\mathcal{K}_n = \{1, 3, 5, \dots, m\}$ being the set of time indices where the thruster of each deputy is turned off, i.e., natural motion periods, $u_{i,max}$ is the maximum allowable acceleration by the onboard thruster of the i th deputy. Generally, it is the maximum allowable thrust that is provided in the data sheets of electric thrusters, and not the maximum acceleration. However, the latter can be calculated from the former according to: $u_{i,max} = f_{i,max}/M_i$, with $f_{i,max}$ being the maximum thrust of the i th deputy, and M_i being its mass, which is assumed constant throughout the maneuver, and is set to the mass of the satellite at t_0 . Furthermore, in [Problem 1](#), $R_{CA} \geq 0$ is the radius of the collision avoidance sphere, and $\mathbf{T}_k \in \mathbb{R}^{3 \times 6}$ is the matrix that transforms a dimensional ROE vector into its corresponding position vector in the RTN frame. An explicit expression for this matrix can be found in [\[24\]](#). It is important to note that $\bar{\mathbf{u}}_{i,k}$ are chosen to be included as optimization variables instead of $\mathbf{u}_{i,k}$ in order to ensure that all the decision variables, i.e., $\mathbf{Y}$ and $\bar{\mathbf{U}}$, are of comparable orders of magnitude, which makes it less probable that a solver will run into numerical issues. Poorly-scaled problems typically require longer times to solve, if a solution can be found in the first place.

The constraints imposed on [Problem 1](#) are summarized as follows,

- Eq. (8) represents the boundary constraints which dictates the guidance profile to have the final state exactly equal to the set points defined by the user, while respecting the initial state of each deputy;
- The relative orbital dynamics are imposed on the trajectory optimizer through Eq. (9);
- The acceleration constraints are forced through Eqs. (10) and (11). Note that Eq. (10) is a hard constraint to ensure that the input acceleration onboard of each deputy provided during attitude redirection maneuvers is exactly zero;
- Inter-deputy collision is avoided by imposing Eq. (12), which guarantees that no deputy enters the collision sphere of another, while deputy-chief collision is prohibited at each time step through Eq. (13). Since the time steps of our application are relatively small when considering how slow the relative orbital dynamics can be, inter-step collision avoidance is ignored in this study.

It is clear that the cost function of [Problem 1](#) is the sum of the squared second Lebesgue (L2) norms of all the control Delta-V across all deputies and across all time instances during the reconfiguration maneuver. Minimizing this cost function not only results in a Delta-V-optimal solution, but also produces a fuel-optimal solution. Indeed, the terms “Delta-V-optimal” and “fuel-optimal” can be used interchangeably in our case, since they really do refer to the same thing when the controlled spacecraft are each equipped with a single thruster [\[25\]](#).

[Problem 1](#) is a non-convex optimization problem which requires a series of elaborate processes in order to find its globally optimal solution. The non-convexity of [Problem 1](#) arises solely from the non-convexity of the two last constraints. In fact, if it were not for constraints (12) and (13), [Problem 1](#) would have been a convex optimization problem that is guaranteed to have a globally optimal solution. One way to convexify the problem is to approximate the non-convex constraints by affine ones, then solve the problem through Sequential Convex Programming (SCP). Using the relaxation proposed in [\[22\]](#) for the collision avoidance constraints, [Problem 1](#) can be rewritten in its relaxed convex Quadratically Constrained Quadratic Programming (convex QCQP) form as follows,

Problem 2 (Convex QCQP Formulation).

$$\min_{\mathbf{Y}, \bar{\mathbf{U}}} \frac{1}{a_c^2} \sum_{i \in \mathcal{I}} \sum_{k \in \mathcal{K}_f} (\Delta t_k^2 \bar{\mathbf{u}}_{i,k}^T \bar{\mathbf{u}}_{i,k})$$

subject to,

$$\mathbf{y}_{i,0} = \mathbf{y}_{i,0}, \quad \mathbf{y}_{i,m+1} = \mathbf{y}_{i,f} \quad \forall i \in \mathcal{I}, \quad (14)$$

$$\mathbf{y}_{i,k+1} = \Phi_k \mathbf{y}_{i,k} + \Psi_k \bar{\mathbf{u}}_{i,k} \quad \forall i \in \mathcal{I}, \forall k \in \mathcal{K}, \quad (15)$$

$$\bar{\mathbf{u}}_{i,k} = \mathbf{0} \quad \forall i \in \mathcal{I}, \forall k \in \mathcal{K}_n, \quad (16)$$

$$\bar{\mathbf{u}}_{i,k}^T \bar{\mathbf{u}}_{i,k} \leq a_c^2 u_{i,max}^2 \quad \forall i \in \mathcal{I}, \forall k \in \mathcal{K}_f, \quad (17)$$

$$(\bar{\mathbf{y}}_{i,k} - \bar{\mathbf{y}}_{j,k})^T \mathbf{T}_k^T \mathbf{T}_k (\bar{\mathbf{y}}_{i,k} - \bar{\mathbf{y}}_{j,k}) \geq R_{CA} \|\mathbf{T}_k (\bar{\mathbf{y}}_{i,k} - \bar{\mathbf{y}}_{j,k})\| \quad \forall i, j \in \mathcal{I}, i \neq j, \forall k \in \mathcal{K}, \quad (18)$$

$$\bar{\mathbf{y}}_{i,k}^T \mathbf{T}_k^T \mathbf{T}_k \bar{\mathbf{y}}_{i,k} \geq R_{CA} \|\mathbf{T}_k \bar{\mathbf{y}}_{i,k}\| \quad \forall i \in \mathcal{I}, \forall k \in \mathcal{K}, \quad (19)$$

where $\bar{\mathbf{y}}_{i,k}$ is a predicted value for the dimensional ROE vector of the i th deputy at time t_k and $\|\cdot\|$ is the second norm of a vector. The solution to [Problem 2](#) can be obtained sequentially through Sequential Convex Programming (SCP), where $\bar{\mathbf{y}}_{i,k}$ for the current iteration is set to $\mathbf{y}_{i,k}$ from the previous iteration. For the first iteration, $\bar{\mathbf{y}}_{i,k}$ can be obtained in a variety of ways, e.g., imposing the dynamics solution, Eq. (4), from the initial to the final times with no control inputs, or alternatively solving the problem without the collision avoidance constraints first and extracting $\mathbf{y}_{i,k}$ from the solution, then setting $\bar{\mathbf{y}}_{i,k} = \mathbf{y}_{i,k}$. The SCP scheme is set to terminate when one of the following criteria is satisfied,

- $\|\bar{\mathbf{y}}_{i,k} - \mathbf{y}_{i,k}\| \leq \epsilon$ at the current iteration, with $\epsilon > 0$ being a user-defined threshold;
- The guidance profile of the current iteration is collision free;
- The user-defined maximum number of iteration is reached, in which case, the solution trajectory is not guaranteed to be collision-free.

One of the main reason why [Problem 1](#) could be easily transformed into the convex form of [Problem 2](#) is keeping the time vector t a user-input. The benefits of this approach, as previously discussed, include improved predictability of the mission and the ability to accommodate operational time constraints. Additionally, it helps in ensuring that the optimization problem does not become overly complicated by turning into a complex Non-Linear Programming (NLP) problem. While allowing the time steps to be decision variables can potentially improve the time-optimality of the solution, it would also significantly increase the complexity of the problem [\[26\]](#).

Since the objective function of [Problem 2](#), and that of [Problem 1](#) for that matter, aggregates the squared L2 norms of the Delta-V vectors, across deputies and across time instances, it comes as no surprise that the solver gives a stronger emphasis on larger values of Delta-V. While a quadratic objective function suppresses the control peaks, and hence enhances the smoothness of the optimal state trajectory [\[27\]](#), it makes [Problems 1](#) and [2](#) close to being minimization problems for the maximum Delta-V instance. An alternative approach is to set the cost function to the sum of the second-norms of the Delta-V vectors, rather than the sum of the squared norms. Adopting this approach, then transforming the problem into its separable epigraph form [\[28\]](#), which is a very close form to the epigraph problem form [\[29\]](#), the new problem can be written as a second-order cone programming problem as follows:

Problem 3 (SOCP Formulation).

$$\min_{\mathbf{Y}, \bar{\mathbf{U}}, \Gamma} \frac{1}{a_c} \sum_{i \in \mathcal{I}} \sum_{k \in \mathcal{K}_f} (\Delta t_k \Gamma_{i,k})$$

subject to,

$$\mathbf{y}_{i,0} = \mathbf{y}_{i,0}, \quad \mathbf{y}_{i,m+1} = \mathbf{y}_{i,f} \quad \forall i \in \mathcal{I}, \quad (20)$$

$$\mathbf{y}_{i,k+1} = \Phi_k \mathbf{y}_{i,k} + \Psi_k \bar{\mathbf{u}}_{i,k} \quad \forall i \in \mathcal{I}, \forall k \in \mathcal{K}, \quad (21)$$

$$\bar{\mathbf{u}}_{i,k} = \mathbf{0} \quad \forall i \in \mathcal{I}, \forall k \in \mathcal{K}_n, \quad (22)$$

$$\|\bar{\mathbf{u}}_{i,k}\| \leq \Gamma_{i,k}, \quad \forall i \in \mathcal{I}, \forall k \in \mathcal{K}_f, \quad (23)$$

$$\Gamma_{i,k} \leq a_c u_{i,max} \quad \forall i \in \mathcal{I}, \forall k \in \mathcal{K}_f, \quad (24)$$

$$(\bar{\mathbf{y}}_{i,k} - \bar{\mathbf{y}}_{j,k})^T \mathbf{T}_k^T \mathbf{T}_k (\mathbf{y}_{i,k} - \mathbf{y}_{j,k}) \geq R_{CA} \|\mathbf{T}_k (\bar{\mathbf{y}}_{i,k} - \bar{\mathbf{y}}_{j,k})\| \quad \forall i, j \in \mathcal{I}, i \neq j, \forall k \in \mathcal{K}, \quad (25)$$

$$\bar{\mathbf{y}}_{i,k}^T \mathbf{T}_k^T \mathbf{T}_k \mathbf{y}_{i,k} \geq R_{CA} \|\mathbf{T}_k \bar{\mathbf{y}}_{i,k}\| \quad \forall i \in \mathcal{I}, \forall k \in \mathcal{K}, \quad (26)$$

where Γ is a matrix which collates all the auxiliary variables that had to be introduced in order to put the problem into the separable epigraph form. Formally,

$$\Gamma = \begin{bmatrix} \Gamma_{1,0} & \Gamma_{2,0} & \cdots & \Gamma_{N,0} \\ \Gamma_{1,2} & \Gamma_{2,2} & \cdots & \Gamma_{N,2} \\ \vdots & \vdots & \ddots & \vdots \\ \Gamma_{1,m-1} & \Gamma_{2,m-1} & \cdots & \Gamma_{N,m-1} \end{bmatrix}. \quad (27)$$

It is important to note that SOCP problems can be handled in their native form, i.e., without having to transform the SOC constraints into quadratic ones, only by a handful of solvers, e.g., SCS, ECOS, and MOSEK. Many of the solvers require the SOC constraints to be transformed into quadratic ones, which, if applied to constraint (23), renders it non-convex, unless an additional linear constraint is added, namely, $\Gamma_{i,k} \geq 0 \quad \forall i \in \mathcal{I}, \forall k \in \mathcal{K}_f$. By adding this additional constraint, many solvers, e.g., Gurobi and Knitro, recognize the transformed quadratic constraint as a second order cone, and treat it as such.

Tracking the behavior of the Second-Order Cone (SOC) constraint, Eq. (23), is quite interesting since it behaves such that,

$$\|\bar{\mathbf{u}}_{i,k}^*\| = \Gamma_{i,k}^*, \quad \forall i \in \mathcal{I}, \forall k \in \mathcal{K}_f, \quad (28)$$

with $(\cdot)^*$ being the optimal solution to the problem, due to the fact that Problem 3 is pushing $\Gamma_{i,k}$ to be as low as possible (refer to the cost function of the problem), and since the lowest possible value for $\Gamma_{i,k}$ is $\|\bar{\mathbf{u}}_{i,k}\|$ according to constraint (23).

Problem 3 can be taken a step further and be formulated as a Linear Programming (LP) problem, which makes it possible for the new formulation to be solved by the vast majority of the numerical optimization solvers. This transformation can be applied through implementing a piece-wise linearization to the Euclidean norm function in Eq. (23). Transforming Problem 3 into an LP one not only enables the possibility to use many solvers that cannot handle the SOCP form, but may also require less time to solve than that of the SOCP formulation, although not necessarily. A less solve time can be anticipated for the LP formulation due to the fact that LP is presumably the simplest form of a convex optimization problem, and also due to the existence of dedicated LP algorithms, e.g., primal and dual simplex, which have matured over the last few decades. However, transforming the SOCP formulation into an LP one, comes at the cost of adding new constraints to the problem.

As previously mentioned, Problem 3 can be approximated to an LP form through relaxing the maximum acceleration SOCP constraint (sphere), Eq. (23), using multiple piece-wise affine constraints (polyhedron) as proposed in [16]. In [16], the piece-wise approximation was applied to the sphere such that its projections (circles) on each of the three planes, the Transversal–Normal (T–N), the Radial–Normal (R–N), and the Transversal–Radial (T–R), are each treated separately. Indeed, the optimal solution of Problem 3 is expected to have minor acceleration components in the radial direction [7,14], which is why a stronger emphasis is put on the acceleration components lying on the T–N plane by employing a finer grid than that which is applied in the R–N and the T–R planes. This is explained graphically in Fig. 2, where the feasibility volumes are projected on the T–R, the T–N, and the R–N planes.

Having introduced the concept behind the transformation of Problem 3 into its LP approximation, this reformulation can be written as follows,

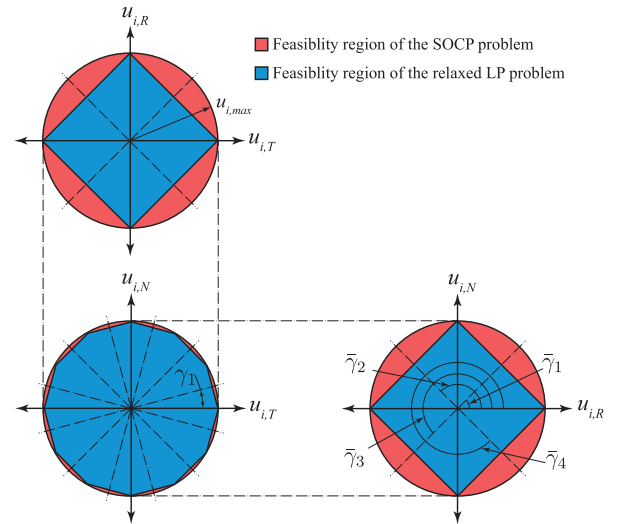


Fig. 2. Comparison between the control input feasibility region of the SOCP formulation against that of the LP one.

