A convex-programming-based guidance algorithm to capture a tumbling object on orbit using a spacecraft equipped with a robotic manipulator

Original
A convex-programming-based guidance algorithm to capture a tumbling object on orbit using a spacecraft equipped with a robotic manipulator / Virgili-Llop, J., Zagaris, C., Zappulla, R., Bradstreet, A., Romano, M.. - In: THE INTERNATIONAL JOURNAL OF ROBOTICS RESEARCH. - ISSN 0278-3649. - 38:1(2019), pp. 40-72. [10.1177/0278364918804660]

Publisher:
SAGE Publications

Published
DOI:10.1177/0278364918804660

Article

The International Journal of
Robotics Research
2019, Vol. 38(1) 40–72
© The Author(s) 2018
Article reuse guidelines:
sagepub.com/journals-permissions
DOI: 10.1177/0278364918804660
journals.sagepub.com/home/ijr

A convex-programming-based guidance algorithm to capture a tumbling object on orbit using a spacecraft equipped with a robotic manipulator

**Josep Virgili-Llop, Costantinos Zagaris, Richard Zappulla II,
Andrew Bradstreet and Marcello Romano**

Abstract

An algorithm to guide the capture of a tumbling resident space object by a spacecraft equipped with a robotic manipulator is presented. A solution to the guidance problem is found by solving a collection of convex programming problems. As convex programming offers deterministic convergence properties, this algorithm is suitable for onboard implementation and real-time use. A set of hardware-in-the-loop experiments substantiates this claim. To cast the guidance problem as a collection of convex programming problems, the capture maneuver is divided into two simultaneously occurring sub-maneuvers: a system-wide translation and an internal re-configuration. These two sub-maneuvers are optimized in two consecutive steps. A sequential convex programming procedure, overcoming the presence of non-convex constraints and nonlinear dynamics, is used on both optimization steps. A proof of convergence is offered for the system-wide translation, while a set of structured heuristics—trust regions—is used for the optimization of the internal re-configuration sub-maneuver. Videos of the numerically simulated and experimentally demonstrated maneuvers are included as supplementary material.

Keywords

Computational guidance and control, convex programming, space robotics, on-orbit servicing, space debris

1. Introduction

The capture of a tumbling resident space object (RSO) by a spacecraft equipped with a robotic manipulator is envisioned for many on-orbit servicing missions (Flores-Abad et al., 2014; Nanjangud et al., 2018; Shan et al., 2016). Guiding the chaser spacecraft during these capture maneuvers is challenging, because the chaser is a multibody system with nonlinear dynamics and the maneuver is constrained by limited actuation capabilities as well as subject to collision avoidance constraints.

Given the problem's relevance, multiple guidance approaches have been proposed. Some authors assume that the chaser initiates the capture maneuver from a hold position where the target's grapple fixture is within the chaser's manipulator reach. In this scenario, guidance strategies that only actuate the manipulator joints and minimize the base reaction during the pre-grappling, grappling (impact), and post-grappling phases are preferred. Prime examples of this work can be found in Yoshida et al. (2006) and more recently in Flores-Abad et al. (2016).

The time-varying, and often large, keep-out zones imposed by the tumbling target and its appendages (e.g., solar panels) may rule out the existence of a safe holding position in the immediate vicinity of the target. In general, the chaser must initiate the capture maneuver from a hold position sufficiently far away and execute a full roto-translation maneuver, actuating both the manipulator and the base-spacecraft. In addition, the rotational state of a tumbling object cannot be accurately predicted well in advance if the initial conditions or inertia properties are not precisely known (Hanßmann, 1999). This limitation indicates that a guidance approach, suitable for onboard

---

Mechanical and Aerospace Engineering Department, Naval Postgraduate School, Monterey, CA, USA

Corresponding author:
Josep Virgili-Llop, Mechanical and Aerospace Engineering Department, Naval Postgraduate School, 1 University Circle, Monterey, CA 93943, USA.
Email: jvirgili@nps.edu

implementation and capable of (re-)planning these maneuvers in real-time, can be advantageous for a wide range of targets and applications.

Sampling-based motion-planning techniques are a promising approach to solve this type of problem (Karaman and Frazzoli, 2011). In particular, Persson and Sharf (2015) achieved substantial success in their experiments using off-board computation and relatively simple vehicle geometries, and target motions. Although optimization-based approaches have traditionally appeared to be too computationally intensive to be embedded in onboard computers and used for real-time motion planning, they have also received considerable attention. Jacobsen et al. (2002) and Lampariello (2010) frame the capture problem as a nonlinear optimization problem and require ample time to find a solution. In an effort to reduce the computation time, an off-line generated look-up table is used by Lampariello and Hirzinger (2013) to seed the optimizer, but their approach falls short of producing an algorithm suitable for onboard implementation.

In this paper, an optimization-based guidance algorithm suitable for onboard implementation and real-time use is presented. The proposed algorithm can handle complex collision avoidance and line-of-sight constraints, satisfies control limitations, and minimizes the control effort. The use of this algorithm is illustrated through numerical simulations and evidence of its real-time capabilities is obtained by hardware-in-the-loop experiments on a planar air bearing test bed (Zappulla II et al., 2017b). Videos of the numerically simulated and experimentally demonstrated maneuvers are included as supplementary material.

The complexity of the presented experimental demonstration surpasses previous experimental work known to the authors, significantly advancing the demonstrated state-of-the-art related to the capture of tumbling RSOs by spacecraft equipped with robotic manipulators. Previous work using planar air bearing test beds (Schwartz et al., 2003) includes Alexander and Cannon (1989), Umetani and Yoshida (1989), Ullman and Cannon (1993), Chen and Cannon (1994), Russakow et al. (1995), Yoshida (1994), Nahon et al. (1995), Menon et al. (2007), Toglia et al. (2011), Wilde et al. (2016), Virgili-Llop et al. (2016b), and Sabatini et al. (2017). Other relevant experimental work on orbit or on other types of experimental facilities (Xu et al., 2011) includes Oda et al. (1996), Ogilvie et al. (2008), Aghili (2008), Xu et al. (2009), and Persson and Sharf (2015) among others.

To obtain a computationally tractable guidance algorithm, the capture maneuver is first divided into two simultaneously occurring sub-maneuvers: a system-wide translation and an internal re-configuration. The sub-maneuvers are then solved in two consecutive optimization steps, the combination of which provides a solution to the original guidance problem. To overcome the presence of non-convex constraints and nonlinear dynamics, a sequential convex programming procedure is used on both optimization steps. As a result, a solution to the guidance problem is obtained by solving a collection of convex programming problems, which are computationally tractable, offer deterministic convergence properties, and can be solved in polynomial-time by interior-point algorithms (Boyd and Vandenberghe, 2004; Nesterov and Nemirovskii, 1994). The proposed guidance algorithm is thus able to rapidly generate a solution, allowing to repeatedly solve the guidance problem as the maneuver progresses.

The use of convex programming techniques in embedded systems to solve aerospace guidance and control problems in real-time has recently shown remarkable promise (Liu et al., 2017), particularly in the areas of powered soft-landing (Açıkmeşe et al., 2013; Szmuk and Acikmese, 2018), and spacecraft proximity operations (Lu and Liu, 2013; Watterson et al., 2016). Of special relevance is the work by Verscheure et al. (2009) and Misra and Bai (2017) applying convex programming techniques to robotic manipulator control problems. Verscheure et al. (2009) introduced a convex relaxation for the minimum-time trajectory tracking problem whereas Misra and Bai (2017) found a convex formulation for the control of a manipulator mounted on a free-floating spacecraft (i.e., internal re-configuration).

In this paper, a convergence proof for the sequential convex programming procedure used during the system-wide translation optimization is offered. This proof extends the work of Lu and Liu (2013) and Morgan et al. (2016) to non-convex keep-out zone constraints. The convergence of the sequential convex programming procedure for the internal re-configuration sub-maneuver is not guaranteed and relies on structured heuristics, namely trust regions. Within this optimization step, an explicitly convex line-of-sight constraint expression, similar to that reported by Kim et al. (2010), is provided.

The material presented here builds upon previous research by the authors and provides, for the first time, a complete and comprehensive overview of the proposed guidance approach. A primitive version of the guidance approach and preliminary experimental results were included in Virgili-Llop et al. (2017a,b). An improved guidance algorithm is used here, with its derivation and properties presented in details. Additional new contributions include the convergence proof, an explicitly convex line-of-sight formulation, extensive numerical simulations, and comprehensive experimental results.

Although the work presented in this paper exclusively focuses on the maneuver's guidance, a navigation system providing relative state estimates is indispensable for the success of the capture maneuver. The interested reader is referred to Opromolla et al. (2017) for a recent survey on relative navigation for spacecraft proximity maneuvering.

The rest of this paper is organized as follows. The original optimal control problem and the high-level guidance approach are presented in Section 2. The optimization of the two sub-maneuvers is discussed in Sections 3 and 4. In Section 5, a numerical simulation case study is presented, illustrating the performance of the proposed guidance.

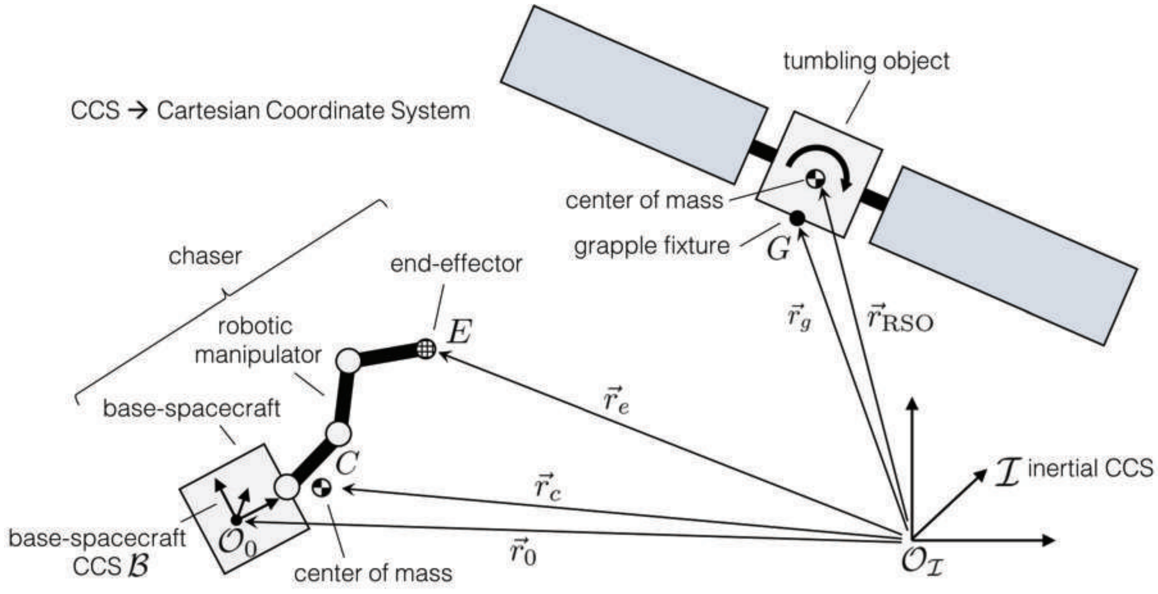


Fig. 1. Illustration of the problem.

Then, the results of a set of hardware-in-the-loop experiments are reported in Section 6. Finally, concluding remarks are offered.

2. Problem statement and approach

In this section, the original guidance problem is presented, followed by a high-level description of the proposed approach as well as the considered assumptions and their implications. The capture maneuver involves a chaser spacecraft, equipped with a robotic manipulator, capturing a tumbling RSO. The maneuver has to satisfy various constraints while minimizing the control effort. A notional overview of the problem is shown in Figure 1.

When formulating the guidance problem, the following underlying assumptions are made.

- **A**.1 Both the chaser and target RSO are composed of rigid bodies moving in three-dimensional space.
- **A**.2 Environmental forces (gravity gradient, solar radiation pressure, etc.) as well as the relative orbital dynamic effects are neglected. This can be justified by the short duration of the maneuver and close proximity of the two vehicles. This assumption implies that an orbiting reference frame can act as an inertial reference frame.
- **A**.3 The state and inertia properties of the chaser and tumbling RSO are known.
- **A**.4 The target RSO has a designated grapple fixture.
- **A**.5 The manipulator's grappling configuration and its motion during the final seconds of the maneuver, $t \geqslant t_{\mathrm{ps}}$, is pre-set (as discussed further later). The t_{ps} time is fixed and marks the transition to the pre-set manipulator motion.
- **A**.6 The chaser's mass remains constant during the maneuver, i.e., the amount of propellant used for the maneuver is small when compared with the chaser's mass.

2.1. Nonlinear optimal control problem

The equations of motion of a spacecraft with a robotic manipulator can be written, in canonical form, as

$$\boldsymbol{H}\dot{\boldsymbol{u}} + \boldsymbol{C}\boldsymbol{u} = \boldsymbol{\tau} \tag{1}$$

where $\boldsymbol{u}$ denotes the generalized velocities, $\boldsymbol{\tau}$ the generalized forces, $\boldsymbol{H}$ the generalized inertia matrix, and $\boldsymbol{C}$ the generalized convective inertia matrix.

The generalized coordinates can be divided between those coordinates referring to the base-spacecraft $|\cdot|_0$ and those referring to the manipulator $|\cdot|_m$:

$$\boldsymbol{u} = \begin{bmatrix} \boldsymbol{u}_0 \\ \boldsymbol{u}_m \end{bmatrix} \quad \boldsymbol{\tau} = \begin{bmatrix} \boldsymbol{\tau}_0 \\ \boldsymbol{\tau}_m \end{bmatrix} \tag{2}$$

The number of degrees of freedom of the manipulator is denoted by n_{DoF} and, thus, $\boldsymbol{u}_m, \boldsymbol{\tau}_m \in \mathbb{R}^{n_{\mathrm{DoF}}}$. The angular or linear displacements of the manipulator joints are denoted by $\boldsymbol{\theta}_m \in \mathbb{R}^{n_{\mathrm{DoF}}}$.