Problem 4 (LP Formulation).

$$\min_{\mathbf{Y}, \mathbf{U}, \Gamma} \frac{1}{a_c} \sum_{i \in \mathcal{I}} \sum_{k \in \mathcal{K}_f} (\Delta t_k \Gamma_{i,k})$$

subject to,

$$\mathbf{y}_{i,0} = \mathbf{y}_{i,0}, \quad \mathbf{y}_{i,m+1} = \mathbf{y}_{i,f} \quad \forall i \in \mathcal{I}, \quad (29)$$

$$\mathbf{y}_{i,k+1} = \Phi_k \mathbf{y}_{i,k} + \Psi_k \bar{\mathbf{u}}_{i,k} \quad \forall i \in \mathcal{I}, \forall k \in \mathcal{K}, \quad (30)$$

$$\bar{\mathbf{u}}_{i,k} = \mathbf{0} \quad \forall i \in \mathcal{I}, \forall k \in \mathcal{K}_n, \quad (31)$$

$$\begin{bmatrix} 0 & \cos(\gamma_d) & \sin(\gamma_d) \end{bmatrix} \bar{\mathbf{u}}_{i,k} \leq \Gamma_{i,k} \cos(\gamma_{max}) \quad \forall d \in \mathcal{D}, \forall i \in \mathcal{I}, \forall k \in \mathcal{K}_f, \quad (32)$$

$$\begin{bmatrix} \cos(\tilde{\gamma}_d) & \sin(\tilde{\gamma}_d) & 0 \\ \cos(\tilde{\gamma}_d) & 0 & \sin(\tilde{\gamma}_d) \end{bmatrix} \bar{\mathbf{u}}_{i,k} \leq \Gamma_{i,k} \cos(\tilde{\gamma}_{max}) \quad \forall d \in \tilde{\mathcal{D}}, \forall i \in \mathcal{I}, \forall k \in \mathcal{K}_f, \quad (33)$$

$$\Gamma_{i,k} \leq a_c u_{i,max} \quad \forall i \in \mathcal{I}, \forall k \in \mathcal{K}_f, \quad (34)$$

$$(\bar{\mathbf{y}}_{i,k} - \bar{\mathbf{y}}_{j,k})^T \mathbf{T}_k^T \mathbf{T}_k (\mathbf{y}_{i,k} - \mathbf{y}_{j,k}) \geq R_{CA} \|\mathbf{T}_k (\bar{\mathbf{y}}_{i,k} - \bar{\mathbf{y}}_{j,k})\| \quad \forall i, j \in \mathcal{I}, i \neq j, \forall k \in \mathcal{K}, \quad (35)$$

$$\bar{\mathbf{y}}_{i,k}^T \mathbf{T}_k^T \mathbf{T}_k \mathbf{y}_{i,k} \geq R_{CA} \|\mathbf{T}_k \bar{\mathbf{y}}_{i,k}\| \quad \forall i \in \mathcal{I}, \forall k \in \mathcal{K}. \quad (36)$$

where $\mathcal{D} = \{1, 2, \dots, n_{dir}\}$, with $n_{dir} > 4$ being the desired number of affine inequality constraints that approximate the projection of the SOC constraint, Eq. (23), on the Transversal–Normal plane, $\gamma_d = \frac{(2d-1)\pi}{n_{dir}} \quad \forall d \in \mathcal{D}$, and $\gamma_{max} = \frac{\pi}{4}$. Moreover, $\tilde{\mathcal{D}} = \{1, 2, 3, 4\}$, $\tilde{\gamma}_d = \frac{n_{dir}}{(2d-1)\pi} \quad \forall d \in \tilde{\mathcal{D}}$, and $\tilde{\gamma}_{max} = \frac{\pi}{4}$. In Fig. 2, the constraint relaxations in Eq. (32) is depicted for $n_{dir} = 12$ which covers approximately 95.5% of the original constraining circle in the T–N plane.

Similar to Problem 3, the optimal solution of Problem 4 lies at the boundaries of constraints (32) and (33). This implies that the second norm of the scaled input acceleration, $\|\bar{\mathbf{u}}_{i,k}\|$, lies on the surface of the polyhedron (depicted in Fig. 2 for $\Gamma_{i,k} = a_c u_{i,max}$) which relaxes the original feasibility sphere defined in Eq. (23).

It is to be emphasized that while the polyhedron that relaxes the acceleration norm constraint has projections on the T–N, the R–T, and the R–N planes that lie well inside the original feasibility sphere as seen in Fig. 2, some vertices of the polyhedron can be seen to slightly protrude from this sphere when visualized from different angles in the 3-dimensional space. This can be seen clearly in Fig. 3 for the case of $n_{dir} = 12$. Although the relaxing polyhedron of Problem 4 can change in size depending on the value of $\Gamma_{i,k}$, and hence lie with its entirety within the original constraining sphere, the fact that it can extend

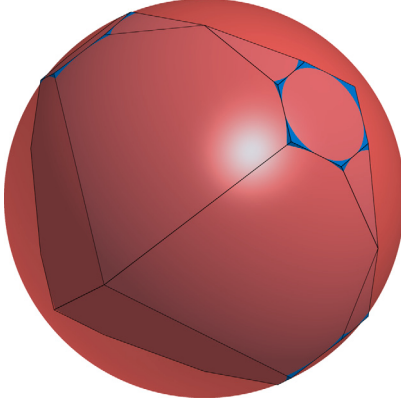


Fig. 3. Control input feasibility regions of Problems 3 and 4.

beyond the limits of the sphere when $\Gamma_{i,k} = a_c \bar{u}_{i,max}$ can be problematic. As the solution of Problem 4 may include points where $\Gamma_{i,k} = a_c \bar{u}_{i,max}$, the event of $\|\mathbf{u}_{i,k}\|$ getting slightly larger than $u_{i,max}$ becomes a real possibility, since the solution has to lie on the boundaries of constraints (32) and (33) as established earlier.

While the effect of this slight violation of the original constraints can be tolerated for short maneuvers, relative orbit correction maneuvers which require extended periods of time can deviate from their set points by tens or even hundreds of meters in the dimensional ROE space as will be discussed in Section 6. It is for this reason that constraints (32) and (33) are modified such that the approximating polyhedron is uniformly scaled down so that it can be contained entirely by the original sphere regardless of the value of $\Gamma_{i,k}$. The resulting problem after this scaling can be written as,

Problem 5 (LP Formulation with Scaled Feas. Region).

$$\min_{\mathbf{Y}, \mathbf{U}, \Gamma} \frac{1}{a_c} \sum_{i \in \mathcal{I}} \sum_{k \in \mathcal{K}_f} (\Delta t_k \Gamma_{i,k})$$

subject to,

$$\mathbf{y}_{i,0} = \mathbf{y}_{i,0}, \quad \mathbf{y}_{i,m+1} = \mathbf{y}_{i,f} \quad \forall i \in \mathcal{I}, \quad (37)$$

$$\mathbf{y}_{i,k+1} = \Phi_k \mathbf{y}_{i,k} + \Psi_k \bar{\mathbf{u}}_{i,k} \quad \forall i \in \mathcal{I}, \quad \forall k \in \mathcal{K}, \quad (38)$$

$$\bar{\mathbf{u}}_{i,k} = \mathbf{0} \quad \forall i \in \mathcal{I}, \quad \forall k \in \mathcal{K}_n, \quad (39)$$

$$\begin{bmatrix} 0 & \cos(\gamma_d) & \sin(\gamma_d) \end{bmatrix} \bar{\mathbf{u}}_{i,k} \leq \Gamma_{i,k} \cos(\gamma_{max}) \quad \forall d \in \mathcal{D}, \quad \forall i \in \mathcal{I}, \quad \forall k \in \mathcal{K}_f, \quad (40)$$

$$\begin{bmatrix} \cos(\tilde{\gamma}_d) & \sin(\tilde{\gamma}_d) & 0 \\ \cos(\tilde{\gamma}_d) & 0 & \sin(\tilde{\gamma}_d) \end{bmatrix} \bar{\mathbf{u}}_{i,k} \leq \Gamma_{i,k} \cos(\tilde{\gamma}_{max}) \quad \forall d \in \tilde{\mathcal{D}}, \quad \forall i \in \mathcal{I}, \quad \forall k \in \mathcal{K}_f, \quad (41)$$

$$\Gamma_{i,k} \leq a_c u_{i,max} \quad \forall i \in \mathcal{I}, \quad \forall k \in \mathcal{K}_f, \quad (42)$$

$$(\bar{\mathbf{y}}_{i,k} - \bar{\mathbf{y}}_{j,k})^T \mathbf{T}_k^T \mathbf{T}_k (\mathbf{y}_{i,k} - \mathbf{y}_{j,k}) \geq R_{CA} \|\mathbf{T}_k (\bar{\mathbf{y}}_{i,k} - \bar{\mathbf{y}}_{j,k})\| \quad \forall i, j \in \mathcal{I}, \quad i \neq j, \quad \forall k \in \mathcal{K}, \quad (43)$$

$$\bar{\mathbf{y}}_{i,k}^T \mathbf{T}_k^T \mathbf{T}_k \mathbf{y}_{i,k} \geq R_{CA} \|\mathbf{T}_k \bar{\mathbf{y}}_{i,k}\| \quad \forall i \in \mathcal{I}, \quad \forall k \in \mathcal{K}, \quad (44)$$

where c is a constant scaling factor that guarantees that the polyhedron of Fig. 3 is situated entirely within the sphere. The protrusion of the polyhedron beyond the constraining sphere is truly minimal, and could be quantified with the help of a 3D modeling software. In fact, the distance from the center of the polyhedron to the furthest vertex is measured to be approximately 1.7% larger than the radius of the original sphere (for $n_{dir} = 12$). In the following discussions, the value of n_{dir} is set to 12 while the constant c is fixed to 1.017. Appendix A provides an overview of various methods for calculating the scaling factor corresponding to different values of n_{dir} , and presents the scaling factors obtained for each scenario. It has been demonstrated that $n_{dir} =$

12 is one of the most effective choices, as it requires the smallest scaling factor among all the n_{dir} values that cover at least 95% of the circumcircle area in the T–N plane.

It is important to note that while mission operators usually favor closed-form solutions over numerical optimization-based ones due to the risk of running into an infeasible situation, infeasibility is never a problem in our context. A feasible solution can always be obtained when the maneuver is allowed enough time regardless of how hard the reconfiguration is, as long as the initial and final states are collision-free. Indeed, the maneuver duration is a user-input that can be planned and controlled by the ground operators.

4. Problem normalization

In most optimization problems, normalization of the cost function as well as the constraints improves numerical stability of the solver and increases its convergence velocity. This fact still remains true in the context of this paper, where the problems have been formulated in such a way that the decision variables are within almost the same order of magnitude. Normalization helps avoid issues of numerical instability due to poorly-scaled matrices which might cause ill-conditioning. Problem formulations involving badly-scaled matrices often result in poor solver convergence or in high error in a solution due to arithmetic on floating-point numbers with a limited number of digits [30]. By normalizing the cost function and, especially, constraints, such discrepancies are reduced, making the formulation easier to work with.

The objective functions of the proposed problems, specifically Problems 2, 3, and 5, are either linear or quadratic. Their constraints fall under three categories: quadratic, linear, and second-order cone constraints. This section provides a brief discussion on the normalization techniques adopted in this paper, for both types of objective functions and the three categories of constraints. Indeed, more sophisticated scaling techniques can be used, however, the simplest normalization methods often prove to be the most effective [30].

4.1. Quadratic objective functions

A quadratic cost function can be written in its general form as,

$$J_Q = 0.5 \mathbf{x}^T \mathbf{Q}_o \mathbf{x} + \mathbf{f}_o^T \mathbf{x}, \quad (45)$$

where $\mathbf{x}$ is the vector that collates all the decision variables. This cost function is normalized by dividing both, $\mathbf{Q}_o$ and $\mathbf{f}_o$ by the Frobenius norm of $\mathbf{Q}_o$, which guarantees that all the values in $\mathbf{Q}_o$ lie between -1 and 1 .

4.2. Linear objective functions

A linear cost function can be written in its general form as,

$$J_L = \mathbf{f}_o^T \mathbf{x}. \quad (46)$$

The linear cost function is normalized in this research by dividing $\mathbf{f}_o$ by its L_∞ norm.

4.3. Quadratic constraints

A quadratic constraint can be written in its general form as,

$$0.5 \mathbf{x}^T \mathbf{Q}_c \mathbf{x} + \mathbf{f}_c^T \mathbf{x} \leq b_u, \quad (47)$$

where the quadratic constraint is convex if and only if $\mathbf{Q}_c$ is positive semi-definite, and where b_u is the upper bound of the constraint. Such constraint is normalized by dividing $\mathbf{Q}_c$, $\mathbf{f}_c$, and b_u by the Frobenius norm of $\mathbf{Q}_c$.

4.4. Linear constraints

A collection of linear constraints can be written in the general form of,

$$\mathbf{b}_l \leq \mathbf{A}_c \mathbf{x} \leq \mathbf{b}_u, \quad (48)$$

where $\mathbf{b}_l$ and $\mathbf{b}_u$ are the vectors of lower and upper bounds, respectively. Equilibration is utilized to normalize these linear constraints, where each row of the matrix $\mathbf{A}_c$ is divided by its L_∞ norm. The corresponding elements in the upper and lower bounds are divided by the same norm to keep the scaling non-effective in the constraints.

4.5. Second-order cone constraints

For a SOC constraint of the form,

$$\|\mathbf{A}_c \mathbf{x} - \mathbf{b}_c\| \leq \mathbf{f}_c^T \mathbf{x} - g_c, \quad (49)$$

the scaling is done by dividing $\mathbf{A}_c$, $\mathbf{b}_c$, $\mathbf{f}_c$, and g_c by the Frobenius norm of $\mathbf{A}_c$.