Without lack of generality, all vectors—unless specified explicitly—are projected into the inertial Cartesian coordinate system (CCS) $\mathcal{I}$, so, in general, for a vector $\vec{a}$, the 3×1 column matrix containing the components of the projection of $\vec{a}$ into $\mathcal{I}$ is denoted by $\boldsymbol{a} \in \mathbb{R}^3$. Note that non-bold, italic symbols with an arrow (or hat) are strictly reserved for physical vector quantities (or unit vectors). Bold symbols are used for matrices, including the projection of vectors in a particular reference frame (i.e., a 3×1 matrix).

The base-spacecraft generalized velocities with respect to the inertial frame, $\boldsymbol{u}_0 \in \mathbb{R}^6$, contain the base-spacecraft's linear ($\dot{\boldsymbol{r}}_0 \in \mathbb{R}^3$) and angular ($\boldsymbol{\omega}_0 \in \mathbb{R}^3$) velocity. Equivalently, the base-spacecraft generalized forces, $\boldsymbol{\tau}_0 \in \mathbb{R}^6$, contain the forces ($\boldsymbol{f}_0 \in \mathbb{R}^3$) and torques ($\boldsymbol{n}_0 \in \mathbb{R}^3$) applied to the base-spacecraft:

$$\boldsymbol{u}_0 = \begin{bmatrix} \dot{\boldsymbol{r}}_0 \\ \boldsymbol{\omega}_0 \end{bmatrix} \quad \boldsymbol{\tau}_0 = \begin{bmatrix} \boldsymbol{f}_0 \\ \boldsymbol{n}_0 \end{bmatrix} \tag{3}$$

Finally, let $\boldsymbol{r}_0 \in \mathbb{R}^3$ denote the base-spacecraft position and $\boldsymbol{q}_0 \in S^3$ denote a unit quaternion ($\boldsymbol{q}_0 \in \{\mathbb{H} | \|q_0\| = 1\}$), representing the orientation of the base-spacecraft CCS $\mathcal{B}$ with respect to the inertial CCS $\mathcal{I}$. The differential kinematics of the attitude quaternion can be expressed as follows:

$$\dot{\boldsymbol{q}}_0 = \frac{1}{2} \boldsymbol{\omega}_0 \otimes \mathbf{q}_0 \tag{4}$$

where $\boldsymbol{\omega}_0$ is promoted to a pure quaternion (i.e., with zero scalar part) and with $\otimes$ denoting the quaternion product.

The function to be minimized by the optimization procedure is formulated as the following quadratic cost:

$$J = \int_0^{t_f} \boldsymbol{\tau}^{\mathrm{T}} \boldsymbol{W} \boldsymbol{\tau} dt \tag{5}$$

with $\boldsymbol{W}$ denoting a $6 + n_{\mathrm{DoF}} \times 6 + n_{\mathrm{DoF}}$ positive-definite weight matrix and t_f the maneuver duration, or final time.

During the capture maneuver the manipulator displacements and torques are constrained to lie within certain limits:

$$\boldsymbol{\theta}_{m\,\min} \leqslant \boldsymbol{\theta}_m \leqslant \boldsymbol{\theta}_{m\,\max} \tag{6}$$

$$\boldsymbol{\tau}_{m\,\min} \leqslant \boldsymbol{\tau}_m \leqslant \boldsymbol{\tau}_{m\,\max} \tag{7}$$

The forces and torques of the base-spacecraft are also bounded:

$$\|\boldsymbol{f}_0\| \leqslant f_{0\,\max} \tag{8}$$

$$\|\boldsymbol{n}_0\| \leqslant n_{0\,\max} \tag{9}$$

with $\|\cdot\|$ denoting the L_2 norm.

To avoid a collision with the target, a keep-out zone is enforced. Let the chaser's body be defined by the closed set S_{chaser}, and the target RSO by the closed set S_{RSO}. The keep-out zone constraint can then be formulated as follows:

$$S_{\mathrm{chaser}} \cap S_{\mathrm{RSO}} = \varnothing \tag{10}$$

A line-of-sight constraint is imposed throughout the maneuver, as it is assumed that the chaser needs to keep track of the target's grapple fixture for navigation purposes. If this constraint is not required, or if it is only needed for a portion of the maneuver, the constraint can be removed or only enforced during the applicable period. The line-of-sight constraint is formulated considering that a body-mounted sensor, with boresight $\hat{\boldsymbol{v}}$, needs to be within a cone, with half-angle ϕ, that points towards the grapple fixture G. Figure 2 illustrates this line-of-sight constraint, formulated using the following equation:

$$\boldsymbol{r}_d^{\mathrm{T}} \hat{\boldsymbol{v}} \geqslant \|\boldsymbol{r}_d\| \cos\phi \tag{11a}$$

$$\boldsymbol{r}_d = \boldsymbol{r}_g - \boldsymbol{r}_s \tag{11b}$$

with $\boldsymbol{r}_s$ denoting the projection of the position vector from the origin of the inertial CCS to the point S, where the sensor is located.

Assumption **A**.5 imposes a known manipulator motion from t_{ps} until the end of the maneuver at t_f:

$$\boldsymbol{\theta}_m(t) = \boldsymbol{\theta}_m^{\mathrm{ps}}(t) \quad \text{for } t \in [t_{\mathrm{ps}}, t_f] \tag{12}$$

For a successful capture, the location and velocity of the end-effector $\boldsymbol{r}_e$ and the grappling fixture $\boldsymbol{r}_g$ must match:

$$\boldsymbol{r}_e(t_f) = \boldsymbol{r}_g(t_f) \tag{13}$$

$$\dot{\boldsymbol{r}}_e(t_f) = \dot{\boldsymbol{r}}_g(t_f) \tag{14}$$

$$\boldsymbol{\omega}_0(t_f) = \boldsymbol{\omega}_{\mathrm{RSO}}(t_f) \tag{15}$$

with $\boldsymbol{\omega}_{\mathrm{RSO}} \in \mathbb{R}^3$ denoting the RSO's angular velocity with respect to the inertial frame.

Finally, the complete optimal control problem is written as follows.

Problem 1. Original nonlinear optimal control problem.[1]

$$\min : \int_0^{t_f} \boldsymbol{\tau}^{\mathrm{T}} \boldsymbol{W} \boldsymbol{\tau} dt \tag{5}$$

$$\text{s.t.} : \boldsymbol{H}\dot{\boldsymbol{u}} + \boldsymbol{C}\boldsymbol{u} = \boldsymbol{\tau} \tag{1}$$

$$\dot{\boldsymbol{q}}_0 = \frac{1}{2} \boldsymbol{\omega}_0 \otimes \mathbf{q}_0 \tag{4}$$

$$\boldsymbol{\theta}_{m\,\min} \leqslant \boldsymbol{\theta}_m \leqslant \boldsymbol{\theta}_{m\,\max} \tag{6}$$

$$\boldsymbol{\tau}_{m\,\min} \leqslant \boldsymbol{\tau}_m \leqslant \boldsymbol{\tau}_{m\,\max} \tag{7}$$

$$\|\boldsymbol{f}_0\| \leqslant f_{0\,\max} \tag{8}$$

$$\|\boldsymbol{n}_0\| \leqslant n_{0\,\max} \tag{9}$$

$$S_{\mathrm{chaser}} \cap S_{\mathrm{RSO}} = \varnothing \tag{10}$$

$$\boldsymbol{r}_d^{\mathrm{T}} \hat{\boldsymbol{v}} \geqslant \|\boldsymbol{r}_d\| \cos\phi \tag{11a}$$

$$\boldsymbol{\theta}_m(t) = \boldsymbol{\theta}_m^{\mathrm{ps}}(t) \qquad \text{for } t \in [t_{\mathrm{ps}}, t_f] \tag{12}$$

$$\boldsymbol{r}_e(t_f) = \boldsymbol{r}_g(t_f) \tag{13}$$

$$\dot{\boldsymbol{r}}_e(t_f) = \dot{\boldsymbol{r}}_g(t_f) \tag{14}$$

$$\boldsymbol{\omega}_0(t_f) = \boldsymbol{\omega}_{\mathrm{RSO}}(t_f) \tag{15}$$

Problem 1 is a nonlinear optimal control problem that is, in general, difficult to solve. The multibody dynamics (1) and quaternion differential kinematics (4) are nonlinear, and the keep-out zone constraint (10) is non-convex.

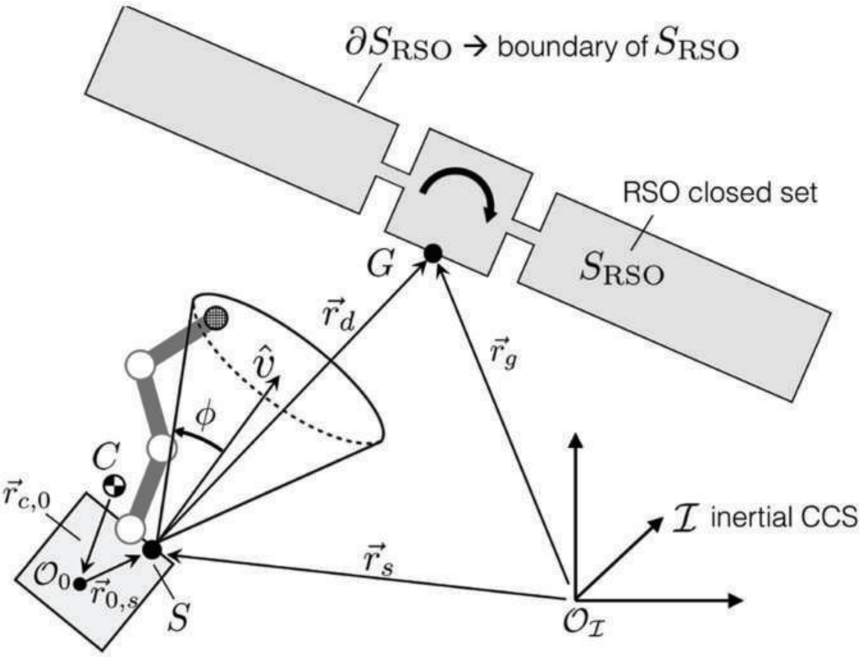


Fig. 2. Line-of-sight constraint.

2.2. Proposed guidance approach

To make Problem 1 more tractable, the capture maneuver is divided into two sub-maneuvers: a system-wide translation and an internal re-configuration. The system-wide translation concerns only the motion of the chaser system center of mass. The internal re-configuration focuses on the manipulator motion and chaser re-orientation around the system's center of mass. These two sub-maneuvers are individually optimized in a two-step process. Figure 3 illustrates the division.

Despite being dynamically coupled through Equation (1), the two concurrent sub-maneuvers can be decoupled, allowing one of them to be optimized first and independently from the other. Afterwards, the remaining sub-maneuver is optimized using the results of the first optimization step to restore the dynamic coupling. Recombining both solutions forms a full capture maneuver, and an admissible point of the original optimal control problem.

In the proposed approach, the first optimization step solves the system-wide translation and the second one solves the internal re-configuration. The system-wide translation sub-maneuver is subjected to the simple whole-body translational dynamics and can be easily decoupled from the internal re-configuration. With the system-wide translation solved, the more complex internal re-configuration can be solved, forcing this second sub-maneuver to accommodate the effects caused by a moving center of mass.

Remark 1. Consecutively optimizing these two sub-maneuvers decouples their cost functions. The proposed optimization order implicitly prioritizes the minimization of the system-wide translation cost, at the expense of a higher internal re-configuration cost. This prioritization can be justified by the fact that translational maneuvers make use of the spacecraft's limited supply of propellant, whereas internal re-configuration maneuvers can be completed using momentum exchange devices and the manipulator's joint actuators.

In summary, the proposed approach consists of two optimization steps. Step 1: Optimize a decoupled system-wide translation sub-maneuver. Step 2: Optimize the internal re-configuration sub-maneuver, using the results of Step 1 to restore the dynamic coupling. The complete capture maneuver is obtained when the solutions of the two optimization steps are combined. In the next two sections these two optimization steps are detailed.

3. Step 1: system-wide translation optimization

In this section, the details of the first optimization step are presented and discussed. This optimization step results in an optimal control problem, which is solved using a sequential convex programming procedure with guaranteed convergence to a locally optimal solution.

The system-wide translation, solely focusing on the motion of the system's center of mass, exploits the simple whole-body translational dynamics. If m denotes the chaser's mass, $\boldsymbol{r}_c \in \mathbb{R}^3$ the projection of the center of mass position vector, and $\boldsymbol{f}_c \in \mathbb{R}^3$ the forces applied to it, then the equation of motion can be written as $\boldsymbol{f}_c = m\ddot{\boldsymbol{r}}_c$. However, because the thrusters on the chaser's base-spacecraft are the only actuators capable of imparting linear momentum (i.e., $\boldsymbol{f}_0 = \boldsymbol{f}_c$), the equation of motion can be written as

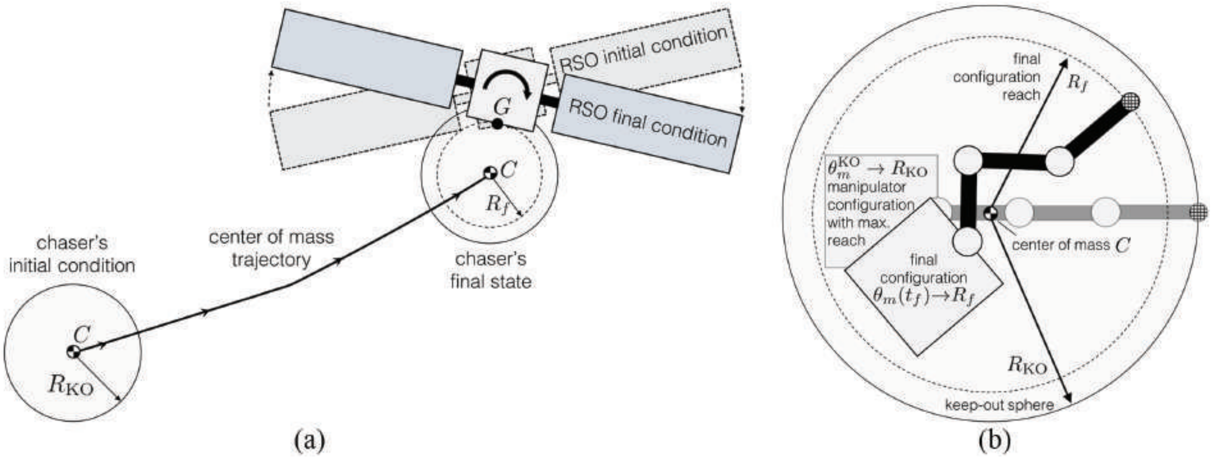


Fig. 3. Sub-maneuvers: (a) system-wide translation; (b) internal re-configuration.

$$\boldsymbol{f}_0 = m\ddot{\boldsymbol{r}}_c \tag{16}$$

To formulate the optimal control problem, the quadratic cost

$$J_1 = \int_0^{t_f} \boldsymbol{f}_0^{\mathrm{T}} \boldsymbol{W}_1 \boldsymbol{f}_0 dt \tag{17}$$

with $\boldsymbol{W}_1 \in \mathbb{R}^{3\times 3}$ denoting a positive-definite weight matrix is used.

Remark 2. A quadratic cost function is chosen because it generates smoother maneuvers, leading to more numerically stable solutions. An L_1-norm-based cost function, minimizing the amount of propellant used, produces more aggressive maneuvers with a bang-off-bang control profile.

This sub-maneuver is constrained with the relevant constraints of Problem 1. These are: maximum force (Equation 8), keep-out zones (Equation 10), and the terminal conditions (Equations 13 and 14). The maximum force constraint can be applied as-is, but the others need to be modified to decouple them from the internal re-configuration sub-maneuver.

As the chaser's internal configuration is undefined during this first optimization step, S_{chaser} needs to enclose all admissible manipulator configurations, $\boldsymbol{\theta}_m$, and base-spacecraft orientations $\boldsymbol{q}_0$. Such an enclosing set can be modeled as a sphere of radius R, centered at the center of mass C:

$$S_{\mathrm{chaser}} = \left\{ \boldsymbol{r} \in \mathbb{R}^3 | \|r - \boldsymbol{r}_c\| \leq R \right\} \tag{18}$$

As illustrated in Figure 3(b), there is a configuration $\boldsymbol{\theta}_m^{\mathrm{KO}}$ that defines the maximum sphere's radius R_{KO}. This radius is defined by the maximum possible distance between any part of the chaser and its center of mass for all admissible manipulator configurations, and it is equivalent, in the absence of any large chaser appendages, to the free-floating reachable workspace radius (Umetani and Yoshida, 2001). The fixed and constant radius R_{KO} is then used to define S_{chaser} during the initial period of the maneuver $t \in [0, t_{ps})$.

Assumption **A**.5 imposes a manipulator motion during the last period of the maneuver $t \in [t_{ps}, t_f]$, as expressed in Equation (12). During this final period the manipulator configuration is known and thus a smaller radius for S_{chaser} can be used, as illustrated in Figure 4:

$$R_{\mathrm{KO}}^{\mathrm{ps}}(t) \leq R_{\mathrm{KO}} \quad \text{for } t \in [t_{\mathrm{ps}}, t_f] \tag{19a}$$

$$R_{\mathrm{KO}}^{\mathrm{ps}}(t_f) = R_f \tag{19b}$$

In conclusion, the chaser's closed set S_{chaser} is defined, during the two periods, as follows:

$$S_{\mathrm{chaser}} = \begin{cases} \left\{ \boldsymbol{r} \in \mathbb{R}^3 | \|\boldsymbol{r} - \boldsymbol{r}_c\| \leq R_{\mathrm{KO}} \right\}, & t \in [0, t_{\mathrm{ps}}) \\ \left\{ \boldsymbol{r} \in \mathbb{R}^3 | \|\boldsymbol{r} - \boldsymbol{r}_c\| \leq R_{\mathrm{KO}}^{\mathrm{ps}}(t) \right\}, & t \in [t_{\mathrm{ps}}, t_f] \end{cases} \tag{20}$$

In contrast, the target RSO's closed set, S_{RSO}, is directly defined by the RSO's geometry, as shown in Figure 2.

The keep-out zone constraint in Equation 10 can be reformulated using the signed distance d between S_{chaser} and S_{RSO} (Schulman et al., 2014) (see Figure 5):

$$d = d^+ - d^- \tag{21a}$$

$$d^+ = \inf\left\{ \left\|\vec{d}\right\| | \left(S_{\mathrm{chaser}} + \vec{d} \right) \cap S_{\mathrm{RSO}} \neq \varnothing \right\} \tag{21b}$$

$$d^- = \inf\left\{ \left\|\vec{d}\right\| | \left(S_{\mathrm{chaser}} + \vec{d} \right) \cap S_{\mathrm{RSO}} = \varnothing \right\} \tag{21c}$$

where inf denotes the inferior of a set and $\vec{d}$ the vector between the two closest supporting points $P_{\mathrm{chaser}} \in \partial S_{\mathrm{chaser}}$ and $P_{\mathrm{RSO}} \in \partial S_{\mathrm{RSO}}$, as illustrated in Figure 5:

$$\vec{d} = \begin{cases} \text{vector from } P_{\mathrm{RSO}} \text{ to } P_{\mathrm{chaser}} & \text{for } d > 0 \\ \text{vector from } P_{\mathrm{chaser}} \text{ to } P_{\mathrm{RSO}} & \text{for } d < 0 \end{cases} \tag{22}$$

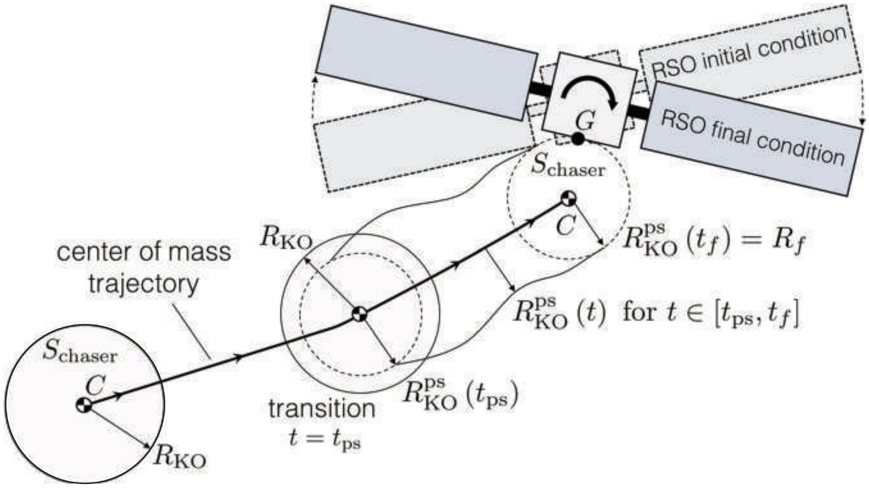


Fig. 4. Transition of chaser's enclosing sphere.

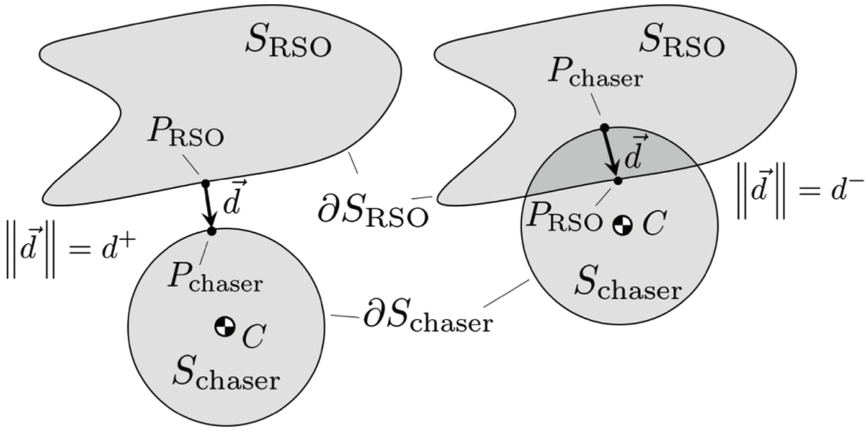


Fig. 5. Signed distances d^+ and d^- between S_{chaser} and S_{RSO}.

The signed distance d is positive when the two sets are not in contact and negative when they intersect. Using d, the keep-out zone constraint can be expressed as

$$d \geqslant 0 \tag{23}$$

In a general case, the Gilbert–Johnson–Keerthi algorithm (Gilbert et al., 1988) can be used to obtain d^+, whereas the expanding polytope algorithm (Bergen, 1999) can be used to efficiently compute d^-.

The terminal constraint also relies on assumption **A.**5. The manipulator's final configuration $\boldsymbol{\theta}_m^{\text{ps}}(t_f)$ is assumed to be known, allowing to pre-compute the distance R_f from the center of mass to the end-effector (see Figures 3 and 4). The terminal constraint on the center of mass then becomes

$$\|\boldsymbol{r}_g(t_f) - \boldsymbol{r}_c(t_f)\| = R_f \tag{24}$$

To ensure a zero relative velocity between the chaser and the target's center of mass, the following terminal constraint is imposed:

$$\dot{\boldsymbol{r}}_c(t_f) = \boldsymbol{\omega}^{\times}_{\text{RSO}}(t_f)\big(\boldsymbol{r}_c(t_f) - \boldsymbol{r}_{\text{RSO}}(t_f)\big) \tag{25}$$

with $\boldsymbol{r}_{\text{RSO}} \in \mathbb{R}^3$ denoting the projection of the RSO's center of mass position vector and the $|\cdot|^{\times}$ operator representing the left-hand side matricial equivalent of the vector cross-product:

$$\boldsymbol{\omega}^{\times} = \begin{bmatrix} 0 & -\omega_3 & \omega_2 \\ \omega_3 & 0 & -\omega_1 \\ -\omega_2 & \omega_1 & 0 \end{bmatrix} \tag{26}$$

The optimal control problem related to the system-wide translation sub-maneuver is now formulated as follows.

Problem 2. System-wide translation optimal control problem.

$$\min: J_1 = \int_0^{t_f} \boldsymbol{f}_0^{\mathrm{T}} \boldsymbol{W}_1 \boldsymbol{f}_0 dt \tag{17}$$

$$\text{s.t.}: \boldsymbol{f}_0 = m\ddot{\boldsymbol{r}}_c \tag{16}$$

$$\|\boldsymbol{f}_0\| \leqslant f_{0\,\max} \tag{8}$$

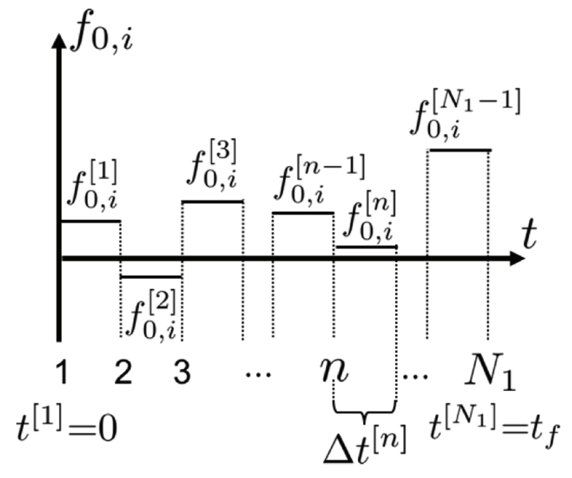


Fig. 6. The components of $\boldsymbol{f}_0$ are piecewise constant.

$$d \geqslant 0 \tag{23}$$

$$\|\boldsymbol{r}_g - \boldsymbol{r}_c\| = R_f \quad \text{for } t = t_f \tag{24}$$

$$\dot{\boldsymbol{r}}_c = \boldsymbol{\omega}_{\mathrm{RSO}}^{\times}(\boldsymbol{r}_c - \boldsymbol{r}_{\mathrm{RSO}}) \quad \text{for } t = t_f \tag{25}$$

For the problem to be feasible, the end-effector—not any other part of the chaser—must also define the chaser's keep-out sphere at capture $t = t_f$, as shown in Figure 3.

3.1. Convexification of Problem 2

Problem 2 is non-convex and, thus, needs to be convexified before it can be solved using convex programming techniques. To cast it as a convex programming problem, the final time needs to be fixed and the optimal control problem transcribed. Then, the non-convex keep-out zone constraint in Equation (23) needs to be convexified, and the nonlinear terminal (equality) constraint in Equation (24) relaxed to a convex inequality constraint.

Let us for the moment assume that the final time t_f is fixed and focus on the other remaining issues. The implications of converting the problem into a fixed-final-time one are discussed later.