5. Parameters sensitivity analysis

Upon examining the proposed formulations of the guidance problems and their concept of operation in Fig. 1, it is evident that certain user-provided parameters may require tuning. These parameters include the time instances for thruster activation and deactivation, or alternatively, the time periods $T_{f,l}$ and $T_{n,l} \forall l \in \mathcal{L}$, as well as the allocated maneuver time, $t_f - t_0$. The selection of the time periods $T_{f,l}$ and $T_{n,l}$ is typically constrained by mission requirements, such as avoiding thrust during eclipses, ground contact, or scientific experiments. In the absence of such constraints, it is natural to fix the time periods $T_{f,l}$ and $T_{n,l}$, i.e., $T_{f,l} = T_f$, $T_{n,l} = T_n \forall l \in \mathcal{L}$. For this specific scenario, a sensitivity analysis was conducted to evaluate the feasibility of the optimization problem with varying values of T_f and T_n . In this analysis, Problems 2, 3, and 5 were solved, excluding the collision avoidance constraints, for 100 randomly selected satellite numbers and initial conditions, i.e., N and $\mathbf{y}_{i,0}$, across 70 T_f - T_n combinations drawn from $T_n \in \{60, 90, \dots, 240\}$ s and $T_f \in \{0.05, 0.1, 0.15, \dots, 0.5\}$ orbits. Furthermore, since the difference between the initial and final ROE vectors characterizes the maneuver, rather than the vector values themselves, only the initial values $\mathbf{y}_{i,0}$ were chosen randomly, while the final values $\mathbf{y}_{i,f}$ were set to zero for the 7000 experiments. Specifically, the entries of the initial dimensional ROE vectors, $\mathbf{y}_{i,0}$, were randomly selected from the $[-1 \ 1]$ km range, except for the initial relative mean argument of longitude, $a_c \delta_i \lambda_0$, which was randomly chosen from the $[-100 \ 100]$ km range. The maneuver time was fixed at 20 orbits, and the number of deputies was randomly chosen from $\{1, 2, \dots, 6\}$.

Additionally, the mean orbital elements vector of the chief at the beginning of the maneuver was fixed for all 7000 experiments to $\mathbf{a}_{c,0} = [6761.45 \text{ km} \ 180^\circ \ 1.5 \cdot 10^{-3} \ 0 \ 98^\circ \ 0^\circ]^T$ at t_0 . A histogram of the 100 randomly chosen number of deputies used in the 7000 guidance runs is presented in Fig. 4.

The output of the sensitivity analysis is the success and failure regions in the T_f - T_n plane, where the success region is the one in which the optimizer succeeded to find a feasible solution for all the 100 random initial conditions and number of deputies. The success and failure regions of the aforementioned sensitivity study are depicted in Figs. 5(a), 6(a), and 7(a) for the QCQP, the SOCP, and the LP formulations, respectively.

Indeed, not all points within the success region are equivalent in terms of fuel efficiency, as variations in T_f and T_n alter the optimal Delta-V. To evaluate the effectiveness of each T_f - T_n combination within the success region, a fitness function is introduced. This fitness function

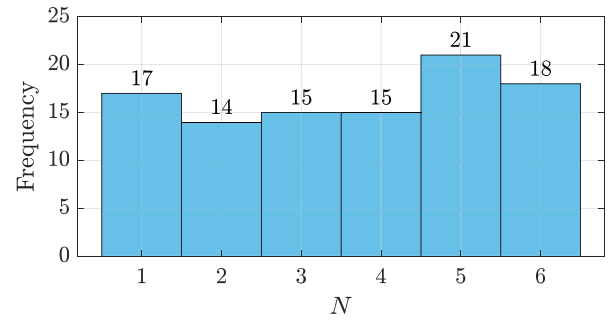


Fig. 4. Formation sizes over the 100 randomly chosen formation scenarios.

is defined as the reciprocal of the average optimal cost over the 100 initial conditions at each T_f - T_n point. Formally, it is expressed as,

$$\text{Fitness} = N_{IC} \sum_{l=1}^{N_{IC}} \frac{1}{J_l^*}, \quad (50)$$

where $N_{IC} = 100$ represents the number of initial conditions solved at each T_f - T_n point, and J_l^* denotes the optimal cost of the problem (i.e., QCQP, SOCP, or LP) for the l th initial condition. The fitness is calculated using Eq. (50) at each point in the success region, with the results presented in Figs. 5(b), 6(b), and 7(b) for the QCQP, SOCP, and LP formulations, respectively.

It is hardly surprising that the success region of the QCQP formulation (see Fig. 5(a)) mirrors that of the SOCP formulation (see Fig. 6(a)), as both problems explore the same control acceleration space. The success region of the LP formulation, however, is smaller (see Fig. 7(a)), as it explores a more limited feasibility region of the control acceleration. Notably, all formulations exhibit the same sensitivity to changes in T_f (see Fig. 5(b), Fig. 6(b), and Fig. 7(b)).

The results depicted in Figs. 5, 6, and 7 align with intuition. Changes in T_n , representing the duration of the coast arcs, minimally affect fitness for a fixed T_f value. However, if T_n becomes excessively large, the problem becomes infeasible. Conversely, a small T_f value allows for precise control over thrust timing, reflected in the optimal cost (higher fitness $\equiv$ lower cost). A very large T_f , however, is undesirable from both fitness and feasibility perspectives. Although large T_f values reduce the number of variables and expedite problem-solving, they eventually lead to infeasibility, as the optimization problem formulations require constant thrust throughout the T_f period, which can be overly restrictive.

To choose a proper value for T_f , a compromise has to be made between the complexity of the problem from one side, and the fitness of the T_n - T_f from another. It is for this reason that the adopted value of T_f in many of the numerical experiments to follow is set to 0.2, which is a value that guarantees acceptable fitness, and at the same time, produces an acceptable number of decision variables. Conversely, choosing an adroit value for T_n is rather straightforward, since T_n is typically calculated analytically using the equality option in Eq. (5) with a T_{safety} of around 10 s.

6. Results and discussion

In this section, the proposed LP guidance scheme is validated through a case-study and the results of this case-study are analyzed. The LP scheme offers more inherent details, such as the scaling factor, for discussion. Additionally, a brief comparison between the convex QCQP, the SOCP, and the relaxed LP formulations is carried out, through a benchmark study of multiple solvers. The limitations of the proposed guidance strategies are discussed at the end of the section.

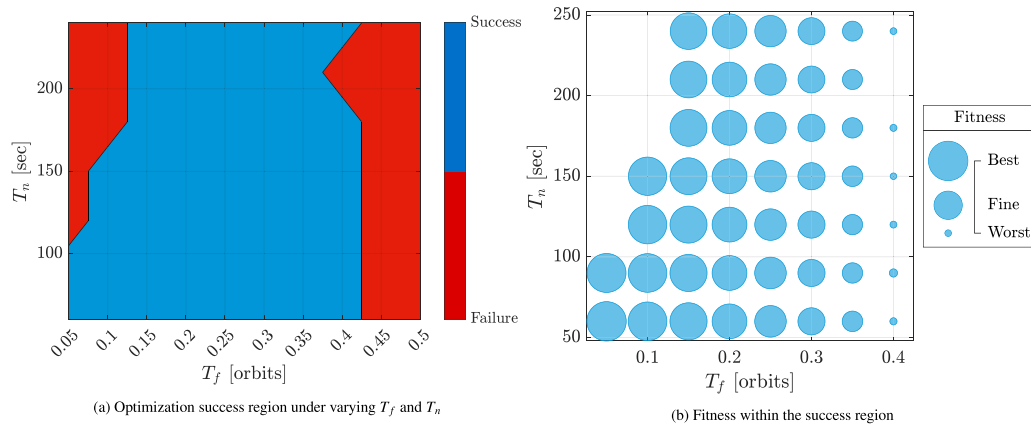


Fig. 5. Sensitivity of the QCQP formulation (Problem 2) to T_f and T_n variations.

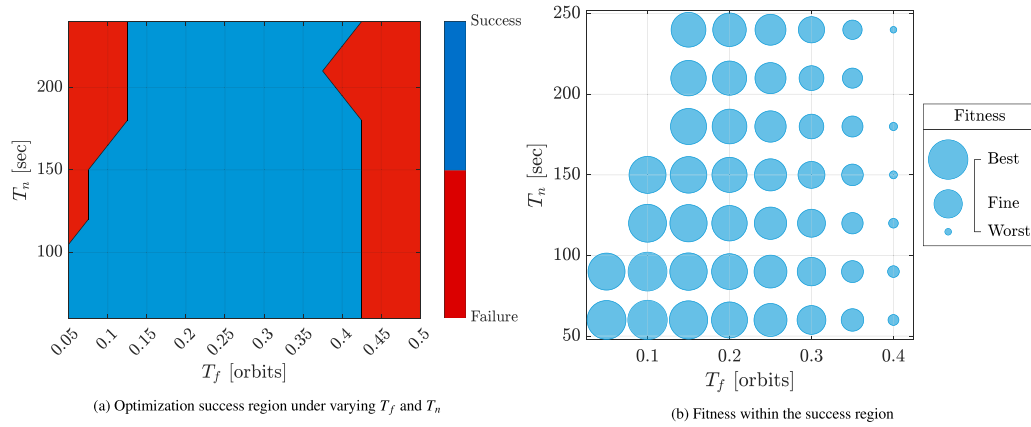


Fig. 6. Sensitivity of the SOCP formulation (Problem 3) to T_f and T_n variations.

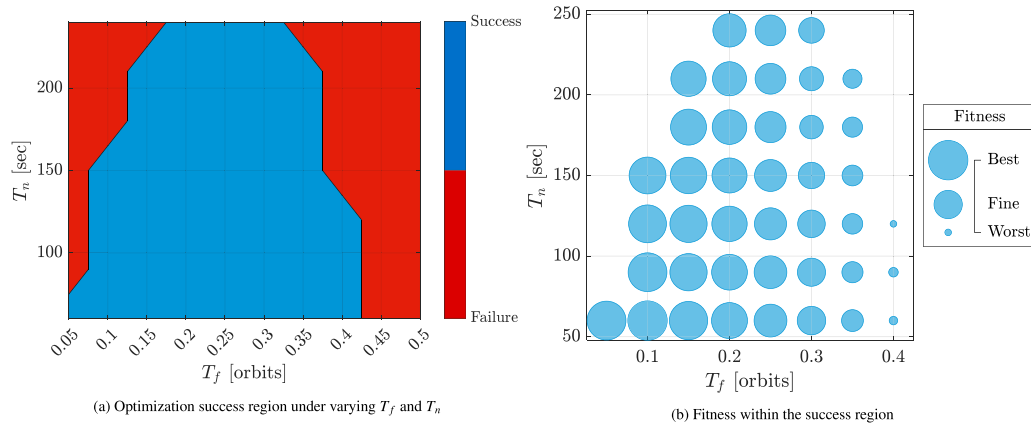


Fig. 7. Sensitivity of the LP formulation (Problem 5) to T_f and T_n variations.

6.1. Case-study

To test the effectiveness of the guidance schemes, they were run over multiple simulation scenarios. As an example, the details of one of these scenarios are presented here. Considering 4 identical satellites, Sat. A, Sat. B, Sat. C, and Sat. D, that are flying in a close formation with a chief. The four deputies are assumed to be initially equidistant from each other along the transversal direction (trailing/coplanar configuration) with the chief at the center of the formation, and are required to perform a formation reconfiguration such that they are distributed along a Projected Circular Orbit (PCO) at the final time.

The details of parameterizing the coplanar as well as the PCO configurations in terms of mean ROE are discussed in [10], and the initial as well as the final (set-point) states of each of the deputies are summarized in Table 1. The numbers in Table 1 indicate that the initial distance between each two consecutive satellites in the trailing formation is 200 m, while the radius of the final PCO is 300 m. Moreover, the orbit of the chief is characterized by $\alpha_{c,0} = [6761.45 \text{ km} \ 180^\circ \ 1.5 \cdot 10^{-3} \ 0 \ 98^\circ \ 0^\circ]^T$ at t_0 .

Since the 4 deputies are identical, $u_{i,max}$ are set to u_{max} and $\omega_{i,max}$ are set to ω_{max} for all $i \in I$. Furthermore, the simulation is set up such that there are no operational time constraints, and the durations of the

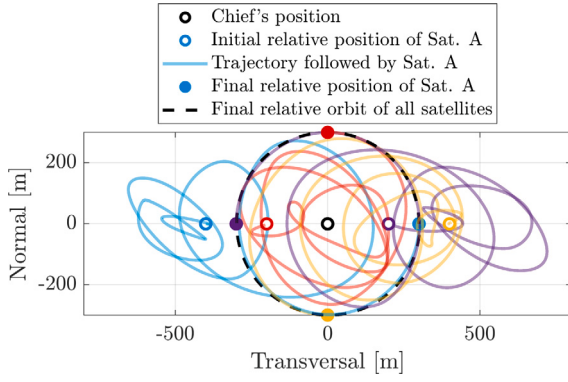


Fig. 8. Trajectories followed by each of the deputies throughout the coplanar-to-PCO maneuver.

Table 1

Initial and final (required) states for each of the deputies.

Satellite	$\mathbf{y}_0$ [m]	$\mathbf{y}_f$ [m]
Sat. A	$[0 \ -400 \ 0 \ 0 \ 0 \ 0]^T$	$[0 \ 0 \ 0 \ -150 \ 300 \ 0]^T$
Sat. B	$[0 \ -200 \ 0 \ 0 \ 0 \ 0]^T$	$[0 \ 0 \ -150 \ 0 \ 0 \ -300]^T$
Sat. C	$[0 \ 200 \ 0 \ 0 \ 0 \ 0]^T$	$[0 \ 0 \ 0 \ 150 \ -300 \ 0]^T$
Sat. D	$[0 \ 400 \ 0 \ 0 \ 0 \ 0]^T$	$[0 \ 0 \ 150 \ 0 \ 0 \ 300]^T$

Table 2

Parameters used in Problem 5 validation simulation.

$t_f - t_0$ [orbits]	T_f [orbits]	T_n [s]	u_{max} [$\mu\text{m/s}^2$]	R_{CA} [m]	n_{dir} [-]	c [-]
5	0.2	100	35	100	12	1.017

coast arcs are thus set to be all equal, i.e., $T_{n,l} = T_n \forall l \in \mathcal{L}$, where T_n is calculated according to the equality option in Eq. (5) for $\omega_{max} = 2^\circ/\text{s}$ and $T_{safety} = 10$ s. The forced motion periods are also fixed to a constant value, i.e., $T_{f,l} = T_f \forall l \in \mathcal{L}$, which was tuned around the results of the sensitivity analysis presented in Section 5. A full list of the parameters used in the simulation is given in Table 2. These parameters correspond to that of *Triton-Xheavy*, designed and manufactured by Luxspace.

Running the LP guidance scheme, Problem 5, over the described reconfiguration setting, profiles for the state vector, the control input vector, and the slack variables are obtained for each deputy at every step of the defined time vector. The trajectory followed by each of the deputies is depicted in Fig. 8, where the final relative orbit for all the deputies is seen to indeed resemble a PCO with a 300 m radius. Note that Fig. 8 contains legends only for Sat. A to explain the line and marker shapes' convention in use. Legends for other satellites would have had the same shapes as those of Sat. A, yet with their respective colors.

Note that Fig. 8 shows the 2-dimensional view of the trajectory's projection onto the T–N plane. In fact, Table 1 suggests that the maneuver is a general one, which requires both, in-plane and out-of-plane corrections. This could be verified by looking either at the 3-dimensional visualization of the trajectories, or at the ROE profile of the maneuver. The ROE profile of the reconfiguration maneuver is depicted in Fig. 9, which not only shows corrections of the in-plane variables as well as the out-of-plane ones, but also demonstrates how each of the relative orbital elements of all the deputies matches its set-point at the final time of the maneuver.

The optimality of the results were investigated through exploring the acceleration profiles of the maneuver. The radial, the transversal, and the normal components of the acceleration provided by each deputy are shown in Fig. 10.

One interesting aspect which can be seen in Fig. 10 is that the radial acceleration component is barely used even though it was not explicitly restricted in the formulation of Problem 5, simply because optimality

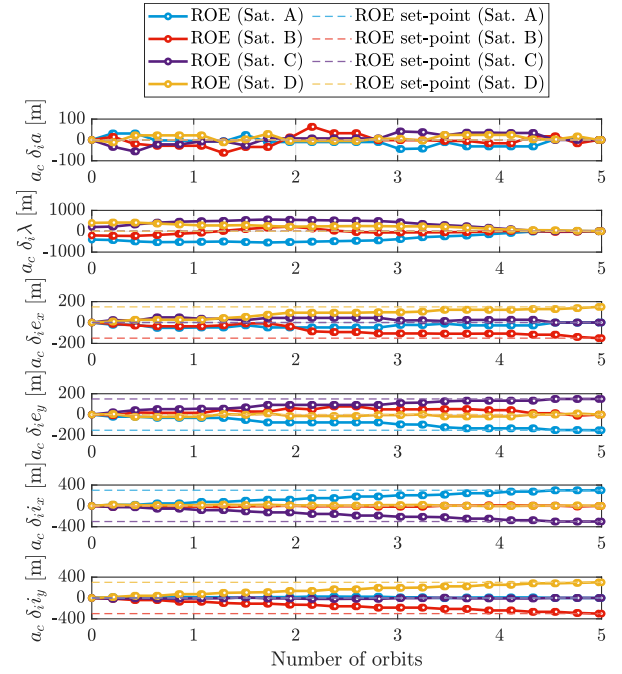


Fig. 9. ROE profile over the coplanar-to-PCO maneuver.

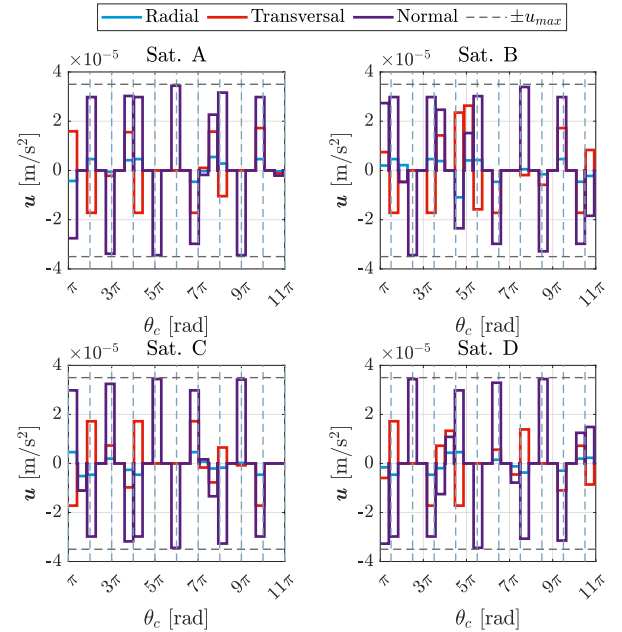


Fig. 10. Control acceleration vector profile over the coplanar-to-PCO maneuver. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

dictates not exploiting it. Leveraging mostly the transversal and normal components does indeed coincide with our initial expectations, which formed a foundation for approximating the projections of the maximum acceleration constraint on the R–T and the R–N planes with only 4 affine constraints for each plane, unlike the 12 constraints which approximate the projection of the same constraint on the T–N plane [16]. Moreover, the fact that the radial acceleration component is minimally utilized suggests that $\delta_i \lambda$ corrections are not performed directly through thrust, but rather through varying the value of $\delta_i a$ which, in-turn, changes $\delta_i \lambda$ according to the natural dynamics [7]. The small corrections of the relative eccentricity vector are conceivably done using the

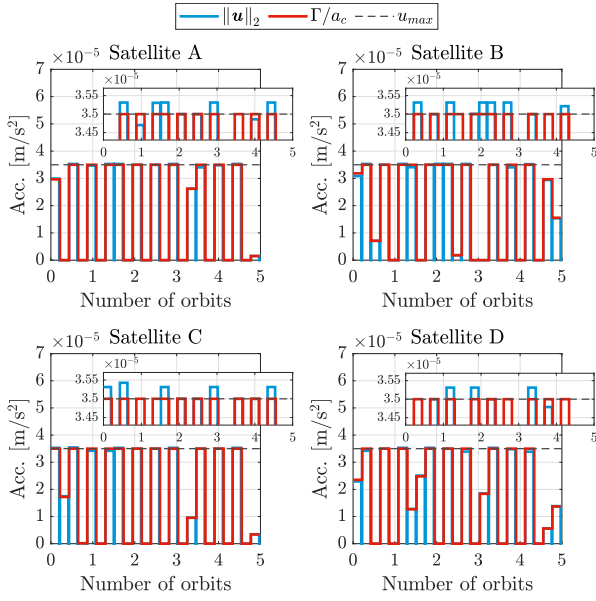


Fig. 11. Control acceleration vector and slack variables over the coplanar-to-PCO maneuver, Problem 4 (LP formulation). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

transversal acceleration component since it is half as expensive as using the radial one from the Delta-V point of view [7]. Although the in-plane corrections are larger in magnitude, they are mostly done by exploiting the natural dynamics as discussed earlier. Thus, it comes as no surprise that the normal acceleration component, which is responsible only for out-of-plane corrections, is the most component that has been utilized. In Fig. 10, the dashed teal vertical lines represent the Delta-V-optimal locations to provide impulsive acceleration in the normal direction to achieve the required out-of-plane corrections [7]. That said, the obtained solution, once again, matches the expectations for an optimal control profile as the normal acceleration component is seen in Fig. 10 to act as a bang–bang controller around the optimal locations, while being zero away from them.

The difference between the two problems, 4 and 5, is only the scaling factor, c , which scales down the feasibility region of the maximum acceleration constraint. The effect of this scaling coefficient can be clearly seen by looking into the acceleration (norm) profiles throughout the maneuver, which are depicted in Figs. 11 and 12 for Problems 4 and 5, respectively.

When c is set to unity, which is the case in Problem 4, the acceleration is prone to surpass the maximum level as seen in Fig. 11. Conversely, in Fig. 12, our claim that setting c to 1.017 (for $n_{\text{dir}} = 12$) guarantees that the L2 norm of the acceleration never exceeds the maximum acceleration level is verified (see the blue line). It is important to note that in the former case, the acceleration will never surpass the maximum acceleration by more than 1.7% (for $n_{\text{dir}} = 12$). Despite being minimal, this violation may lead to large errors over time in the ROE space which cannot be simply ignored when thrusters saturation is imposed, especially when the $\delta_i a$ variations are used to drive the in-plane variables to their set points. Figs. 11 and 12 also depict a scaled profile of the slack variable, Γ , throughout the reconfiguration maneuver (red lines), which verifies that constraint (42) never gets violated. Note that $\Gamma_{i,k}^* \neq a_c \|\mathbf{u}_{i,k}^*\|$ in the two LP formulations of the problem, however, $\Gamma_{i,k}^* = \|\mathbf{u}_{i,k}^*\|$ in the SOCP formulation, as stated by Eq. (28). This can be verified by inspecting the profiles of the L2 norm of the control acceleration vector when Problem 3 is solved. These profiles are depicted for the case-study reconfiguration scenario in Fig. 13, which shows that, for all the forced-motion periods, the

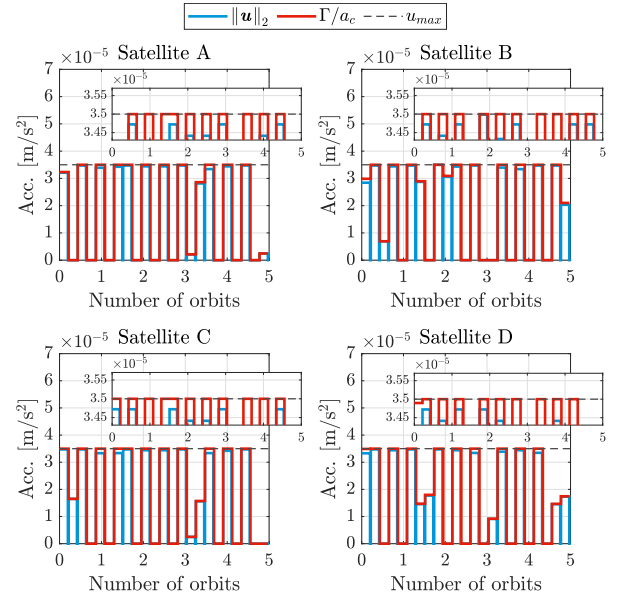


Fig. 12. Control acceleration vector and slack variables over the coplanar-to-PCO maneuver, Problem 5 (LP formulation), $c = 1.017$. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

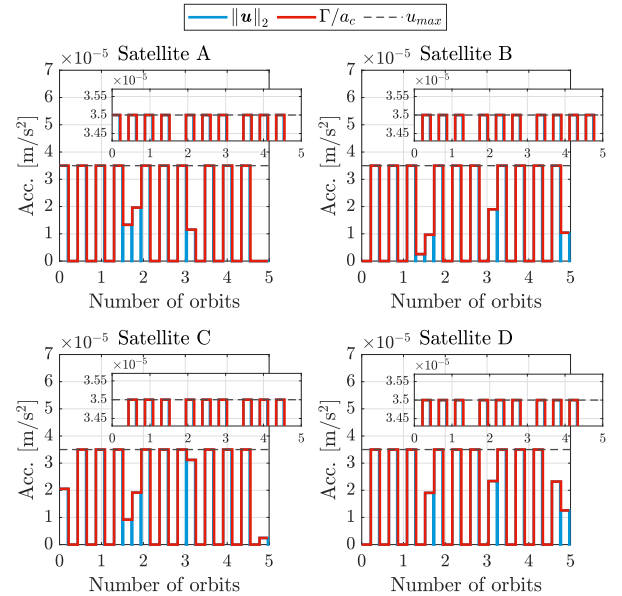


Fig. 13. Control acceleration vector and slack variables over the coplanar-to-PCO maneuver, Problem 3 (SOCP formulation).

optimal Γ is indeed equal to the acceleration level provided by each deputy.

A very important aspect of the modified guidance schemes, Problems 2 through 5, is that they all rely on sequential convex programming, which may require the problem to be solved multiple times before an optimal solution is obtained. The approach we adopted for solving Problems 2 through 5 is to solve the problem first without the collision avoidance constraints, and without the collision avoidance constraints, to obtain estimates for $\bar{\mathbf{y}}_{i,k}$. The problem is subsequently solved iteratively and the values of $\bar{\mathbf{y}}_{i,k}$ are updated at each iteration until any of the stopping criteria is met. The termination criteria were introduced in Section 3, but they are mentioned here once again to allow for a more elaborate discussion. The SCP termination criteria are (a) $\|\bar{\mathbf{y}}_{i,k} - \mathbf{y}_{i,k}\| \leq \epsilon$ at the current iteration; (b) The guidance profile

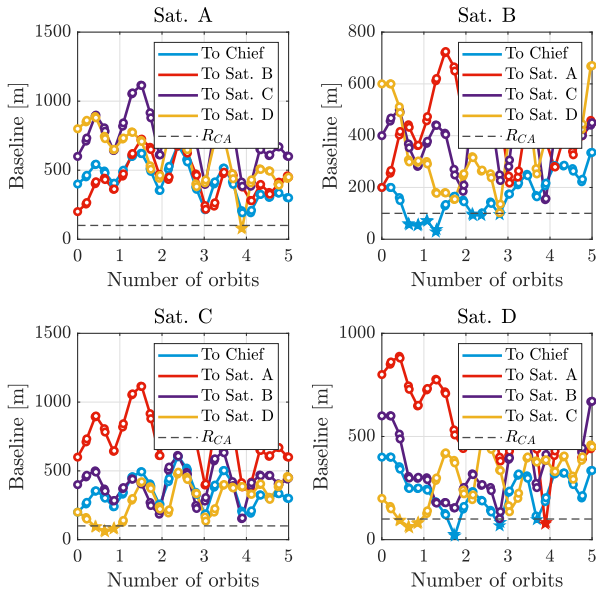


Fig. 14. Intersatellite distances over the coplanar-to-PCO maneuver, zeroth iteration.

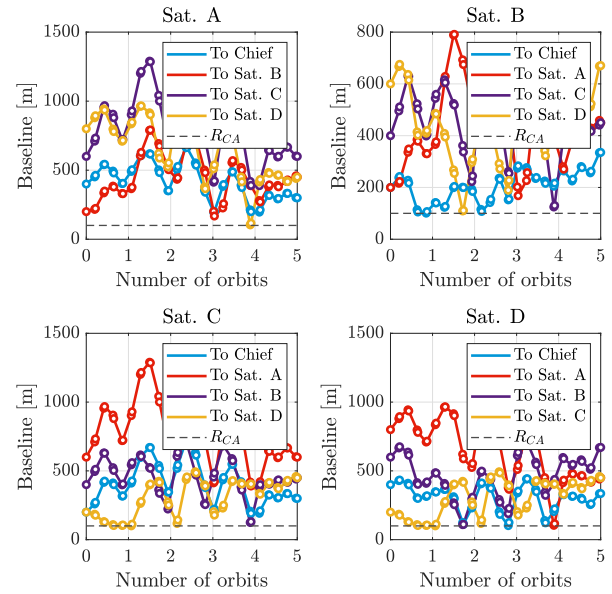


Fig. 15. Intersatellite distances over the coplanar-to-PCO maneuver, first iteration.

of the current iteration is collision free; (c) The user-defined maximum number of iteration is reached. Indeed, adopting only the first criterion is guaranteed to result in an optimal guidance profile, if the problem is feasible to begin with, yet at the cost of computational time, since a large number of iterations might be required if ϵ is chosen to be very small. Implementing the second criterion may lead to a sub-optimal solution, however, it considerably reduces the number of iterations required to solve the problem. The third criterion is nothing but a safeguard to ensure that the solver is not stuck in an infinite loop. In the context of our case-study, two simulations were run (using the formulation of Problem 5) where the first adopted the two stopping criteria, (a) and (c), while the second adopted all of the three. The former required 7 iterations to solve the problem (using $\epsilon = 1$ m), resulting in a solution which requires a total Delta-V of 1.8 m/s, while the latter needed only 1 iteration, providing a guidance profile which requires 1.82 m/s, which calls for only 1% increase in the required Delta-V. One thing which is worth noting is that, throughout our test simulations, adopting the second criterion almost always required a single iteration after the zeroth iteration in which the problem is solved without the collision avoidance constraints. This behavior is especially expected in cases where the solution of the zeroth iteration is almost collision-free. The intersatellite distances for our case-study are depicted in Fig. 14 for the zeroth SCP iteration and in Fig. 15 for the first SCP iteration, where the stars signify a location where collision avoidance is violated.