An optimal control problem can be transcribed using a wide variety of methods (Conway, 2012; Hull, 1997). For simplicity, a direct transcription method with N_1 nodes is used here, keeping the forces $\boldsymbol{f}_0^{[n]}$ constant between the n and $n+1$ nodes and making the components of $\boldsymbol{f}_0$,

$$\boldsymbol{f}_0 = \begin{bmatrix} f_{0,1} \\ f_{0,2} \\ f_{0,3} \end{bmatrix} \tag{27}$$

piecewise constant, as illustrated in Figure 6.

With this assumption, the sub-maneuver's cost is obtained as follows:

$$J_1 = \sum_{n=1}^{N_1-1} \boldsymbol{f}_0^{[n]\mathrm{T}} \boldsymbol{W}_1 \boldsymbol{f}_0^{[n]} \Delta t^{[n]} \tag{28a}$$

$$\Delta t^{[n]} = t^{[n+1]} - t^{[n]} \tag{28b}$$

The dynamics of the system-wide translation can be expressed in a canonical, discrete-time, state-space representation as

$$\boldsymbol{x}^{[n+1]} = \boldsymbol{\Phi}_r^{[n]} \boldsymbol{x}^{[n]} + \boldsymbol{\Psi}_r^{[n]} \boldsymbol{f}_0^{[n]} \tag{29a}$$

$$\boldsymbol{x}^{[n]} = \begin{bmatrix} \boldsymbol{r}_c^{[n]} \\ \dot{\boldsymbol{r}}_c^{[n]} \end{bmatrix} \tag{29b}$$

with the state transition matrix $\boldsymbol{\Phi}_r^{[n]}$ and the control matrix $\boldsymbol{\Psi}_r^{[n]}$ defined as follows:

$$\boldsymbol{\Phi}_r^{[n]} = \begin{bmatrix} \boldsymbol{I}_3 & \Delta t^{[n]} \boldsymbol{I}_3 \\ \boldsymbol{0}_{3\times3} & \boldsymbol{I}_3 \end{bmatrix} \quad \boldsymbol{\Psi}_r^{[n]} = \frac{1}{m} \begin{bmatrix} \frac{\left(\Delta t^{[n]}\right)^2}{2} \boldsymbol{I}_3 \\ \Delta t^{[n]} \boldsymbol{I}_3 \end{bmatrix} \tag{30}$$

with $\boldsymbol{I}_3$ and $\boldsymbol{0}_{3\times3}$ denoting a 3 × 3 identity and zero matrix, respectively.

The non-convex terminal equality constraint on Equation (24) can be relaxed to the following convex inequality constraint:

$$\|\boldsymbol{r}_g(t_f) - \boldsymbol{r}_c(t_f)\| \leqslant R_f \tag{31}$$

Proposition 1. Problem 2 formulated, either with Equation (24) or with Equation (31), are equivalent if $\boldsymbol{r}_g$ lies at the boundary of S_{RSO}.

Proof. Any point meeting

$$\|\boldsymbol{r}_g(t_f) - \boldsymbol{r}_c(t_f)\| < R_f$$

is infeasible, as it violates the keep-out zone constraint defined in Equation (23). ☐

At this point, Problem 2 can be formulated as the following non-convex, fixed-final-time parameter optimization problem.

Problem 3. Non-convex fixed-final-time parameter optimization problem.

$$\min: \sum_{n=1}^{N_1-1} \boldsymbol{f}_0^{[n]\mathrm{T}} \boldsymbol{W}_1 \boldsymbol{f}_0^{[n]} \Delta t^{[n]} \tag{28a}$$

$$\text{s.t: } \boldsymbol{x}^{[n+1]} = \boldsymbol{\Phi}_r^{[n]} \boldsymbol{x}^{[n]} + \boldsymbol{\Psi}_r^{[n]} \boldsymbol{f}_0^{[n]} \quad n = 1 \dots N_1 - 1 \tag{29a}$$

$$\|\boldsymbol{f}_0^{[n]}\| \leqslant f_{0\,\max} \quad n = 1 \dots N_1 - 1 \tag{8}$$

$$d^{[n]} \geqslant 0 \quad n = 1 \dots N_1 \tag{23}$$

$$\|\boldsymbol{r}_g^{[n]} - \boldsymbol{r}_c^{[n]}\| \leqslant R_f \quad n = N_1 \tag{31}$$

$$\dot{\boldsymbol{r}}_c^{[n]} = \boldsymbol{\omega}_{\mathrm{RSO}}^{[n]\times}\left(\boldsymbol{r}_c^{[n]} - \boldsymbol{r}_{\mathrm{RSO}}^{[n]}\right) \quad n = N_1 \tag{25}$$

To overcome the non-convex constraint in Equation (23), a sequential convex programming procedure is used. In a sequential convex programming procedure a convex approximation of the original non-convex programming problem is repeatedly solved until the cost of two

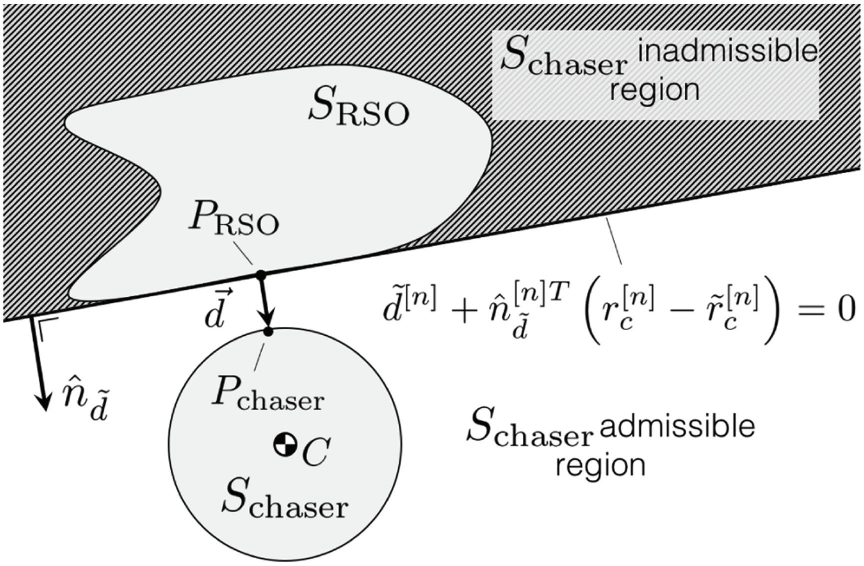


Fig. 7. Convexification of the keep-out zone constraint.

consecutive solutions, k and $k+1$, converges below a certain threshold ϵ_1.

$$\left\| {}^{[k-1]}J_1{}^{\star} - {}^{[k]}J_1{}^{\star} \right\|_1 \leqslant \epsilon_1 \tag{32}$$

The notation $|\cdot|^{\star}$ denotes the solution of a programming problem.

The convex approximation of the problem is obtained by linearizing the keep-out zone constraint in Equation (23) around the previous iteration solution, ${}^{[k-1]}\boldsymbol{f}_0^{\star}$. Using this previous solution, a reference trajectory $\tilde{\boldsymbol{r}}_c$ is obtained,

$${}^{[k-1]}\boldsymbol{f}_0^{\star} \rightarrow {}^{[k-1]}\boldsymbol{r}_c^{\star} = {}^{[k]}\tilde{\boldsymbol{r}}_c \tag{33}$$

which is used to obtain a reference signed distance $\tilde{d}$ and the vector projection $\tilde{\boldsymbol{d}}$ at every node n:

$$\tilde{\boldsymbol{r}}_c \rightarrow \tilde{d}^{[n]}, \tilde{\boldsymbol{d}}^{[n]} \tag{34}$$

A linear approximation of the signed distance d can then be obtained (Schulman et al., 2014):

$$d^{[n]} \approx \tilde{d}^{[n]} + \hat{\boldsymbol{n}}_{\tilde{\boldsymbol{d}}}^{[n]T}\left(\boldsymbol{r}_c^{[n]} - \tilde{\boldsymbol{r}}_c^{[n]}\right) \tag{35a}$$

$$\hat{\boldsymbol{n}}_{\tilde{\boldsymbol{d}}}^{[n]} = \frac{\tilde{\boldsymbol{d}}^{[n]}}{\left\|\tilde{\boldsymbol{d}}^{[n]}\right\|} \tag{35b}$$

Ultimately, the non-convex keep-out zone constraint in Equation (23) can be approximated by the following linear inequality constraint, as shown in Figure 7:

$$\tilde{d}^{[n]} + \hat{\boldsymbol{n}}_{\tilde{\boldsymbol{d}}}^{[n]T}\left(\boldsymbol{r}_c^{[n]} - \tilde{\boldsymbol{r}}_c^{[n]}\right) \geqslant 0 \tag{36}$$

The convex approximation of Problem 3 is written as follows.

Problem 4. Convex approximation of Problem 3

$$\min: \sum_{n=1}^{N_1-1} \boldsymbol{f}_0^{[n]\mathrm{T}} \boldsymbol{W}_1 \boldsymbol{f}_0^{[n]} \Delta t^{[n]} \tag{28a}$$

$$\text{s.t}: \boldsymbol{x}^{[n+1]} = \boldsymbol{\Phi}_r^{[n]} \boldsymbol{x}^{[n]} + \boldsymbol{\Psi}_r^{[n]} \boldsymbol{f}_0^{[n]} \qquad n = 1 \ldots N_1 - 1 \tag{29a}$$

$$\|\boldsymbol{f}_0^{[n]}\| \leqslant f_{0\,\max} \qquad n = 1 \ldots N_1 - 1 \tag{8}$$

$$\tilde{d}^{[n]} + \hat{\boldsymbol{n}}_{\tilde{\boldsymbol{d}}}^{[n]T}\left(\boldsymbol{r}_c^{[n]} - \tilde{\boldsymbol{r}}_c^{[n]}\right) \geqslant 0 \qquad n = 1 \ldots N_1 \tag{36}$$

$$\|\boldsymbol{r}_g^{[n]} - \boldsymbol{r}_c^{[n]}\| \leqslant R_f \qquad n = N_1 \tag{31}$$

$$\dot{\boldsymbol{r}}_c^{[n]} = \boldsymbol{\omega}_{\text{RSO}}^{[n]\times}\left(\boldsymbol{r}_c^{[n]} - \boldsymbol{r}_{\text{RSO}}^{[n]}\right) \qquad n = N_1 \tag{25}$$

The reference trajectory, $\tilde{\boldsymbol{r}}_c$, required to linearize the keep-out zone constraints is not available when attempting to solve Problem 4 for the first time ($k = 1$). To generate a solution to seed the sequential convex programming procedure, Problem 4 can be solved, replacing the keep-out zone constraint in Equation (36) with the following linear inequality constraint:

$$\left(\boldsymbol{r}_c^{[N_1]} - \boldsymbol{r}_g^{[N_1]}\right)^{\mathrm{T}} \hat{\boldsymbol{n}}_g^{[N_1]} \geqslant 0 \tag{37}$$

with $\hat{n}_g \in \left\{\mathbb{R}^3 | \|\hat{n}_g\| = 1\right\}$ denoting a unitary vector defining a plane that, according to Equation (37) and as illustrated in Figure 8, ensures that S_{chaser} remains outside S_{RSO} at the end of the maneuver, i.e., $n = N_1$.

Remark 3. The constraint in Equation (37) is equivalent to the keep-out zone constraint in Equation (36) at $n = N_1$. This constraint is added to force the chaser to grapple the RSO from the correct side.

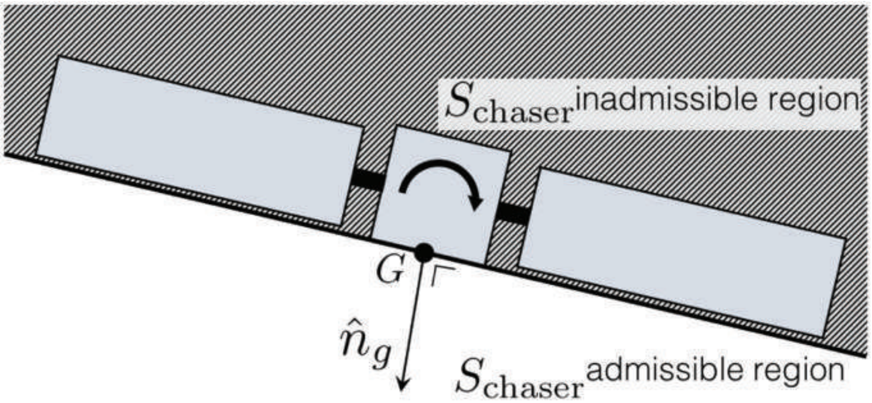
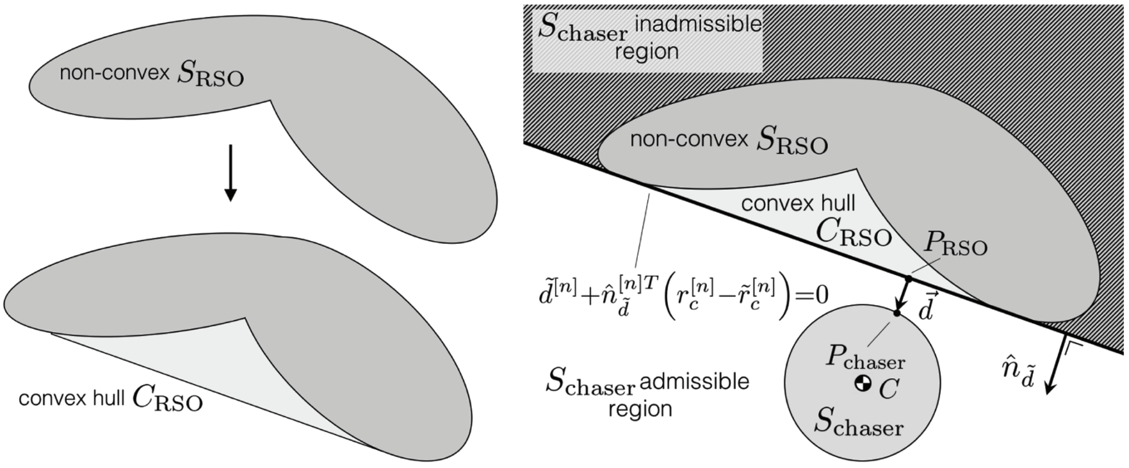


Fig. 8. Visualization of the constraint in Equation (37).