6.2. Solvers benchmark

The total Delta-V cost of the SOCP formulation, Problem 3, is expected to be less than what results from solving the convex QCQP problem, Problem 2. It is for this obvious reason that the QCQP scheme had to be modified into the SOCP formulation. Approximating the SOCP problem by an LP formulation, Problem 5, could be justified by many motives, among which is the fact that this relaxation puts the problem in the simplest form of convex programming, which is the easiest to implement. Moreover, the LP formulation is expected to require less solve time, especially for lower dimensional reconfiguration scenarios, i.e., those scenarios that involve a low number of sampling instances, and consequently involve a low number of constraints. The LP formulation is not expected to be very fast for higher dimensional reconfiguration scenarios, since one SOCP constraint is approximated

by 20 linear constraints; 12 in the T–N plane, 4 in the R–N plane, and 4 in the T–R plane. One other advantage of transforming an SOCP problem into an approximated linear program is to use dedicated solvers for linear programming that could not be possibly used in an SOCP context. This opens the door to many open-source non-commercial solvers that can handle only linear constraints such as GLPK, HiGHS, OSQP, and many others. Furthermore, some solvers which can handle SOCP problems still use dedicated algorithms for linear and quadratic programs that are not suitable for solving SOCP problems. Examples of such solvers include CPLEX, Gurobi, and Xpress, which usually favor simplex methods for linear programs, while using interior-point algorithms for QCQP and SOCP problems.

To get a clearer insight into the key differences between Problems 2, 3, and 5, four different reconfiguration scenarios were identified, and were used for the purpose of benchmarking different solvers when run over each of these formulations. The four scenarios were carefully selected to reflect wide ranges of formation and problem sizes, i.e., intersatellite distances, and number of variables and constraints, while being relevant from the applications point of view. The first reconfiguration requires the formation to go from a multi-pendulum configuration into a PCO, the second reshapes the formation from a PCO into a multi-cartwheel configuration, the third requests a reconfiguration from multi-cartwheel into a generic bounded configuration, and the fourth starts from a generic bounded configuration and ends up in a multi-pendulum one. The term “multi-pendulum” indicates that the deputies do not share a common pendulum relative orbit, similar to how “multi-cartwheel” signifies that the deputies are distributed across multiple cartwheel relative orbits. In contrast, a “PCO configuration” refers to all deputies sharing the same PCO. The reader is referred to Appendix B for the full details of the reconfiguration scenarios. It is to be emphasized that the assumed configuration geometries, i.e., PCO, multi-pendulum, and multi-cartwheel, are all of a great interest for remote sensing applications [31,32], and the identified reconfigurations may resemble a formation geometry change within a multi-static Synthetic Aperture Radar (SAR) mission. In the benchmark test, 15 of the most commonly used solvers for the types of problems in hand, according to the Mittelmann’s benchmark¹ and according to

¹ This benchmark used to act a decision tree for optimization software. It can be accessed through: <https://plato.asu.edu/bench.html>.

Table 3

Solvers overview.

Solver	Free	QCQP	SOCP	LP
GLPK [33]	✓	✗	✗	✓
CLP [34]	✓	✗	✗	✓
OSQP [35]	✓	✗	✗	✓
OOQP [36]	✓	✗	✗	✓
SCS [37]	✓	✗	✓	✓
ECOS [38]	✓	✗	✓	✓
IPOPT [39]	✓	✓	✓	✓
SCIP [40]	✓	✓	✓	✓
MOSEK [41]	✗ ^a	✓	✓	✓
Gurobi [42]	✗ ^a	✓	✓	✓
CPLEX [43]	✗ ^a	✓	✓	✓
COPT [44]	✗ ^a	✓	✓	✓
Knitro [45]	✗ ^b	✓	✓	✓
Xpress [46]	✗ ^c	✓	✓	✓
Matlab [47]	✗	✓	✓	✓

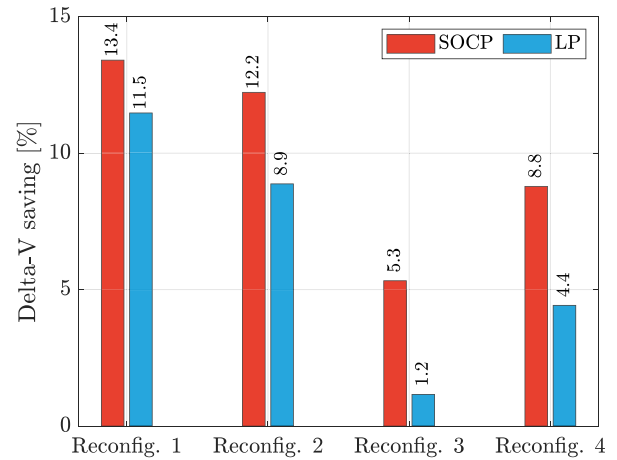
^a Offers a free academic license.

^b Offers a free academic licenses for the professor and the students during the time of the course.

^c Offers a community license which is limited to a maximum of 5000 variables and constraints.

the statistics of NEOS server,² were compared. An overview of the adopted 15 solvers is demonstrated in Table 3, where the problem types that could be handled by a solver are presented, together with information on whether the solver can be freely used in a commercial setting, or a software license needs to be purchased (on the date of writing this study, in June 2024). Note that every QCQP problem can be formulated as an SOCP one [28], nonetheless, the solvers that cannot handle QCQPs in their native form but can handle SOCP problems, e.g., SCS and ECOS, are marked with an ✗ mark in the QCQP column. The opposite is not true, however. Solvers which do not accept SOC constraints in their native form but rather need a reformulation of these constraints into the quadratic form are still marked with a ✓ mark in the SOCP column, since they still recognize the SOC nature of the reformulated constraints and treat them as such.

Using their default parameters, each solver was run 10 times over each reconfiguration scenario using the three problem formulations developed in this paper, i.e., the QCQP formulation, Problem 2, the SOCP formulation, Problem 3, and the LP formulation, Problem 5. The average time it took a solver to complete each of the 10 iterations was then recorded. The benchmark was conducted on a Windows-PC which comprises an Intel Core i9-10885H CPU with 16 cores and a clock speed of 2.4 GHz. The language that was used formulate the problems is Matlab, and the problems were passed to each solver either through its native Matlab Application Programming Interface (API). e.g., OSQP and SCS, or through a third-party Matlab interface, e.g., GLPK and CLP. Some interesting details on the 15 adopted solvers and on how they were interfaced with Matlab are presented in Appendix C. It is of utmost importance to declare that the purpose of this benchmark is not to compare the performance of the selected solvers. After all, tweaking the parameters of a solver may very probably result in a different performance. The purpose is rather to draw some recommendations as to which formulation is better used under which conditions, and also to showcase that using the different proposed formulations for the guidance problem has a major affect on both, the solve time of the problem, and the total Delta-V required for the maneuver. The results of the benchmark are presented in Table 4. The table also conveys some of the problem properties which are solver-independent. Namely the following properties are reported; the number of decision variables, the number of constraints excluding decision variables' bounds, the total Delta-V, and the number of required SCP iterations after the zeroth

² The statistics of NEOS server can be accessed through: <https://neos-server.org/neos/report.html>.

Fig. 16. Delta-V saving by adopting a different formulation from the QCQP.

iterations in order to arrive to a collision-free reconfiguration. The log files for the benchmark experiment are available as supplementary materials to this article.

Table 4 presents, in a quantitative manner, how much of the total Delta-V can be saved simply by adopting the SOCP or the LP formulations. This is shown graphically in Fig. 16, where the Delta-V savings (in percentage with respect to the Delta-V required by the QCQP problem) are depicted for the SOCP and the LP formulations and for the four reconfiguration scenarios employed in the benchmark experiment. It comes as no surprise that the SOCP formulation is requiring less total Delta-V than the LP formulation, since the SOCP problem is exploring a much larger action space than the LP is (refer to Figs. 2 and 3).

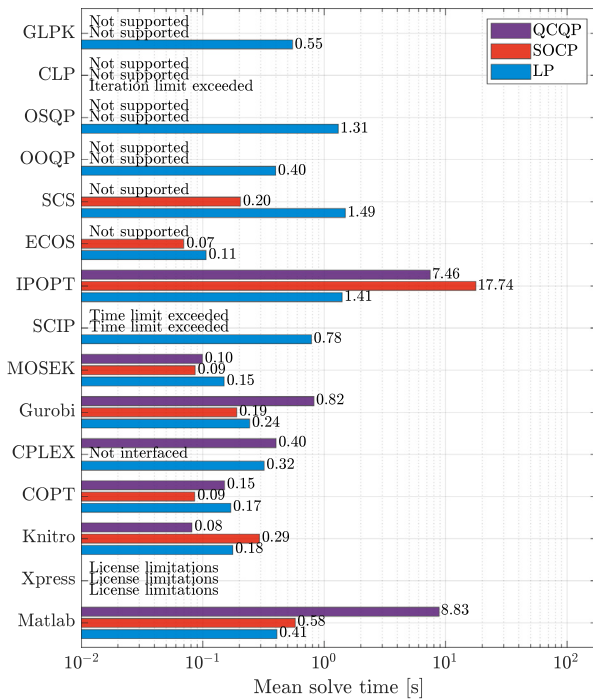
The results of Table 4 show how the number of variables is identical for the SOCP and the LP formulations, since they both consider all the entries of the Γ matrix as decision variables. Conversely, the number of constraints for the LP formulation is always much higher than the identical number of constraints for the QCQP and the SOCP formulations. The reason for this is that each instance of Eq. (23) is approximated by 20 linear constraints, Eqs. (40) and (41), in the LP formulation. The effect of adopting the various problem formulations on the number of required SCP iterations (after the zeroth one) is unclear, mainly due to the fact that SCP is set to terminate once a collision-free reconfiguration is obtained, which is a sub-optimal approach that turns out to be much faster, as has been pointed out earlier. All three formulations required no additional SCP iterations for Reconfiguration 3, as the zeroth iteration already produced a collision-free trajectory.

To get a better idea of how fast the different solvers return the optimal solution to the problem in hand, the solve time data, reported in Table 4, were averaged across the four reconfiguration scenarios, and the mean is depicted for each solver in Fig. 17. For a fair comparison, any solver that failed to produce a solution for one or more reconfiguration scenarios, due to reasons such as time limits or license restrictions, was excluded from the summary for the specific formulation where it was unable to generate a feasible result.

At a first glance on Fig. 17, it becomes very clear that, among all the solvers, those which are not dedicated to convex optimization, i.e., IPOPT, SCIP, and Matlab in the QCQP case, are the slowest, although they perform moderately in the LP case due to the simplicity of the problem. It is to be noted that many of the benchmarked solvers can still handle Non-Linear Programming (NLP) problems, e.g., CPLEX and Gurobi, however, they do recognize convex optimization problems, and hence, use dedicated algorithms that work best for the convex case. Most of the free solvers which are dedicated for linear and/or quadratic programming problems, i.e., GLPK, CLP, OSQP, and OOQP, present themselves as plausible candidates, although the commercial

Table 4
Benchmark results.

	Reconfiguration 1			Reconfiguration 2			Reconfiguration 3			Reconfiguration 4		
	QCQP	SQCP	LP	QCQP	SQCP	LP	QCQP	SQCP	LP	QCQP	SQCP	LP
N. variables	1608	1696	1696	2412	2544	2544	960	1012	1012	3816	4026	4026
N. constraints	1594	1594	3266	2661	2661	5169	676	676	1664	4221	4221	8211
Total ΔV [m/s]	1.11	0.96	0.98	3.03	2.66	2.76	1.32	1.25	1.31	4.71	4.29	4.50
N. SCP iter.	1	1	1	1	1	1	0	0	0	1	1	1
Solver	Solve time [s]											
GLPK	–	–	0.28	–	–	0.56	–	–	0.02	–	–	1.33
CLP	–	–	0.57	–	–	1.96	–	–	0.02	–	–	I.L. ^a
OSQP	–	–	1.43	–	–	0.92	–	–	0.35	–	–	2.52
OOQP	–	–	0.26	–	–	0.44	–	–	0.05	–	–	0.84
SCS	–	0.23	1.42	–	0.22	1.14	–	0.02	0.15	–	0.35	3.26
ECOS	–	0.04	0.08	–	0.08	0.13	–	0.01	0.01	–	0.16	0.21
IPOPT	1.61	20.24	0.98	3.56	20.25	1.62	6.71	10.11	0.25	17.94	20.34	2.77
SCIP	T.L. ^b	T.L. ^b	0.34	T.L. ^b	T.L. ^b	0.54	T.L. ^b	6.43	0.09	T.L. ^b	T.L. ^b	2.16
MOSEK	0.07	0.06	0.13	0.14	0.11	0.17	0.01	0.01	0.02	0.17	0.16	0.28
Gurobi	1.06	0.15	0.19	0.71	0.18	0.32	0.55	0.14	0.08	0.97	0.28	0.39
CPLEX	0.38	N.I. ^c	0.34	0.44	N.I. ^c	0.37	0.14	N.I. ^c	0.11	0.64	N.I. ^c	0.45
COPT	0.12	0.07	0.15	0.18	0.10	0.20	0.04	0.03	0.06	0.26	0.14	0.27
Knitro	0.06	0.23	0.13	0.08	0.32	0.19	0.01	0.03	0.03	0.16	0.58	0.36
Xpress	0.19	0.19	0.18	L.L. ^d	L.L. ^d	L.L. ^d	0.08	0.07	0.03	L.L. ^d	L.L. ^d	L.L. ^d
Matlab	2.69	0.22	0.14	7.91	0.55	0.52	0.96	0.04	0.01	23.77	1.50	0.95

^a The solver reached the Iteration Limit (I.L.) before finding a feasible solution.^b The solver reached the Time Limit (T.L.) (30 s) before finding a feasible solution.^c The solver is Not Interfaced (N.I.) for the type of problem in question. Refer to [Appendix C](#) for the reason.^d The solver did not solve the problem because of License Limitations (L.L.). The community license of Xpress is limited to a maximum of 5000 variables and constraints.**Fig. 17.** Benchmark summary.

ones appear to generally perform better for the same problem formulation, with an exception to Matlab's LP solver. Moreover, it is hardly surprising that dedicated conic solvers, namely SCS and ECOS, perform generally faster for the conic case, i.e., SQCP, than for the LP case. In fact, ECOS seems to be performing exceptionally well for a free solver for the two problem formulations it supports, i.e., SQCP and LP. There is little to no conclusion that can be drawn based on the results of the commercial solvers. For instance, some solvers perform the best for the QCQP formulation, some are favored for the SQCP case, and some are faster when handling the LP problems. The main

recommendation from [Table 4](#) as well as [Fig. 17](#) is to avoid the QCQP formulation since it is the most Delta-V intensive, while being generally slower to solve than either of the two other formulations, despite involving the least number of variables and constraints. One other takeaway which can be observed by looking into the details of [Table 4](#) is that the performance of the solvers over the LP problems is better when the number of constraints of the original SQCP problem is small, e.g., Reconfiguration 3. This claim is supported by the results of ECOS and specifically Gurobi. Conversely, for larger problem sizes, the SQCP problems are generally solved faster. It is for this reason that the LP formulation could only be recommended for small problem sizes, while the SQCP formulation is recommended for larger problems, although being harder to implement.

6.3. Limitations of the proposed schemes

The proposed guidance strategy can be, theoretically, used for an arbitrary number of deputies. However, employing the proposed guidance schemes in a constellation setting, i.e., with a very large number of satellites, might not be practical from the solve time point of view. To test the limitations of the centralized guidance plan, a last experiment was performed where the number of deputies is allowed to vary from 1 to 20 deputies, while the formation is requested to, once again, perform a Coplanar-to-PCO maneuver. The distance between every two consecutive deputies is set to 200 m for the initial configuration, while the radius of the final PCO is set 500 m, and, for a fair comparison, each of the 20 reconfigurations are allowed 10 orbits to complete. Since it has been established that the SQCP formulation is generally recommended, and since ECOS appears to be the fastest solver that handles the SQCP problem in our context, ECOS was run over the 20 reconfiguration scenarios and the solve time is depicted against the number of deputies in [Fig. 18](#).

It is obvious, and also quite conceivable, that the solve time is increasing exponentially with the number of deputies. It is for this reason that the proposed guidance schemes are only recommended for formations with small numbers of deputies. It might be worthwhile, however, to investigate distributed approaches for formations with large numbers of satellites.

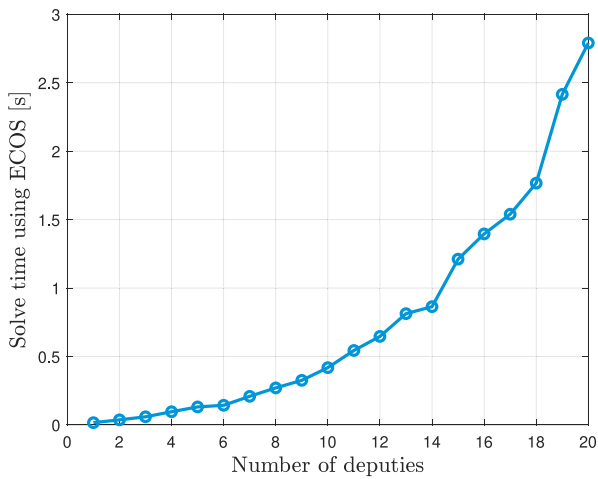


Fig. 18. Solve time as the number of deputies increase.

7. Conclusion

This article proposed centralized guidance schemes for the purpose of reconfiguring the relative orbits of multiple deputies around a chief satellite. One main characteristic of the considered formation reconfiguration is that each deputy is equipped with a single electric thruster, while the chief is uncontrolled, and is treated as the central processing unit for the formation trajectory optimization. In the development of the guidance strategies, the inter-deputy and the deputy-chief collision avoidance is considered, together with the fact that each deputy is under-actuated. The article proposed five different numerical optimization formulations for the guidance problem, where each formulation is a modified/relaxed version of the preceding one. The second (QCQP), the third (SOCQP), and the fifth (LP) formulations were identified as the most promising ones, and were involved in an experiment where fifteen solvers were benchmarked across four different reconfiguration scenarios. It was concluded from the results of the benchmark that the QCQP could never be recommended, as it requires the most total Delta-V change for a maneuver. Moreover, the SOCP formulation is generally recommended, for being fast to solve by many of the benchmarked solvers and also for requiring the least total Delta-V for the maneuver in question. The LP Formulation could only be recommended for small-scale problems, for being very efficient to solve, while being supported by the almost every numerical optimization solver there is.

CRedit authorship contribution statement

Ahmed Mahfouz: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Investigation, Formal analysis, Conceptualization. **Gabriella Gaias:** Writing – review & editing, Validation, Supervision, Methodology, Investigation, Formal analysis. **Florio Dalla Vedova:** Writing – review & editing, Supervision, Funding acquisition. **Holger Voos:** Writing – review & editing, Supervision, Resources, Project administration, Investigation, Funding acquisition, Formal analysis, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This research was funded in whole, or in part, by the Luxembourg National Research Fund (FNR), grant reference BRIDGES/19/MS/14302465. For the purpose of open access, and in fulfillment of the obligations arising from the grant agreement, the author has applied a Creative Commons Attribution 4.0 International (CC BY 4.0) license to any Author Accepted Manuscript version arising from this submission.

Appendix A. Affine relaxation of the maximum acceleration constraint

In order to put Problem 3 into the LP form, the SOC constraint in Eq. (23) had to be transformed into multiple piece-wise affine constraints. Specifically, Eq. (23) defines a constraining sphere with circular projections on the T–N, R–T, and R–N planes. These circular projections were approximated by regular polygons, where the polygons in the R–T and R–N planes form rhombuses, and the number of sides for the polygon in the T–N plane, n_{dir} , is left to the user to choose. This transformation is graphically presented in Fig. 2 and the details of it are discussed in [16]. It is important to highlight that the union of all the affine constraints forms a constraining polyhedron in 3D space. A notable aspect of this piece-wise affine relaxation is that, although the approximating polyhedron is designed to have projections on the three planes that remain within the original constraining sphere, the entire polyhedron is not necessarily contained within the sphere. Therefore, in Eq. (40) and Eq. (41), and in order to ensure that the constraining polyhedron lies with its entirety inside the original constraining sphere, the relaxed constraints had to be scaled using a constant scaling factor, c , which represents the ratio of the distance from the center of the sphere to the furthest point on the polyhedron relative to the radius of the sphere. In this appendix, a method to estimate the scaling factor, c , is presented for different values of n_{dir} .

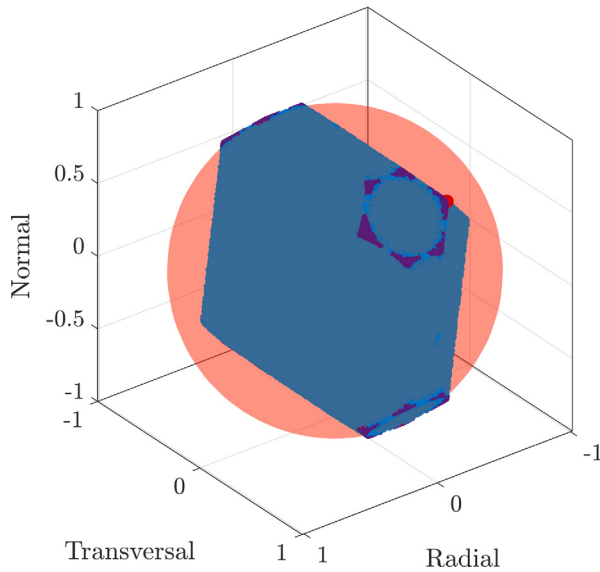
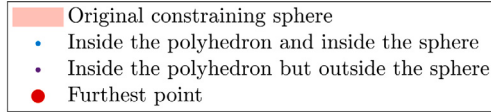
Given the value of n_{dir} , one approach to calculating the scaling factor involves using a 3D modeling package to model both the polyhedron and the sphere. By doing so, the distance from the center of the sphere to the furthest vertex of the polyhedron can be measured directly. However, this method is not scalable, as it requires constructing a 3D model for each specific value of n_{dir} . A more precise method takes advantage of the fact that the polyhedron is formed by the intersection of the $n_{dir} + 8$ planes that define the piece-wise linear constraints. The scaling factor c can be calculated accurately by constructing a convex hull from the vertices created by the intersection of each pair of planes with one of the three principal planes (R–T, R–N, and T–N). While this method guarantees precision, it requires implementing a 3-plane intersection algorithm. An alternative method involves a Monte Carlo simulation, where a large number of random points is generated within the cube that circumscribes the constraining sphere. Each point is tested to determine whether it lies within the polyhedron. The points that are inside the polyhedron are kept, and their distances from the origin are calculated. The maximum distance found is divided by the radius of the sphere to calculate the scaling factor. This method offers a rather easy implementation, however, it does not guarantee the precision of the scaling factor. Therefore, in this study, the Monte Carlo method is adopted, with an additional safety margin added to the calculated scaling factor to account for any potential inaccuracies.

The Monte Carlo simulation was run for 9 different values of n_{dir} , namely {4, 6, ..., 20}, and the results of these experiments are summarized in Table A.1. The table also includes the ratio of the area of relaxing polygon in the T–N plane relative to the area of the original constraining circle.

Table A.1 indicates that the rhombuses used to approximate the circles in the R–T and R–N planes cover approximately 64% of the area of the corresponding constraining circles. In contrast, the smallest polygon that covers at least 95% of the area of the constraining circle in the T–N plane is the polygon with 12 sides. The choice of $n_{dir} = 12$

Table A.1Monte Carlo results over different values of n_{dir} .

n_{dir}	Area ratio [%]	c
4	63.7	0.999
6	82.7	1.008
8	90.0	1.041
10	93.5	1.018
12	95.5	1.016
14	96.7	1.027
16	97.4	1.040
18	98.0	1.032
20	98.4	1.031

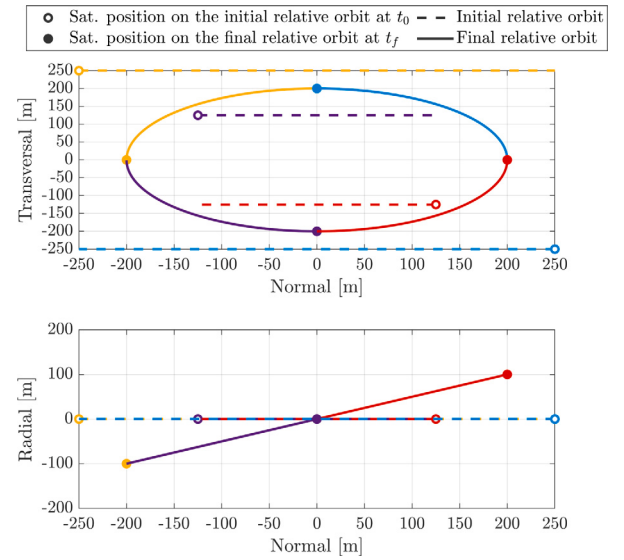
**Fig. A.1.** Monte Carlo results for $n_{dir} = 12$.

has additional advantages beyond just covering 95% of the constraining area in the T–N plane; it also requires the least amount of scaling among all the cases that cover at least 95% of the circumcircle.

Fig. A.1 depicts the results of the Monte Carlo simulation over the case when $n_{dir} = 12$. The resulting scaling factor of this experiment is 1.016, which is very close to the one calculated by the 3D modeling technique in Section 3, which was calculated to be 1.0166, and was approximated to 1.017. Indeed, the Monte Carlo result is less precise.

Appendix B. Reconfiguration scenarios used in the benchmark experiment

Four reconfiguration scenarios have been identified for the benchmark experiments. The initial orbit of the chief is assumed to be a sun-synchronous orbit which is shared in all of the four reconfigurations. The chief's orbit is parameterized by $\tilde{\mathbf{a}}_{c,0} = [6978 \text{ km } 90^\circ 10^{-3} 0 97.87^\circ 0^\circ]^T$ at t_0 . Furthermore, the deputies are assumed identical in all the scenarios, and consequently, $u_{i,max}$ are set to u_{max} for all $i \in \{1, 2, \dots, N\}$ with N being the number of deputies. Furthermore, the durations of the coast arcs are assumed to be all equal, i.e., $T_{n,l} = T_n \forall l \in \mathcal{L}$. The forced motion periods are also fixed to a constant value, i.e., $T_{f,l} = T_f \forall l \in \mathcal{L}$. The

**Fig. B.1.** Initial and final orbits of Reconfiguration 1.**Table B.1**

Initial and final (required) states for each of the deputies in Reconfiguration 1.

Satellite	$\mathbf{y}_0$ [m]	$\mathbf{y}_f$ [m]
Sat. A	$[0 \ -250 \ 0 \ 0 \ 0 \ -250]^T$	$[0 \ 0 \ 0 \ -100 \ 200 \ 0]^T$
Sat. B	$[0 \ -125 \ 0 \ 0 \ 0 \ -125]^T$	$[0 \ 0 \ -100 \ 0 \ 0 \ -200]^T$
Sat. C	$[0 \ 125 \ 0 \ 0 \ 0 \ 125]^T$	$[0 \ 0 \ 0 \ 100 \ -200 \ 0]^T$
Sat. D	$[0 \ 250 \ 0 \ 0 \ 0 \ 250]^T$	$[0 \ 0 \ 100 \ 0 \ 0 \ 200]^T$

simulation parameters that are used in the benchmark experiment are the same as those in **Table 2** and are shared in the four reconfigurations, except for the maneuver duration, which is defined separately for each reconfiguration scenario.

B.1. Reconfiguration 1 - multi-pendulum to PCO

In this reconfiguration scenario, 4 deputy satellites are assumed to be in a multi-pendulum configuration at t_0 and are required to be reconfigured into a Projected Circular Orbit (PCO) at t_f , where $t_f - t_0 = 5$ orbits. **Table B.1** summarizes the initial and final dimensional ROE vectors, $\mathbf{y}_0$ and $\mathbf{y}_f$, in meters for all the deputies.