(a) (b)

Fig. 9. (a) Convex hull of a non-convex set and (b) linearized keep-out zone constraint.

3.2. Convergence properties with S_{RSO} approximated as a single convex hull

To begin exploring the convergence properties of the sequential convex programming procedure, let us assume that the target RSO is modeled using the convex hull of S_{RSO},

$$C_{\text{RSO}} = \text{Conv}(S_{\text{RSO}}) \tag{38}$$

Remark 4. If S_{RSO} is convex, then $C_{\text{RSO}} = S_{\text{RSO}}$.

The convex hull of an arbitrary non-convex set is notionally shown in Figure 9, along with a visualization of the linearized keep-out zone constraint of Equation (36).

If C_{RSO} is used, the sequential convex programming of Problem 4 can be shown to be a recursively feasible descent algorithm that converges to a locally optimal Karush–Kuhn–Tucker (KKT) point (Boyd and Vandenberghe, 2004) of the non-convex Problem 3. The proposition and its proof are included here for completeness. Variations of this proof have been reported by other authors (Lu and Liu, 2013; Morgan et al., 2016), and the proof can also be seen as a particular case of the more general convex–concave programming procedure (Lipp and Boyd, 2016; Yuille and Rangarajan, 2003).

Proposition 2. The sequential solutions to the convex Problem 4 converge to a KKT point of the non-convex Problem 3 if: S_{RSO} is approximated by its convex hull C_{RSO}, an initial admissible trajectory $\tilde{\boldsymbol{r}}_c$ is used to seed the sequential convex programming procedure, and $\boldsymbol{r}_g \in \partial C_{\text{RSO}}$.

Proof. Linearizing the non-convex keep-out zone constraints in Equation (23) with Equation (36) and using C_{RSO} to model the RSO keep-out zone, ensures that the entirety of S_{RSO} remains within the inadmissible region, as shown in Figure 9. This guarantees that the solution ${}^{[k]}\boldsymbol{f}_0^{\star}$ to the convex Problem 4 is an admissible point of the non-convex Problem 3.

In addition, a solution to the convex Problem 4 at iteration k, ${}^{[k]}\boldsymbol{f}_0^{\star}$, is an admissible point of the convex Problem 4 at iteration $k+1$, as the solution at iteration k is used to linearize the non-convex inequality constraint (Equation 23). It then follows that the cost of successive solutions ${}^{[k]}J_1^\star$ is non-strictly decreasing:

$$ {}^{[k]}J_1^\star \geqslant {}^{[k+1]}J_1^\star $$

As Problem 4 is lower bounded, the infinite sequence of solutions ${}^{[k]}\boldsymbol{f}_0^\star$ converges to ${}^{[\infty]}\boldsymbol{f}_0^\star$ with cost ${}^{[\infty]}J_1^\star \geqslant 0$:

$$ {}^{[\infty]}\boldsymbol{f}_0^\star = {}^{[\infty+1]}\boldsymbol{f}_0^\star $$

The converged solution ${}^{[\infty]}\boldsymbol{f}_0^\star$, being the solution to the convex Problem 4, automatically satisfies the stationarity, complementary slackness, primal feasibility, and dual feasibility conditions for Problem 4.

Then, as ${}^{[\infty]}\boldsymbol{f}_0^\star$ represents an admissible point of the non-convex Problem 3 it satisfies the primal feasibility condition of the non-convex Problem 3.

The only difference between Problems 3 and 4 is the non-convex keep-out zone constraint (23). This constraint is linearized (36) at $k=\infty$ using the converged solution to the convex Problem 4,

$$ {}^{[\infty]}\boldsymbol{f}_0^\star \rightarrow {}^{[\infty]}\boldsymbol{r}_c^\star = {}^{[\infty+1]}\tilde{\boldsymbol{r}}_c $$

thus,

$$ {}^{[\infty+1]}\boldsymbol{r}_c^\star = {}^{[\infty]}\boldsymbol{r}_c^\star = {}^{[\infty+1]}\tilde{\boldsymbol{r}}_c $$

Therefore, at $k \rightarrow \infty$ the following equalities hold:

$$ d^{[n]} = \tilde{d}^{[n]} = \left(\tilde{d}^{[n]} + \hat{\boldsymbol{n}}_{\tilde{\boldsymbol{d}}}^{[n]T}\left(\boldsymbol{r}_c^{[n]} - \tilde{\boldsymbol{r}}_c^{[n]}\right)\right) $$

$$ \nabla d^{[n]} = \nabla\left(\tilde{d}^{[n]} + \hat{\boldsymbol{n}}_{\tilde{\boldsymbol{d}}}^{[n]T}\left(\boldsymbol{r}_c^{[n]} - \tilde{\boldsymbol{r}}_c^{[n]}\right)\right) $$

implying that ${}^{[\infty]}\boldsymbol{f}_0^\star$ also meets the stationarity, complementary slackness and dual feasibility conditions of the non-convex Problem 3. ☐

Using this proof, two remarks on the problem's feasibility can be offered.

Remark 5. If the keep-out zone constraints are not enforced in Problem 4—as proposed to obtain an initial $\tilde{\boldsymbol{r}}_c$—and the problem is found to be infeasible, then the non-convex Problem 3—having a reduced set of admissible points—is infeasible.

Remark 6. It follows from Proposition 2, that if an $\tilde{\boldsymbol{r}}_c$ is found such that the convex Problem 4 is feasible (enforcing the keep-out zone constraints), then the non-convex Problem 3 is feasible.

3.3. Convergence properties with a non-convex S_{RSO}

The guaranteed convergence properties are extended here for a generic non-convex keep-out zone S_{RSO}, advancing the current state-of-the-art.

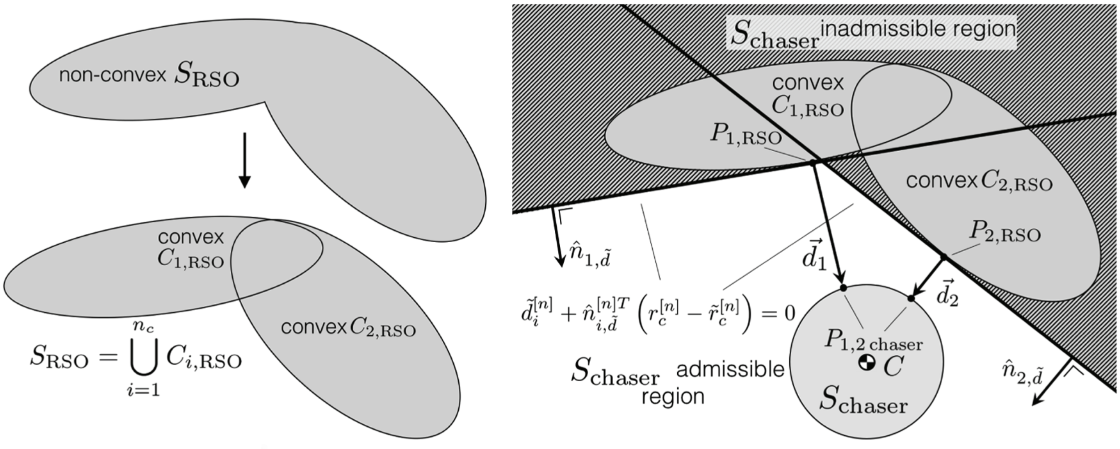
First, the non-convex S_{RSO} is decomposed, as shown in Figure 10, in n_c overlapping convex sets:

$$ S_{\mathrm{RSO}} = \bigcup_{i=1}^{n_c} C_{i,\mathrm{RSO}} \tag{39} $$

Then, the keep-out zone constraint in Equation (23) is applied to all n_c convex sets $C_{i,\mathrm{RSO}}$:

$$ d_i \geqslant 0 \quad \text{for } i = 1 \ldots n_c \tag{40} $$

It is clear that Problem 3 with the keep-out zone constraints formulated using S_{RSO} and Equation (23) or using $C_{i,\mathrm{RSO}}$ and Equation (40) are equivalent.



(a) (b)

Fig. 10. Non-convex keep-out zone S_{RSO} linearized by multiple convex keep-out zone constraints: (a) decomposition of a non-convex set; (b) linearized keep-out zone constraints.

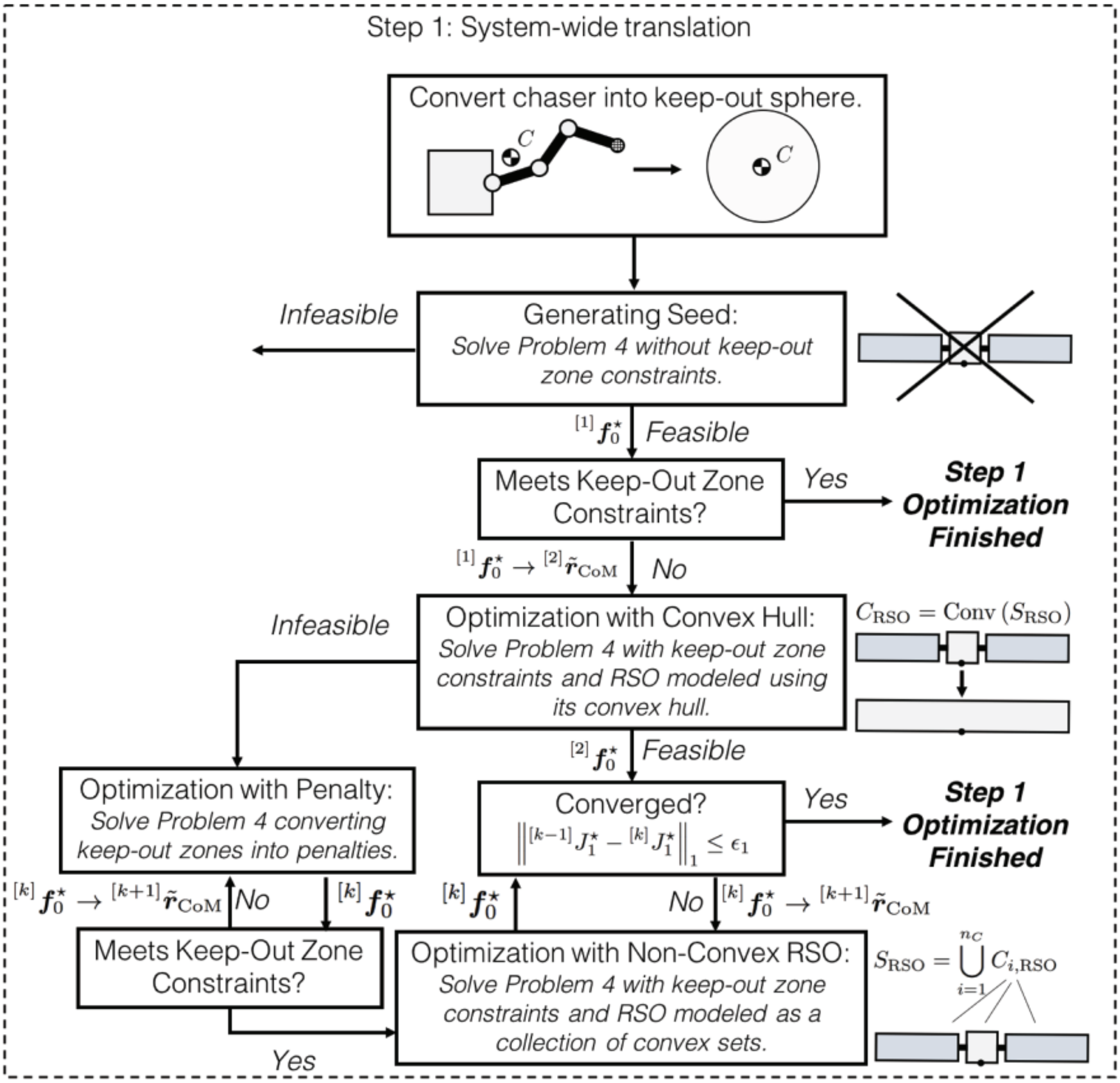


Fig. 11. Step 1: System-wide optimization procedure.

For the convex approximation (Problem 4) the constraint set in Equation (40) is linearized, resulting in

$$\tilde{d}_i^{[n]} + \hat{\boldsymbol{n}}_{i,\tilde{\boldsymbol{d}}}^{[n]T}\left(\boldsymbol{r}_c^{[n]} - \tilde{\boldsymbol{r}}_c^{[n]}\right) \geqslant 0 \quad \text{for } i = 1 \dots n_c \qquad (41)$$

Proposition 3. The sequential solutions to the convex Problem 4 converge to a KKT point of the non-convex Problem 3 if: S_{RSO} is modeled as a collection of n_c convex sets $C_{i,\mathrm{RSO}}$, the keep-out zone constraints are enforced using Equation (41), an initial admissible trajectory $\tilde{\boldsymbol{r}}_c$ is used to seed the sequential convex programming procedure, and $\boldsymbol{r}_g \in \partial S_{\mathrm{RSO}}$.

Proof. If a feasible trajectory $\tilde{\boldsymbol{r}}_c$ is used to start the sequential convex programming, Equation (41) ensures that the solution to Problem 4 is an admissible point of Problem 3, even when using a non-convex S_{RSO}, as shown in Figure 10.