The shape of the initial and final relative orbits of Reconfiguration 1 is depicted in **Fig. B.1**

B.2. Reconfiguration 2 - PCO to multi-cartwheel

In this reconfiguration scenario, 6 deputy satellites are assumed to be in a PCO configuration at t_0 and are required to be reconfigured into a multi-cartwheel configuration at t_f , where $t_f - t_0 = 5$ orbits. **Table B.2** summarizes the initial and final dimensional ROE vectors, $\mathbf{y}_0$ and $\mathbf{y}_f$, in meters for all the deputies.

The shape of the initial and final relative orbits of Reconfiguration 2 is depicted in **Fig. B.2**

B.3. Reconfiguration 3 - multi-cartwheel to generic

In this reconfiguration scenario, 4 deputy satellites are assumed to be in a multi-cartwheel configuration at t_0 and are required to be reconfigured into a generic configuration at t_f , where $t_f - t_0 = 3$ orbits. **Table B.3** summarizes the initial and final dimensional ROE vectors, $\mathbf{y}_0$ and $\mathbf{y}_f$, in meters for all the deputies. Note that the generic configuration is chosen such that the deputies have orthogonal eccentricity and

Table B.2

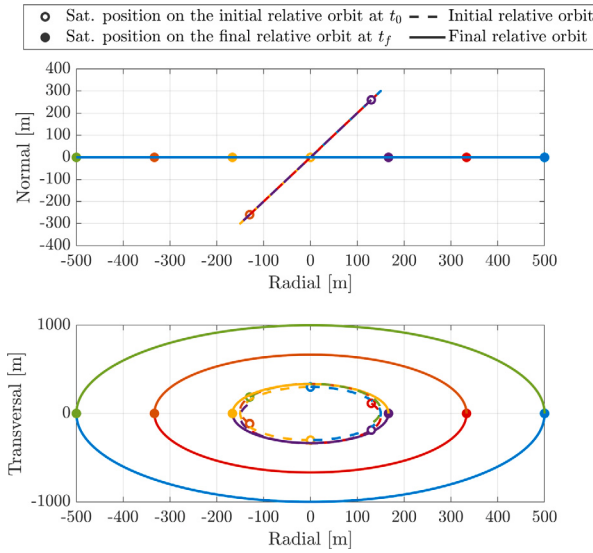
Initial and final (required) states for each of the deputies in Reconfiguration 2.

Satellite	$\mathbf{y}_0$ [m]	$\mathbf{y}_f$ [m]
Sat. A	$[0 \ 0 \ 0 \ -150 \ 300 \ 0]^T$	$[0 \ 0 \ -500 \ 0 \ 0 \ 0]^T$
Sat. B	$[0 \ -35.91 \ -129.90 \ -75 \ 150 \ -259.81]^T$	$[0 \ 0 \ -333.33 \ 0 \ 0 \ 0]^T$
Sat. C	$[0 \ -35.91 \ -129.90 \ 75 \ -150 \ -259.81]^T$	$[0 \ 0 \ -166.67 \ 0 \ 0 \ 0]^T$
Sat. D	$[0 \ 0 \ 0 \ 150 \ -300 \ 0]^T$	$[0 \ 0 \ 166.67 \ 0 \ 0 \ 0]^T$
Sat. E	$[0 \ 35.91 \ 129.90 \ 75 \ -150 \ 259.81]^T$	$[0 \ 0 \ 333.33 \ 0 \ 0 \ 0]^T$
Sat. F	$[0 \ 35.91 \ 129.90 \ -75 \ 150 \ 259.81]^T$	$[0 \ 0 \ 500 \ 0 \ 0 \ 0]^T$

Table B.3

Initial and final (required) states for each of the deputies in Reconfiguration 3.

Satellite	$\mathbf{y}_0$ [m]	$\mathbf{y}_f$ [m]
Sat. A	$[0 \ 0 \ -500 \ 0 \ 0 \ 0]^T$	$[0 \ 34.56 \ -250 \ 0 \ 0 \ -250]^T$
Sat. B	$[0 \ 0 \ -250 \ 0 \ 0 \ 0]^T$	$[0 \ 17.28 \ -125 \ 0 \ 0 \ -125]^T$
Sat. C	$[0 \ 0 \ 250 \ 0 \ 0 \ 0]^T$	$[0 \ -17.28 \ 125 \ 0 \ 0 \ 125]^T$
Sat. D	$[0 \ 0 \ 500 \ 0 \ 0 \ 0]^T$	$[0 \ -34.56 \ 250 \ 0 \ 0 \ 250]^T$

**Fig. B.2.** Initial and final orbits of Reconfiguration 2.

inclination vectors, which puts them in passively-unsafe relative orbits with respect to the chief satellite [48]. This makes the reconfiguration more challenging from the collision avoidance point of view.

The shape of the initial and final relative orbits of Reconfiguration 3 is depicted in Fig. B.3.

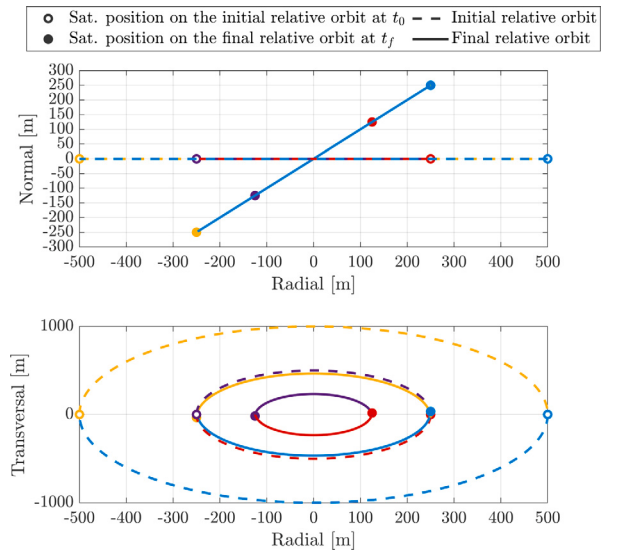
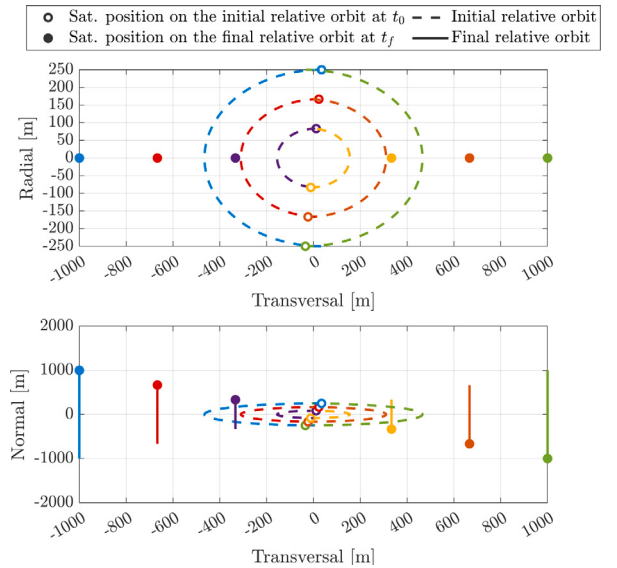
B.4. Reconfiguration 4 - generic to multi-pendulum

In this reconfiguration scenario, 6 deputy satellites are assumed to be in a generic configuration at t_0 and are required to be reconfigured into a multi-pendulum configuration at t_f , where $t_f - t_0 = 8$ orbits. Table B.4 summarizes the initial and final dimensional ROE vectors, $\mathbf{y}_0$ and $\mathbf{y}_f$, in meters for all the deputies. Again, in the initial generic configuration, the deputies are not in passively-safe relative orbits with respect to the chief, which makes avoiding collisions a non-trivial task for the guidance algorithm.

The shape of the initial and final relative orbits of Reconfiguration 4 is depicted in Fig. B.4

Appendix C. Solvers details

Fifteen solvers were chosen for the benchmark using the four reconfiguration scenarios defined in Appendix B. These solvers were chosen from the list of the most commonly used solvers on NEOS server in the

**Fig. B.3.** Initial and final orbits of Reconfiguration 3.**Fig. B.4.** Initial and final orbits of Reconfiguration 4.

category of LP and SOCP problems, and also from the solvers involved in the Mittelman's benchmark. Since NEOS server does not have a distinct category for QCQP, the solvers which are used for the SOCP problems are considered, since SOCP is the closest form of a convex programming formulation to the QCQP, and since any QCQP problem can be eventually formulated as an SOCP one. On top of the common solvers for each category, IPOPT was added for being one of the most commonly used free solvers for a variety of problem classes. Moreover, the solvers of the Matlab Optimization Toolbox were added to the list

Table B.4

Initial and final (required) states for each of the deputies in Reconfiguration 4.

Satellite	$\mathbf{y}_0$ [m]	$\mathbf{y}_f$ [m]
Sat. A	$[0 \ 34.56 \ -250 \ 0 \ 0 \ -250]^T$	$[0 \ -1000 \ 0 \ 0 \ 0 \ -1000]^T$
Sat. B	$[0 \ 23.04 \ -166.67 \ 0 \ 0 \ -166.67]^T$	$[0 \ -666.67 \ 0 \ 0 \ 0 \ -666.67]^T$
Sat. C	$[0 \ 11.52 \ -83.33 \ 0 \ 0 \ -83.33]^T$	$[0 \ -333.33 \ 0 \ 0 \ 0 \ -333.33]^T$
Sat. D	$[0 \ -11.52 \ 83.33 \ 0 \ 0 \ 83.33]^T$	$[0 \ 333.33 \ 0 \ 0 \ 0 \ 333.33]^T$
Sat. E	$[0 \ -23.04 \ 166.67 \ 0 \ 0 \ 166.67]^T$	$[0 \ 666.67 \ 0 \ 0 \ 0 \ 666.67]^T$
Sat. F	$[0 \ -34.56 \ 250.00 \ 0 \ 0 \ 250.00]^T$	$[0 \ 1000.00 \ 0 \ 0 \ 0 \ 1000.00]^T$

of solvers that are involved in our benchmark experiment under the name “Matlab”. A brief overview of the 15 solvers is presented in the following bullet points:

- GLPK (GNU Linear Programming Kit): GLPK is intended for solving large-scale linear programming, mixed integer programming (MIP), and other related problems.
- CLP (COIN-OR Linear Program solver): CLP is part of the COIN-OR project and is a solver meant for linear programming problems.
- OSQP (Operator Splitting Quadratic Program solver): OSQP is a numerical optimization package for solving convex quadratic programs. Linear programs, as a subset of quadratic programming problems, can be handled by the solver.
- OOQP (Object-Oriented software for Quadratic Programs): OOQP is an object-oriented software package for solving convex quadratic programming problems.
- SCS (Splitting Conic Solver): SCS is primarily meant for large-scale convex quadratic cone problems.
- ECOS (Embedded Conic Solver): ECOS is a numerical software for solving convex second-order cone programs.
- IPOPT (Interior Point OPTimizer): IPOPT is vastly used for large scale nonlinear optimization of continuous systems.
- SCIP (Solving Constraint Integer Programs): SCIP is one of the fastest non-commercial solvers for mixed integer programming and mixed integer nonlinear programming (MINLP). SCIP uses the SoPlex solver internally to solve linear programming problems.
- MOSEK: MOSEK is a commercial solver with an emphasis on solving large-scale sparse problems. It supports a wide range of problem types, including linear, quadratic, and convex nonlinear programs (NLP).
- Gurobi: Gurobi is a commercial solver that performs very well in large-scale optimization settings. It can solve various types of optimization problems, including linear as well as nonlinear programming problems.
- IBM ILOG CPLEX Optimization Studio (CPLEX): CPLEX is high-performance optimization solver for linear, mixed-integer and quadratic programming. It was named the for the simplex method implemented in the C programming language, although, at the moment, it supports other types of mathematical optimization and offers interfaces other than C.
- COPT (Cardinal optimizer): COPT is a mathematical optimization solver for large-scale problems. It supports many problem types including LP, SOCP, and Convex QCQP.
- Knitro: Knitro is a commercial software package for solving large scale nonlinear mathematical optimization problems. It supports a wide range of problem types, including linear and nonlinear programs, including non-convex NLPs.
- Xpress: Xpress is a mathematical optimization solver designed to solve linear programming, mixed integer programming, and other types of optimization problems.
- Matlab: Matlab is a high-level language and interactive environment that is used to perform computationally intensive tasks. In our context “Matlab” refers to the solvers offered by the Matlab Optimization Toolbox, which includes a dedicated linear programming solver, `linprog`, a dedicated conic programming

Table C.1

Benchmark solvers information.

Solver	Version	Interface with Matlab
GLPK	v4.62	GLPKmex ^a
CLP	v1.16.11	OPTI ^b
OSQP	v0.6.2	Developer's API
OOQP	v0.99.22	OPTI ^b
SCS	v3.2.3	SCS-Matlab ^c
ECOS	v2.0.10	Developer's API
IPOPT	v3.14.4	mexIPOPT ^d
SCIP	v5.0.1	OPTI ^b
MOSEK	v10.1.15	Developer's API
Gurobi	v11.0.2	Developer's API
CPLEX	v12.10.0.0	OPTI ^b
COPT	v7.1.3	COPT-MATLAB ^e
Knitro	v14.0.0	Developer's API
Xpress	v9.4.0	Developer's API
Matlab	v9.2 (R2021b)	No interface needed

^a The details of the interface as well as its source files are available on: <https://github.com/blegat/glpkmex>.

^b The details of the interface as well as its source files are available on: <https://github.com/jonathancurrie/OPTI>.

^c The details of the interface as well as its source files are available on: <https://github.com/bodono/scs-matlab>.

^d The details of the interface as well as its source files are available on: <https://github.com/ebertolazzi/mexIPOPT>.

^e The details of the interface as well as its source files are available on: <https://github.com/leavesgrp/COPT-MATLAB>.

function, `coneprog`, a general purpose NLP solver, `fmincon`, and many other solvers and capabilities.

Matlab has been used as a modeling language, and the problems were passed to each solver through an interface, either provided by the solver developer, by Matlab, or by a third party. Table C.1 contains the information about the version of each solver used in the benchmark, the interface that has been used between Matlab and the solver. The most interesting interfacing case is that of CPLEX, which is interfaced through the OPTI Matlab toolbox [49]. Although International Business Machines Corporation (IBM) used to provide a Matlab API for their CPLEX solver, this interface was discontinued, and the latest compatible Matlab version with that API was R2019b. Since the OPTI Matlab toolbox provides an interface that works flawlessly with newer Matlab versions, and since the authors used a newer Matlab version to model the problems, with no guarantee that the same code can be used in Matlab2019b to produce the parameters of each problem, it was decided to interface CPLEX with Matlab through OPTI, instead of moving to an older version of Matlab. It is to be noted, however, that SOCP problems are not handled by OPTI. Another thing to note is that most solvers have multiple internal methods that the user can choose from according to the problem in hand. Many solvers offer the possibility to choose the most suitable algorithm for the problem in hand, according to the pre-solve results. In the benchmark experiment, the algorithm was set to the default, which in most cases means that the solve method is automatically chosen by the solver. The specific algorithms used in the benchmark experiment are reported, by most solvers, in the benchmark log files, which are available as supplementary materials to this article.

Appendix D. Supplementary data

Supplementary material related to this article can be found online at <https://doi.org/10.1016/j.actaastro.2024.11.048>.

References

- [1] J.N. Pelton, S. Madry, Introduction to the small satellite revolution and its many implications, in: *Handbook of Small Satellites: Technology, Design, Manufacture, Applications, Economics and Regulation*, Springer International Publishing, Cham, 2020, pp. 3–31, http://dx.doi.org/10.1007/978-3-030-36308-6_1, URL https://doi.org/10.1007/978-3-030-36308-6_1.
- [2] S. Miller, M.L.R. Walker, J. Agolli, J. Dankanich, Survey and performance evaluation of small-satellite propulsion technologies, *J. Spacecr. Rockets* 58 (1) (2021) 222–231, <http://dx.doi.org/10.2514/1.A34774>.
- [3] D. O'Reilly, G. Herdrich, D.F. Kavanagh, Electric propulsion methods for small satellites: A review, *Aerospace* 8 (1) (2021) <http://dx.doi.org/10.3390/aerospace8010022>, URL <https://www.mdpi.com/2226-4310/8/1/22>.
- [4] V. Stanzione, B. Sabbatinelli, PLATINO project: A new Italian multi-application small satellite platform for highly competitive missions, in: 69th International Astronautical Congress (IAC), International Astronautical Federation, Bremen, Germany, 2018.
- [5] E. Helmeid, J. Buursink, M. Poppe, P. Ries, M. Gales, The integrated avionics unit - performance, innovation and application, 2022, Poster presentation at The 4S Symposium, Vilamoura, Portugal, 2022.
- [6] R. Larsson, R. Noteborn, P. Bodin, D. Simone, T. Karlsson, A. Carlsson, Autonomous formation flying in LEO - seven months of routine formation flying with frequent reconfigurations, in: 4th International Conference on Spacecraft Formation Flying Missions & Technologies, 2011, URL <https://elib.dlr.de/74364/>.
- [7] G. Gaias, S. D'Amico, Impulsive maneuvers for formation reconfiguration using relative orbital elements, *J. Guid. Control Dyn.* 38 (6) (2015) 1036–1049, <http://dx.doi.org/10.2514/1.G000189>.
- [8] G. Di Mauro, R. Bevilacqua, D. Spiller, J. Sullivan, S. D'Amico, Continuous maneuvers for spacecraft formation flying reconfiguration using relative orbit elements, *Acta Astronaut.* 153 (2018) 311–326, <http://dx.doi.org/10.1016/j.actaastro.2018.01.043>.
- [9] M. Chernick, S. D'Amico, New closed-form solutions for optimal impulsive control of spacecraft relative motion, *J. Guid. Control Dyn.* 41 (2) (2018) 301–319, <http://dx.doi.org/10.2514/1.G002848>.
- [10] F. Scala, G. Gaias, C. Colombo, M. Martín-Neira, Design of optimal low-thrust manoeuvres for remote sensing multi-satellite formation flying in low Earth orbit, *Adv. Space Res.* 68 (11) (2021) 4359–4378, <http://dx.doi.org/10.1016/j.asr.2021.09.030>.
- [11] A. De Vittori, M.F. Palermo, P.D. Lizia, R. Armellin, Low-thrust collision avoidance maneuver optimization, *J. Guid. Control Dyn.* 45 (10) (2022) 1815–1829, <http://dx.doi.org/10.2514/1.G006630>.
- [12] G. Gaias, J.-S. Ardaens, Flight demonstration of autonomous noncooperative rendezvous in low Earth orbit, *J. Guid. Control Dyn.* 41 (6) (2018) 1337–1354, <http://dx.doi.org/10.2514/1.G003239>.
- [13] G. Gaias, S. D'Amico, J.-S. Ardaens, Generalised multi-impulsive manoeuvres for optimum spacecraft rendezvous in near-circular orbit, *Int. J. Space Sci. Eng.* 3 (1) (2015) 68–88, <http://dx.doi.org/10.1504/IJSPACESE.2015.069361>.
- [14] E. Belloni, S. Silvestrini, J. Prinetto, M. Lavagna, Relative and absolute on-board optimal formation acquisition and keeping for scientific activities in high-drag low-orbit environment, *Adv. Space Res.* 73 (11) (2024) 5595–5613, <http://dx.doi.org/10.1016/j.asr.2023.07.051>, Recent Advances in Satellite Constellations and Formation Flying. URL <https://www.sciencedirect.com/science/article/pii/S0273117723006002>.
- [15] A. Mahfouz, G. Gaias, D.M.K.K. Venkateswara Rao, H. Voos, Autonomous optimal absolute orbit keeping through formation flying techniques, *Aerospace* 10 (11) (2023) <http://dx.doi.org/10.3390/aerospace10110959>, URL <https://www.mdpi.com/2226-4310/10/11/959>.
- [16] A. Mahfouz, G. Gaias, F.D. Vedova, H. Voos, Fuel-optimal formation reconfiguration by means of unidirectional low-thrust propulsion system, *J. Guid. Control Dyn.* (2024) <http://dx.doi.org/10.2514/1.G008214>, in press.
- [17] D. Menzio, A. Mahfouz, D. Vedova, et al., Formation design of an inter-satellite link demonstration mission, in: *Proceedings of the 11th International Workshop on Satellite Constellations & Formation Flying*, Milan, IT, 2022.
- [18] A. Mahfouz, D. Mezio, F. Dalla-Vedova, H. Voos, Relative state estimation for LEO formations with large inter-satellite distances using single-frequency GNSS receivers, in: *Proceedings of the 11th International Workshop on Satellite Constellations & Formation Flying*, Milan, IT, 2022.
- [19] A. Mahfouz, D. Mezio, F. Dalla-Vedova, H. Voos, GNSS-based baseline vector determination for widely separated cooperative satellites using L1-only receivers, *Adv. Space Res.* (2023) <http://dx.doi.org/10.1016/j.asr.2023.06.037>.
- [20] D.A. Vallado, *Fundamentals of Astrodynamics and Applications*. Vol. 12, Springer Science & Business Media, 2001.
- [21] G. Gaias, C. Colombo, M. Lara, Analytical framework for precise relative motion in low Earth orbits, *J. Guid. Control Dyn.* 43 (5) (2020) 915–927, <http://dx.doi.org/10.2514/1.G004716>.
- [22] D. Morgan, S.-J. Chung, F.Y. Hadaegh, Model predictive control of swarms of spacecraft using sequential convex programming, *J. Guid. Control Dyn.* 37 (6) (2014) 1725–1740, <http://dx.doi.org/10.2514/1.G000218>.
- [23] T. Pippia, V. Preda, S. Bannani, T. Keviczky, Reconfiguration of a satellite constellation in circular formation orbit with decentralized model predictive control, in: *Proceedings of the 2022 CEAS EuroGNC Conference*, Berlin, Germany, 2022, CEAS-GNC-2022-027.
- [24] G. Gaias, M. Lovera, Trajectory design for proximity operations: The relative orbital elements' perspective, *J. Guid. Control Dyn.* 44 (12) (2021) 2294–2302, <http://dx.doi.org/10.2514/1.G006175>.
- [25] I.M. Ross, Space trajectory optimization and L1-optimal control problems, in: P. Gurfil (Ed.), *Modern Astrodynamics*, in: Elsevier Astrodynamics Series, vol. 1, Butterworth-Heinemann, 2006, pp. 155–VIII, [http://dx.doi.org/10.1016/S1874-9305\(07\)80008-2](http://dx.doi.org/10.1016/S1874-9305(07)80008-2).
- [26] D. Venkateswara Rao, T.H. Go, Optimization of aircraft spin recovery maneuvers, *Aerosp. Sci. Technol.* 90 (2019) 222–232, <http://dx.doi.org/10.1016/j.ast.2019.04.046>, URL <https://www.sciencedirect.com/science/article/pii/S1270963818329092>.
- [27] A. Abdelkariem, Y. Jia, D. Gorges, Optimization of vehicle-to-grid profiles for peak shaving in microgrids considering battery health, in: *IECON 2023-49th Annual Conference of the IEEE Industrial Electronics Society*, IEEE, 2023, pp. 1–6, <http://dx.doi.org/10.1109/IECON51785.2023.10312094>.
- [28] M.S. Lobo, L. Vandenberghe, S. Boyd, H. Lebret, Applications of second-order cone programming, *Linear Algebra Appl.* 284 (1) (1998) 193–228, [http://dx.doi.org/10.1016/S0024-3795\(98\)10032-0](http://dx.doi.org/10.1016/S0024-3795(98)10032-0), International Linear Algebra Society (ILAS) Symposium on Fast Algorithms for Control, Signals and Image Processing. URL <https://www.sciencedirect.com/science/article/pii/S0024379598100320>.
- [29] S.P. Boyd, L. Vandenberghe, *Convex optimization*, Cambridge University Press, 2004, pp. 136–137.
- [30] J.M. Elble, N.V. Sahinidis, Scaling linear optimization problems prior to application of the simplex method, *Comput. Optim. Appl.* 52 (2) (2012) 345–371, <http://dx.doi.org/10.1007/s10589-011-9420-4>.
- [31] G. Fasano, A. Renga, M. D'Errico, Formation geometries for multistatic SAR tomography, *Acta Astronaut.* 96 (2014) 11–22, <http://dx.doi.org/10.1016/j.actaastro.2013.11.024>, URL <https://www.sciencedirect.com/science/article/pii/S009457651300427X>.
- [32] W. Wang, D. Wu, R. Sun, H. Baoyin, Optimal projected circular orbit for spacecraft formation flying near a slowly rotating asteroid, *IEEE Trans. Aerosp. Electron. Syst.* 58 (6) (2022) 5483–5493, <http://dx.doi.org/10.1109/TAES.2022.3173247>.
- [33] A. Makhori, GNU linear programming kit (GLPK), 2012, URL <https://www.gnu.org/software/glpk/>. (Accessed 21 May 2024).
- [34] J. Forrest, S. Vigerske, T. Ralphs, L. Hafer, H.G. Santos, Jan-Willem, M. Saltzman, B. Kristjansson, A. King, P. Bonami, R. Luies, S. Brito, et al., coin-or/Clp: Release releases/1.17.9, Zenodo, 2023, <http://dx.doi.org/10.5281/zenodo.10041272>.
- [35] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, S. Boyd, OSQP: An operator splitting solver for quadratic programs, *Math. Program. Comput.* 12 (4) (2020) 637–672, <http://dx.doi.org/10.1007/s12532-020-00179-2>.
- [36] E.M. Gertz, S.J. Wright, Object-oriented software for quadratic programming, *ACM Trans. Math. Software* 29 (1) (2003) 58–81, <http://dx.doi.org/10.1145/641876.641880>, URL <https://doi.org/10.1145/641876.641880>.
- [37] B. O'Donoghue, E. Chu, N. Parikh, S. Boyd, Conic optimization via operator splitting and Homogeneous self-dual embedding, *J. Optim. Theory Appl.* 169 (3) (2016) 1042–1068, URL <http://stanford.edu/~boyd/papers/scs.html>.
- [38] A. Domahidi, E. Chu, S. Boyd, ECOS: An SOCP solver for embedded systems, in: *European Control Conference, ECC*, 2013, pp. 3071–3076.
- [39] A. Wächter, L.T. Biegler, On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming, *Math. Program.* 106 (1) (2006) 25–57, <http://dx.doi.org/10.1007/s10107-004-0559-y>, URL <https://doi.org/10.1007/s10107-004-0559-y>.
- [40] A. Gleixner, L. Eifler, T. Gally, G. Gamrath, P. Gemander, R.L. Gottwald, G. Hendel, C. Hojny, T. Koch, M. Miltenberger, B. Müller, M.E. Pfetsch, C. Puchert, D. Rehfeldt, F. Schlösser, F. Serrano, Y. Shinano, J.M. Viernickel, S. Vigerske, D. Weninger, J.T. Witt, J. Witzig, The SCIP Optimization Suite 5.0, Technical report, Optimization Online, 2017, URL http://www.optimization-online.org/DB_HTML/2017/12/6385.html.
- [41] MOSEK ApS, The MOSEK optimization toolbox for MATLAB manual. Version 10.0., 2024, URL <https://docs.mosek.com/10.0/toolbox/index.html>.
- [42] Gurobi Optimization, LLC, Gurobi optimizer reference manual. Version 11.0., 2024, URL <https://www.gurobi.com/documentation/current/refman/index.html>.
- [43] CPLEX, IBM ILOG, V12. 1: User's manual for CPLEX, *Int. Bus. Mach. Corp.* 46 (53) (2009) 157.
- [44] D. Ge, Q. Huangfu, Z. Wang, J. Wu, Y. Ye, Cardinal optimizer (COPT) user guide, 2023, <https://guide.coap.online/copt/en-doc>.
- [45] R.H. Byrd, J. Nocedal, R.A. Waltz, Knitro: An integrated package for nonlinear optimization, in: *Large-Scale Nonlinear Optimization*, Springer US, Boston, MA, 2006, pp. 35–59, http://dx.doi.org/10.1007/0-387-30065-1_4, URL https://doi.org/10.1007/0-387-30065-1_4.

- [46] Fair Isaac Corporation, FICO, FICO xpress optimizer reference manual. Version 9.4., 2024, URL <https://www.fico.com/fico-xpress-optimization/docs/latest/solver/optimizer/HTML/GUID-3BEAAE64-B07F-302C-B880-A11C2C4AF4F6.html>.
- [47] The MathWorks Inc., Optimization Toolbox version: 9.2 (R2021b), The MathWorks Inc., Natick, Massachusetts, United States, 2021, URL <https://www.mathworks.com/products/optimization.html>.
- [48] G. Borelli, G. Gaias, C. Colombo, Safe guidance scheme for proximity operations forced motion, J. Guid. Control Dyn., 1–18, <http://dx.doi.org/10.2514/1.G008179>.
- [49] J. Currie, D.I. Wilson, OPTI: Lowering the barrier between open source optimizers and the industrial MATLAB user, in: N. Sahinidis, J. Pinto (Eds.), Foundations of Computer-Aided Process Operations, Savannah, Georgia, USA, 2012.