Using an analogous procedure to that used during the proof of Proposition 2, it follows that the sequential solutions converge to ${}^{[\infty]}\boldsymbol{f}_0^\star$ and that at $k \to \infty$:

$$d_i^{[n]} = \tilde{d}_i^{[n]} = \left(\tilde{d}_i^{[n]} + \hat{\boldsymbol{n}}_{i,\hat{\boldsymbol{d}}}^{[n]T}\left(\boldsymbol{r}_c^{[n]} - \tilde{\boldsymbol{r}}_c^{[n]}\right)\right) \quad \text{for } i = 1 \dots n_c$$

$$\nabla d_i^{[n]} = \nabla\left(\tilde{d}_i^{[n]} + \hat{\boldsymbol{n}}_{i,\hat{\boldsymbol{d}}}^{[n]T}\left(\boldsymbol{r}_c^{[n]} - \tilde{\boldsymbol{r}}_c^{[n]}\right)\right) \quad \text{for } i = 1 \dots n_c$$

therefore, as in the proof of Proposition 2, it follows that ${}^{[\infty]}\boldsymbol{f}_0^\star$ is a KKT point of the non-convex Problem 3. ☐

Remark 7. All successive solutions to the convex Problem 4 are admissible points of the non-convex Problem 3. Therefore, the iterative process can stop at any time, obtaining an admissible control ${}^{[k]}\boldsymbol{f}_0^{\star}$ and trajectory ${}^{[k]}\boldsymbol{r}_c^{\star}$ of the original non-convex Problem 3.

3.4. Summary of Step 1

The Step 1 (**S1**) optimization procedure consists of the following steps, shown schematically in Figure 11.

- **S1**.1. Solve Problem 4 without enforcing the keep-out zone constraint in Equation (36) and adding the constraint in Equation (37). This initial solution is used as the initial reference trajectory $\tilde{\boldsymbol{r}}_c$ in the sequential convex programming procedure. If this convex problem is infeasible the maneuver is infeasible.
- **S1**.2. If $\tilde{\boldsymbol{r}}_c$ meets the keep-out zone constraints in Equation (23) using the non-convex S_{RSO}, the globally optimal solution to Problem 3 has been found. Skip the rest of these steps and move to the internal re-configuration sub-maneuver optimization. Otherwise, continue.
- **S1**.3. Approximate the RSO by its convex hull (C_{RSO}) and solve Problem 4 (using Equation (36) for the keep-out zone constraints). If this problem is feasible, it generates an admissible solution to Problem 3, seeding the sequential convex programming procedure used for the non-convex S_{RSO} (convergence is now guaranteed).

Remark 8. If Problem 4 is found to be feasible at the start of **S1**.3, then the non-convex Problem 3 is feasible. Solving two convex optimization problems is required to obtain a sufficient condition of feasibility.

If this problem is infeasible, then two options are available.

(a) Stop and consider the maneuver infeasible.

(b) Convert the keep-out zone constraint into a penalty as follows

$$J_1 = \sum_{n=1}^{N_1-1} \boldsymbol{f}_0^{[n]\mathrm{T}} \boldsymbol{W}_1 \boldsymbol{f}_0^{[n]} \Delta t^{[n]} + \sum_{n=1}^{N_1} \left| -\tilde{d}^{[n]} - \hat{\boldsymbol{n}}_{\tilde{\boldsymbol{d}}}^{[n]T} \left(\boldsymbol{r}_c^{[n]} - \tilde{\boldsymbol{r}}_c^{[n]} \right) \right|^{+} \tag{42a}$$

$$|x|^{+} = \begin{cases} x & \text{for } x \geqslant 0 \\ 0 & \text{for } x < 0 \end{cases} \tag{42b}$$

where $|\cdot|^{+}$ denotes the positive part (Schulman et al., 2014). Then repeatedly solve Problem 4 until a feasible solution is obtained (no guarantees are available). The use of penalties has been studied extensively, and when set correctly, they can be used to form equivalent problems, in what is known as exact penalties (Lipp and Boyd, 2016; Pillo and Grippo, 1989).

If the grapple fixture lies inside the convex hull, $\boldsymbol{r}_g \in C_{\mathrm{RSO}}$, and not in its boundary $\boldsymbol{r}_g \notin \partial C_{\mathrm{RSO}}$, the **S1**.3 problem is automatically infeasible. Refer to Section 3.5 for a procedure to circumvent this issue.

- **S1**.4. Repeatedly solve Problem 4, decomposing S_{RSO} with a collection of convex sets to enforce the keep-out zone constraints using Equation (41).
- **S1**.5. Stop at any time during **S1**.4 to obtain an admissible solution or continue with the sequential programming procedure until the desired convergence level is reached (see Equation (32)).

3.5. Solving Problem 3 when the grapple fixture is inside the convex hull

If the grapple fixture, $\boldsymbol{r}_g$, lies within—and not on the boundary of—the RSO's convex hull C_{RSO}, the terminal and obstacle avoidance constraints at $t = t_f$ of the convex problem to be solved in **S1**.3 will be incompatible. This incompatibility can be successfully circumvented by enforcing a waypoint, $\boldsymbol{x}_{\mathrm{w}}$ at $n = N_{\mathrm{w}}$ on the chaser's trajectory (i.e., $\boldsymbol{x}^{[N_{\mathrm{w}}]} = \boldsymbol{x}_{\mathrm{w}}$). Up to the waypoint, which must be compatible with the chaser's convex hull, the keep-out zones are enforced against the convex hull, yet after the waypoint, the keep-out zone constraints are dropped in favor of an obstacle-free polytope keep-in constraint, $\boldsymbol{A}\boldsymbol{x} \leqslant \boldsymbol{b}$. The solution to the resulting optimization problem is a collision-free trajectory that allows the sequential convex programming procedure to continue to **S1**.4 and eventually converge to a locally optimal solution. Figure 12 contains a notional representation of this case and the proposed workaround, which is similar to the method used by Watterson et al. (2016).

3.6. Effects of approximating the chaser as a sphere

Bounding the chaser's geometry by a keep-out sphere and using this sphere to enforce keep-out zone constraints is conservative and guarantees collision-free trajectories. However, this approximation may potentially reduce the solution's optimality and the problem's admissible set.

For example, any target protrusions that penetrate the chaser's bounding sphere in its final configuration produce incompatible terminal and obstacle avoidance constraints at $t = t_f$, thus rendering the problem infeasible. A potential method to overcome this limitation is to omit these protrusions from the obstacle avoidance considerations. Once the full maneuver optimization is complete, including the internal reconfiguration, collisions against the target's complete geometry can be checked. Given that the bounding sphere is conservative, it is possible that the generated trajectory is collision-free and usable.

3.7. Setting the initial and final time

To cast Problem 2 as a convex programming problem, it is converted into a fixed-final-time problem. The final time is considered a user-defined parameter. As suggested by Pinson and Lu (2016), a second optimization procedure

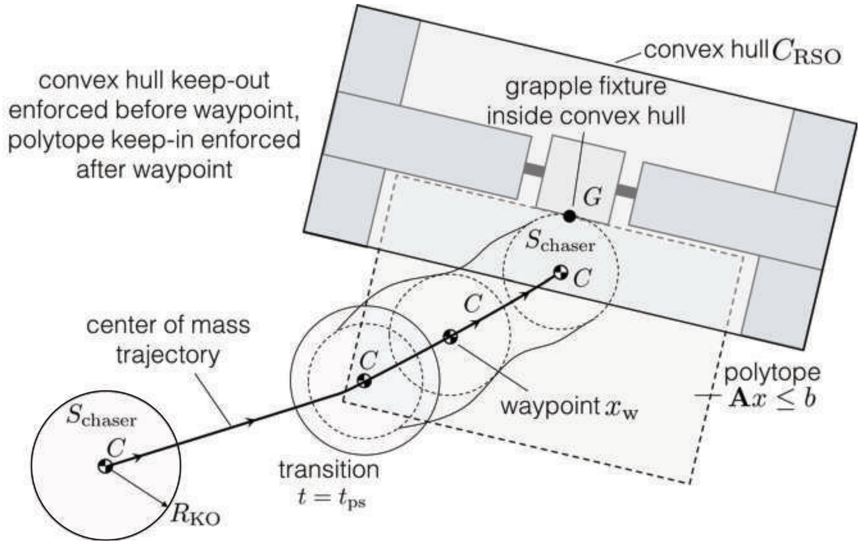


Fig. 12. Handling grapple fixtures inside the RSO's convex hull.

can be wrapped around the sequential convex programming procedure to determine a locally optimal final time $t_f^\star$.

A simpler approach to set the final time exploits the fact that, if the chaser starts at rest, the cost, as a function of time $J^\star(t_f)$, is largely monotonically decreasing (as shown in Section 5). This observation prompts the user to select the largest possible t_f, only limited by other considerations (e.g., illumination conditions or communication windows). One of these limiting factors may be the available computational power. Longer maneuvers require a higher number of nodes to maintain the same level of accuracy, increasing the computational complexity, and effectively upper-bounding t_f. Shorter maneuvers may also be preferred as their lower computational load allows for higher refresh rates, increasing the maneuver's robustness to uncertainties and perturbations.

Assuming that the final time is set to its maximum practical value, then the question becomes: when is the optimal time to start the maneuver? As the RSO is tumbling, the maneuver cost ${}^{[\infty]}J_1^\star(t_f)$ oscillates in a periodic fashion, as shown in Section 5. The oscillation is tied to the RSO's initial attitude, which changes as the maneuver starting time becomes delayed. The maneuver cost is certainly lower bounded and the bound can be estimated by offline simulations or by having the chaser continuously solving the optimization problem while the target RSO tumbles. Once the bound has been determined, the chaser, from its hold position, can continue to solve the optimal control problem, and when the cost of the computed maneuver is close to the estimated lower bound the maneuver is allowed to start.

3.8. Alternative approaches

It is worth acknowledging that alternative approaches to solve the system-wide translation exist. For example, inverse dynamic (Sternberg and Miller, 2018; Ventura et al., 2015; Virgili-Llop et al., 2018; Wilde et al., 2016), sampling-based (Starek et al., 2016; Zappulla et al., 2017), model predictive control (Park et al., 2017; Zagaris et al., 2018), or other convex optimization-based approaches (Szmuk and Acikmese, 2018; Watterson et al., 2016) may be able to solve the system-wide translation and be suitable for onboard implementation and real-time use. The advantage of the convex programming approach used here is that it can directly handle non-convex obstacles while retaining convergence guarantees. In addition, the proposed approach suitability for onboard implementation and real-time use has been validated through hardware-in-the-loop experiments on a planar air bearing test bed (see Section 6).

4. Step 2: internal re-configuration optimization

Once the system-wide translation has been optimized, also setting the final time t_f, it is the re-configuration sub-maneuver's turn. This second optimization step uses the results of the system-wide translation to couple the two sub-maneuvers.

Before formulating the optimal control problem for this sub-maneuver, let us derive the equations of motion of the system (1) on a CCS $\mathcal{C}$ attached to chaser's center of mass (see Figure 13). The axes of the $\mathcal{C}$ CCS are parallel to the inertial CCS $\mathcal{I}$ axes, but the origin of the $\mathcal{C}$ CCS is displaced by $\vec{r}_c$.

The equations of motion in the $\mathcal{C}$ CCS take the equivalent form of

$$\boldsymbol{H}|_{\mathcal{C}}\dot{\boldsymbol{u}}|_{\mathcal{C}} + \boldsymbol{C}|_{\mathcal{C}}\boldsymbol{u}|_{\mathcal{C}} = \boldsymbol{\tau}|_{\mathcal{C}} \tag{43}$$

As the orientation of the non-inertial $\mathcal{C}$ and inertial CCS $\mathcal{I}$ are the same, the inertia matrices are equivalent:

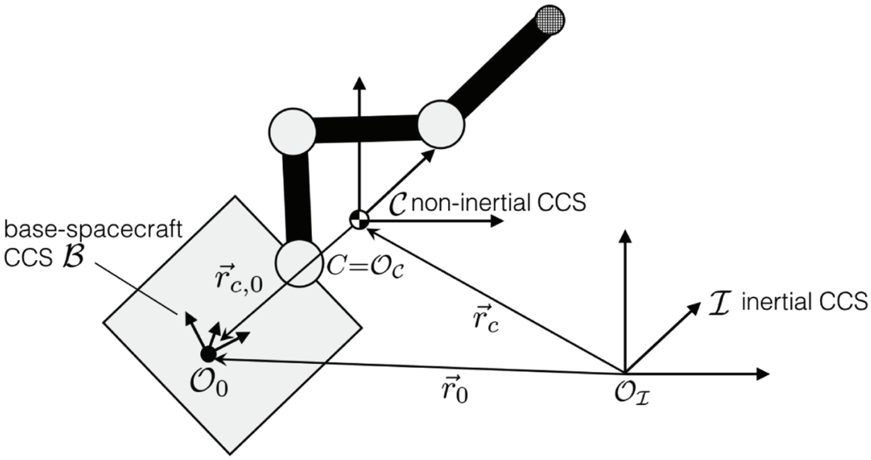


Fig. 13. CCS used for the internal re-configuration.

$$\boldsymbol{H}|_{\mathcal{C}} = \boldsymbol{H} \tag{44a}$$

$$\boldsymbol{C}|_{\mathcal{C}} = \boldsymbol{C} \tag{44b}$$

The manipulator velocities and accelerations, as well as the base-spacecraft angular velocities and accelerations are also equivalent:

$$\boldsymbol{u}_m|_{\mathcal{C}} = \boldsymbol{u}_m \tag{45a}$$

$$\dot{\boldsymbol{u}}_m|_{\mathcal{C}} = \dot{\boldsymbol{u}}_m \tag{45b}$$

$$\boldsymbol{\omega}_0|_{\mathcal{C}} = \boldsymbol{\omega}_0 \tag{45c}$$

$$\dot{\boldsymbol{\omega}}_0|_{\mathcal{C}} = \dot{\boldsymbol{\omega}}_0 \tag{45d}$$

The base-spacecraft linear velocities in the $\mathcal{C}$ CCS are with respect to the center of mass, resulting in the following definitions:

$$\dot{\boldsymbol{r}}_0|_{\mathcal{C}} = \dot{\boldsymbol{r}}_0 - \dot{\boldsymbol{r}}_c = \dot{\boldsymbol{r}}_{c,0} \tag{46a}$$

$$\ddot{\boldsymbol{r}}_0|_{\mathcal{C}} = \ddot{\boldsymbol{r}}_0 - \ddot{\boldsymbol{r}}_c = \ddot{\boldsymbol{r}}_{c,0} \tag{46b}$$

Finally, as we are using a non-inertial CCS, a set of generalized inertial forces $\boldsymbol{\tau}_{\mathcal{I}}$, derived from the frame's acceleration, appear:

$$\boldsymbol{\tau}|_{\mathcal{C}} = \boldsymbol{\tau} + \boldsymbol{\tau}_{\mathcal{I}} \tag{47}$$

These inertial forces ($\boldsymbol{\tau}_{\mathcal{I}}$) are the reaction to the forces applied to the center of mass $\boldsymbol{f}_c^{\star} = \boldsymbol{f}_0^{\star}$ mapped into equivalent generalized forces. The mapping is accomplished by exploiting the kineto-static duality (Siciliano et al., 2009) as follows:

$$\boldsymbol{\tau}_{\mathcal{I}} = -\boldsymbol{J}_c^{\mathrm{T}} \boldsymbol{f}_0^{\star} \tag{48}$$

with $\boldsymbol{J}_c^{\mathrm{T}}$ denoting the Jacobian of the center of mass, which can be obtained as

$$\boldsymbol{J}_c = \frac{\sum_{i=0}^{i=n_{\mathrm{DoF}}} \boldsymbol{J}_i m_i}{\sum_{i=0}^{i=n_{\mathrm{DoF}}} m_i} \tag{49}$$

and with m_i and $\boldsymbol{J}_i$ denoting the mass and geometric Jacobian of the ith link, respectively.

Although not apparent from Equation (48), it follows that

$$\boldsymbol{f}_{0,\mathcal{I}} = -\boldsymbol{f}_0^{\star} \tag{50}$$

as $\dot{\boldsymbol{r}}_0$ is used for both, the base-spacecraft linear velocity in joint and operational space, thus making the part of the Jacobian $\boldsymbol{J}_c$ that maps between these two velocities a 3 × 3 identity matrix.

The expression in Equation (50) allows the equations of motion in $\mathcal{C}$ CCS to re-written as

$$\boldsymbol{H}\begin{bmatrix} \ddot{\boldsymbol{r}}_{c,0} \\ \dot{\bar{\boldsymbol{u}}} \end{bmatrix} + \boldsymbol{C}\begin{bmatrix} \dot{\boldsymbol{r}}_{c,0} \\ \bar{\boldsymbol{u}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{0}_{3\times 1} \\ \bar{\boldsymbol{\tau}} - \bar{\boldsymbol{J}}_c^{\mathrm{T}} \boldsymbol{f}_0^{\star} \end{bmatrix} \tag{51a}$$

$$\dot{\bar{\boldsymbol{u}}} = \begin{bmatrix} \dot{\boldsymbol{\omega}}_0 \\ \dot{\boldsymbol{u}}_m \end{bmatrix} \quad \bar{\boldsymbol{u}} = \begin{bmatrix} \boldsymbol{\omega}_0 \\ \boldsymbol{u}_m \end{bmatrix} \quad \bar{\boldsymbol{\tau}} = \begin{bmatrix} \boldsymbol{n}_0 \\ \boldsymbol{\tau}_m \end{bmatrix} \tag{51b}$$

with $\bar{\boldsymbol{\tau}} \in \mathbb{R}^{3+n_{\mathrm{DoF}}}$ denoting the control variables and $\bar{\boldsymbol{J}}_c^{\mathrm{T}}$ the sub-Jacobian, corresponding to the internal re-configuration sub-maneuver control variable $\bar{\boldsymbol{\tau}}$.

Remark 9. The upper part of Equation (64) can be solved to obtain the base-spacecraft's reaction $\ddot{\boldsymbol{r}}_{c,0}$.

With the equations of motion defined, the optimal control problem can be formulated. The cost is defined by the following quadratic cost function,

$$J_2 = \int_0^{t_f} \bar{\boldsymbol{\tau}}^{\mathrm{T}} \boldsymbol{W}_2 \bar{\boldsymbol{\tau}} dt \tag{52}$$

with $\boldsymbol{W}_2 \in \mathbb{R}^{3+n_{\mathrm{DoF}} \times 3+n_{\mathrm{DoF}}}$ denoting a positive-definite weight matrix.

The constraints for this sub-maneuver are: limited manipulator joint deflections (Equation 6) and torques (Equation 7), base-spacecraft torque limits (Equation 9), line-of-sight constraint (Equation 11a), final pre-set

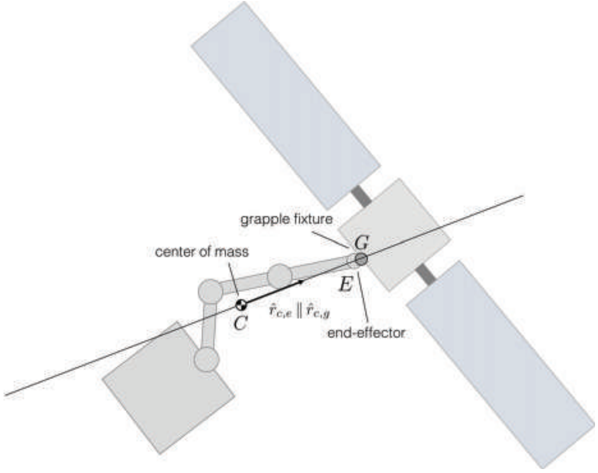


Fig. 14. Terminal attitude constraint.

manipulator motion (Equation 12), and the terminal constraints in Equations (13) and (15).

These constraints can be applied as-is, except the terminal constraint related to the matching of the end-effector and the grapple fixture, formally defined in Equation (13). During the first optimization step, the position of the center of mass is set so that the grapple fixture is within reach of the end-effector, yet the orientation of the chaser is let undefined. To ensure that the end-effector is at the grapple fixture at t_f, the base-spacecraft attitude must guarantee that the unit vector pointing from the center of mass towards the end-effector $\hat{\boldsymbol{r}}_{c,e}$ is aligned with the unit vector from the center of mass towards the grappling point $\hat{\boldsymbol{r}}_{c,g}$. This constraint, illustrated in Figure 14, is formulated as follows:

$$\hat{\boldsymbol{r}}_{c,e}^{\mathrm{T}}\hat{\boldsymbol{r}}_{c,g} = 1 \tag{53}$$

Note that the projection of $\hat{r}_{c,e}$ on the body CCS $\mathcal{B}$ ($\hat{\boldsymbol{r}}_{c,e}^{\{\mathcal{B}\}}$) is known, given the pre-set manipulator configuration at t_f. The projection of $\hat{r}_{c,g}$ on the inertial CCS $\mathcal{I}$ ($\hat{\boldsymbol{r}}_{c,g}^{\{\mathcal{I}\}}$) is also known, in this case from the solution to the system-wide translation sub-maneuver optimization.

Remark 10. The constraint in Equation (53) is equivalent to the line-of-sight constraint expressed in Equation (11a), yet with a half-cone angle of $\phi = 0$. The chaser is free to rotate around the axis defined by $\hat{r}_{c,g}$ while still meeting the constraint.

The resulting optimal control problem is presented in Problem 5. Given the nonlinear kinematics and dynamics this problem is non-convex.

Problem 5. Internal re-configuration optimal control problem.

$$\min: \int_0^{t_f} \bar{\boldsymbol{\tau}}^{\mathrm{T}}\boldsymbol{W}_2\bar{\boldsymbol{\tau}}dt \tag{52}$$

$$\text{s.t.:}\ \boldsymbol{H}\begin{bmatrix}\ddot{\boldsymbol{r}}_{c,0}\\ \dot{\boldsymbol{u}}\end{bmatrix} + \boldsymbol{C}\begin{bmatrix}\dot{\boldsymbol{r}}_{c,0}\\ \boldsymbol{u}\end{bmatrix} = \begin{bmatrix}\boldsymbol{0}\\ \bar{\boldsymbol{\tau}} - \bar{\boldsymbol{J}}_c^{\mathrm{T}}\boldsymbol{f}_0^{\star}\end{bmatrix} \tag{51a}$$

$$\dot{\boldsymbol{q}}_0 = \frac{1}{2}\boldsymbol{\omega}_0 \otimes \boldsymbol{q}_0 \tag{4}$$

$$\boldsymbol{\theta}_{m\,\min} \leqslant \boldsymbol{\theta} \leqslant \boldsymbol{\theta}_{m\,\max} \tag{6}$$

$$\boldsymbol{\tau}_{m\,\min} \leqslant \boldsymbol{\tau}_m \leqslant \boldsymbol{\tau}_{m\,\max} \tag{7}$$

$$\|\boldsymbol{n}_0\| \leqslant n_{0\,\max} \tag{9}$$

$$\boldsymbol{r}_d^{\mathrm{T}}\hat{\boldsymbol{v}} \geqslant \|\boldsymbol{r}_d\| \cos\phi \tag{11a}$$

$$\boldsymbol{\theta}_m(t) = \boldsymbol{\theta}_m^{\mathrm{ps}}(t) \quad \text{for } t \in \left[t_{\mathrm{ps}}, t_f\right] \tag{12}$$

$$\hat{\boldsymbol{r}}_{c,e}^{\mathrm{T}}\hat{\boldsymbol{r}}_{c,g} = 1 \tag{53}$$

$$\boldsymbol{\omega}_0\left(t_f\right) = \boldsymbol{\omega}_{\mathrm{RSO}}(t_f) \tag{15}$$

At this point, it is worth discussing why the manipulator motion is pre-set during the last part of the maneuver $t = \left[t_{\mathrm{ps}}, t_f\right]$ (see Equation (12)). If the manipulator motion during $t \geqslant t_{\mathrm{ps}}$ was subject to optimization, additional constraints would be required. For example, it may be desired to have a final manipulator configuration with a high kinematic manipulability—so that the manipulator can quickly correct any deviations. In addition, during this final phase, the end-effector must move outwards, $R_{\mathrm{KO}}^{\mathrm{ps}}(t) \geqslant R_f$, avoiding collisions with the RSO and ensuring that the first optimization step presents a feasible problem.

A pre-set manipulator motion can provide all the desired properties without increasing the complexity of the problem, and, as it only applies at the last segment of the maneuver, it has limited impact on the maneuver's cost. However, it is recommended to consider different pre-set motion approaches, conduct a case-by-case analysis, and select the one that offers better performance. There is a large amount of methods and strategies that can be used to generate suitable manipulator pre-set motions and a comprehensive study of their potential impact on the capture maneuver falls outside the scope of this work.

4.1. Explicitly convex line-of-sight constraint

In Equation (11a), the position of the sensor $\vec{r}_s$ can be expressed as

$$\vec{r}_s = \vec{r}_c + \vec{r}_{c,0} + \vec{r}_{0,s} \tag{54}$$

with $\vec{r}_c$ given by the solution of the first optimization step and with $\vec{r}_{0,s}$ being the known position vector of the sensor from the base-spacecraft.

The dependency of $\vec{r}_s$ on $\vec{r}_{c,0}$ makes this line-of-sight constraint a complex roto-translation constraint. Although, convex formulations of roto-translation line-of-sight constraints, requiring the use of dual quaternions, have been found (Lee and Mesbahi, 2016), a much simpler rotation-only line-of sight constraint can be obtained if $\vec{r}_{c,0}$ and $\vec{r}_{0,s}$ are neglected, approximating $\vec{r}_s$ as

$$\vec{r}_s \approx \vec{r}_c \tag{55}$$

Convex formulations of line-of-sight constraints have been found using semidefinite matrices or quaternion quadratic formulations (Kim et al., 2010; Lee and Mesbahi, 2016). The approach proposed by Kim et al. (2010) is used here to obtain an explicitly convex line-of-sight constraint.

4.1.1. General derivation of an explicitly convex line-of-sight constraint. In general, a line-of-sight constraint can be expressed as follows:

$$\vec{r} \cdot \vec{v} \geqslant \|\vec{v}\| \|\vec{r}\| \cos\phi \tag{56}$$

where $\vec{v}$ is the sensor's boresight direction and $\vec{r}$ is the desired pointing direction.

To compute the scalar product, these two vectors need to be resolved in a common reference coordinate system. In general, $\vec{v}$ is known in a body-fixed CCS $\boldsymbol{v}^{\{\mathcal{B}\}}$ and $\vec{r}$ is known in an inertial CCS $\boldsymbol{r}^{\{\mathcal{I}\}}$. With the attitude quaternion, the CCS where the $\vec{r}$ vector is resolved can be changed:

$$\boldsymbol{r}^{\{\mathcal{B}\}} = \boldsymbol{q}_0^* \otimes \boldsymbol{r}^{\{\mathcal{I}\}} \otimes \boldsymbol{q}_0 \tag{57}$$

with $\boldsymbol{q}_0^*$ denoting the quaternion conjugate of $\boldsymbol{q}_0$ and promoting $\boldsymbol{r}^{\{\mathcal{I}\}}$ to a pure quaternion.

By convention, the matricial representation of the quaternion is here defined as follows:

$$\boldsymbol{q} = \begin{bmatrix} \boldsymbol{q}_v \\ q_s \end{bmatrix} \tag{58}$$

where $\boldsymbol{q}_v$ denotes the "vector" part of the quaternion and q_s its "scalar" part.

The matricial form of the quaternion multiplication is then defined as

$$\boldsymbol{p} \otimes \boldsymbol{q} = [\boldsymbol{p}]_L \boldsymbol{q} = [\boldsymbol{q}]_R \boldsymbol{p} \tag{59a}$$

$$[\boldsymbol{q}]_L = q_s \boldsymbol{I}_4 + \begin{bmatrix} \boldsymbol{q}_v^\times & \boldsymbol{q}_v \\ -\boldsymbol{q}_v^\mathrm{T} & 0 \end{bmatrix} \tag{59b}$$

$$[\boldsymbol{q}]_R = q_s \boldsymbol{I}_4 + \begin{bmatrix} -\boldsymbol{q}_v^\times & \boldsymbol{q}_v \\ -\boldsymbol{q}_v^\mathrm{T} & 0 \end{bmatrix} \tag{59c}$$

When multiplying two pure quaternions $\boldsymbol{a}$ and $\boldsymbol{b}$ (i.e., with $q_s = 0$), the result provides the cross-product and the scalar dot product between these two "vectorial" quantities:

$$\boldsymbol{a} \otimes \boldsymbol{b} = \begin{bmatrix} \boldsymbol{a}^\times \boldsymbol{b} \\ -\boldsymbol{a}^\mathrm{T} \boldsymbol{b} \end{bmatrix} = \|\boldsymbol{a}\|\,\|\boldsymbol{b}\| \begin{bmatrix} \hat{\boldsymbol{n}} \sin\varphi \\ -\cos\varphi \end{bmatrix} \tag{60}$$

with φ denoting the angle between $\boldsymbol{a}$ and $\boldsymbol{b}$, and $\hat{\boldsymbol{n}}$ denoting the components of the unit vector normal to both $\boldsymbol{a}$ and $\boldsymbol{b}$.

Considering these properties, the line-of-sight constraint can be re-written as follows:

$$\boldsymbol{q}_0^* \otimes \boldsymbol{r}^{\{\mathcal{I}\}} \otimes \boldsymbol{q}_0 \otimes \boldsymbol{v}^{\{\mathcal{B}\}} = \|\boldsymbol{v}\|\,\|\boldsymbol{r}\| \begin{bmatrix} \hat{\boldsymbol{n}} \sin\varphi \\ -\cos\varphi \end{bmatrix} \tag{61}$$

These properties allow the line-of-sight constraint to be expressed as

$$\left[-[\boldsymbol{q}_0^*]_L [\boldsymbol{r}^{\{\mathcal{I}\}}]_L [\boldsymbol{v}^{\{\mathcal{B}\}}]_R \boldsymbol{q}_0 \right]^\mathrm{T} \begin{bmatrix} \boldsymbol{0}_{3\times1} \\ 1 \end{bmatrix} \geqslant \|\boldsymbol{v}\|\,\|\boldsymbol{r}\| \cos\phi \tag{62}$$

As

$$[\boldsymbol{q}^*]_L^\mathrm{T} \begin{bmatrix} \boldsymbol{0} \\ 1 \end{bmatrix} = \boldsymbol{q} \tag{63}$$

the line-of-sight constraint can be formulated using the following quadratic form:

$$-\boldsymbol{q}_0^\mathrm{T} \boldsymbol{A} \boldsymbol{q}_0 \geqslant \|\boldsymbol{v}\| \|\boldsymbol{r}\| \cos\phi \tag{64a}$$

$$\boldsymbol{A} = \left[\boldsymbol{v}^{\{\mathcal{B}\}}\right]_R \left[\boldsymbol{r}^{\{\mathcal{I}\}}\right]_L \tag{64b}$$

Note that $[\boldsymbol{q}]_R$ and $[\boldsymbol{q}]_L$ are skew-symmetric matrices and, thus, $\boldsymbol{A}$ is a symmetric indefinite matrix.

Skew-symmetric matrices have imaginary eigenvalues and, when using pure quaternions, $[\boldsymbol{a}]_R$ and $[\boldsymbol{a}]_L$ present two pairs of complex conjugate eigenvalues with $\lambda = \pm\|\mathbf{a}\|i$.

As $[\boldsymbol{q}]_R$ and $[\boldsymbol{q}]_L$ commute over multiplication, it is shown that the eigenvalues of $\boldsymbol{A}$ are the multiplication of the eigenvalues of the two skew-symmetric matrices. Therefore, the eigenvalues of $\boldsymbol{A}$ are $\lambda_A = \pm\|\boldsymbol{v}\|\|\boldsymbol{r}\|$.

As $\|\boldsymbol{q}_0\| = \boldsymbol{q}_0^\mathrm{T} \mathbf{q}_0 = 1$, the following identity is readily apparent:

$$\boldsymbol{q}_0^\mathrm{T}(\boldsymbol{A} - \mu \boldsymbol{I}_4)\boldsymbol{q}_0 = \boldsymbol{q}_0^\mathrm{T} \boldsymbol{A} \boldsymbol{q}_0 - \mu \boldsymbol{q}_0^\mathrm{T} \boldsymbol{q}_0 = \boldsymbol{q}_0^\mathrm{T} \boldsymbol{A} \boldsymbol{q}_0 - \mu \tag{65}$$

If μ is chosen to be smaller or equal than the minimum eigenvalue of $\boldsymbol{A}$, it follows that $\boldsymbol{A} - \mu \boldsymbol{I}_4$ is positive semidefinite,

$$\mu \geqslant \min(\lambda_A) = -\|\boldsymbol{v}\|\,\|\boldsymbol{r}\| \Rightarrow (\boldsymbol{A} - \mu \boldsymbol{I}_4) \succeq 0 \tag{66}$$

and the line-of-sight can be re-formulated as follows:

$$-\boldsymbol{q}_0^\mathrm{T}(\boldsymbol{A} - \mu \boldsymbol{I}_4)\boldsymbol{q}_0 - \mu \geqslant \|\boldsymbol{v}\|\,\|\boldsymbol{r}\| \cos\phi \tag{67}$$

Choosing

$$\mu = -\|\boldsymbol{v}\|\|\boldsymbol{r}\| \tag{68}$$

yields

$$-\boldsymbol{q}_0^\mathrm{T}(\boldsymbol{A} + \|\boldsymbol{v}\|\|\boldsymbol{r}\|\boldsymbol{I}_4)\boldsymbol{q}_0 + \|\boldsymbol{v}\|\|\boldsymbol{r}\| \geqslant \|\boldsymbol{v}\|\|\boldsymbol{r}\| \cos\phi \tag{69}$$

Finally, the line-of-sight constraint can be cast as an explicitly convex inequality constraint:

$$\boldsymbol{q}_0^\mathrm{T} \boldsymbol{A}^+ \boldsymbol{q}_0 + \|\boldsymbol{v}\|\|\boldsymbol{r}\|(\cos\phi - 1) \leqslant 0 \tag{70a}$$

$$\boldsymbol{A}^+ = (\boldsymbol{A} + \|\boldsymbol{v}\|\|\boldsymbol{r}\|\boldsymbol{I}_4) \tag{70b}$$

In an equivalent manner, an explicitly convex attitude keep-out zone constraint can also be formulated:

$$-\boldsymbol{q}_0^{\mathrm{T}}\boldsymbol{A}^{-}\boldsymbol{q}_0 - \|\boldsymbol{v}\|\|\boldsymbol{r}\|(1+\cos\phi) \leqslant 0 \tag{71a}$$

$$\boldsymbol{A}^{-} = (\boldsymbol{A} - \|\boldsymbol{v}\|\|\boldsymbol{r}\|\boldsymbol{I}_4) \tag{71b}$$

4.1.2. Explicitly convex line-of-sight constraint for Problem 5. Going back to the specific constraint of the problem at hand, we can express the line-of-sight constraint in Equation (11a) as follows:

$$\boldsymbol{q}_0^{\mathrm{T}}\boldsymbol{A}_{\mathrm{LoS}}^{+}\boldsymbol{q}_0 + \|\boldsymbol{r}_d\|(\cos\phi - 1) \leqslant 0 \tag{72a}$$

$$\boldsymbol{A}_{\mathrm{LoS}}^{+} = (\boldsymbol{A} + \|r_d\|\boldsymbol{I}_4) \tag{72b}$$

$$\boldsymbol{A}_{\mathrm{LoS}} = \left[\hat{\boldsymbol{v}}^{\{\mathcal{B}\}}\right]_R \left[\boldsymbol{r}_d^{\{\mathcal{I}\}}\right]_L \tag{72c}$$

Analogously, for the terminal constraint in Equation (53):

$$\boldsymbol{q}_0^{\mathrm{T}}\boldsymbol{A}_{\mathrm{Ter}}^{+}\boldsymbol{q}_0 \leqslant 0 \tag{73a}$$

$$\boldsymbol{A}_{\mathrm{Ter}}^{+} = \left(\boldsymbol{A} + \|\boldsymbol{r}_{c,e}\|\|\boldsymbol{r}_{c,g}\|\boldsymbol{I}_4\right) \tag{73b}$$

$$\boldsymbol{A}_{\mathrm{Ter}} = \left[\boldsymbol{r}_{c,e}^{\{\mathcal{B}\}}\right]_R \left[\boldsymbol{r}_{c,g}^{\{\mathcal{I}\}}\right]_L \tag{73c}$$

4.2. Convexification of kinematic and dynamic constraints

To overcome the nonlinear kinematics and dynamics a sequential convex programming procedure is used. A convex approximation of Problem 5 is obtained by linearizing the kinematics and dynamics around a reference trajectory, denoted by $\tilde{\bar{\boldsymbol{u}}}, \tilde{\boldsymbol{\theta}}_m$. The resulting convex programming problem is repeatedly solved, using the previous iteration's solution as the linearization trajectory. A trust region keeps the solution within a region where the linearization approximation is valid. Unlike the sequential convex procedure used during the first optimization step, this proposed procedure offers no guarantees of recursive feasibility or convergence.

A direct transcription method with N_2 nodes is used to transcribe Problem 5. The optimization variables are the generalized accelerations $\dot{\bar{\boldsymbol{u}}}^{[n]}$. Therefore, the components of the discretized accelerations ($\dot{\bar{\boldsymbol{u}}}^{[n]}$) are piecewise constant, the components of the velocities ($\bar{\boldsymbol{u}}^{[n]}$) piecewise linear, and the components of the generalized forces ($\bar{\boldsymbol{\tau}}^{[n]}$) piecewise nonlinear, as illustrated in Figure 15. A trapezoidal integration scheme is used to approximate the cost:

$$J_2 \approx \sum_{n=1}^{N_2-1} \frac{\bar{\boldsymbol{\tau}}^{[n+1]^{\mathrm{T}}}\boldsymbol{W}_2\bar{\boldsymbol{\tau}}^{[n+1]} - \bar{\boldsymbol{\tau}}^{[n]^{\mathrm{T}}}\boldsymbol{W}_2\bar{\boldsymbol{\tau}}^{[n]}}{2}\Delta t^{[n]} \tag{74}$$

The velocities $\bar{\boldsymbol{u}}$ can be propagated using state transition matrices:

$$\bar{\boldsymbol{u}}^{[n+1]} = \boldsymbol{\Phi}_{\boldsymbol{u}}\bar{\boldsymbol{u}}^{[n]} + \boldsymbol{\Theta}_{\boldsymbol{u}}^{[n]}\dot{\bar{\boldsymbol{u}}}^{[n]} \tag{75a}$$

$$\boldsymbol{\Phi}_{\boldsymbol{u}} = \boldsymbol{I}_{3+n_{\mathrm{DoF}}} \tag{75b}$$

$$\boldsymbol{\Theta}_{\boldsymbol{u}}^{[n]} = \Delta t^{[n]}\boldsymbol{I}_{3+n_{\mathrm{DoF}}} \tag{75c}$$

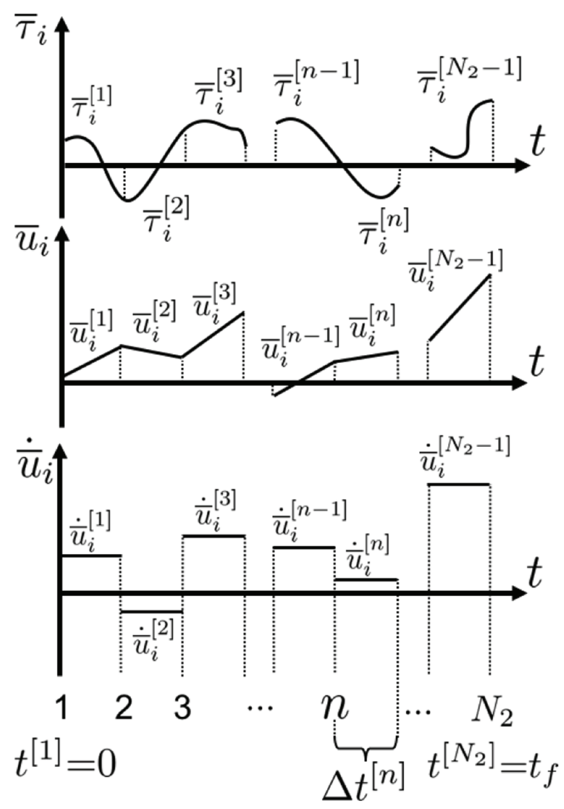


Fig. 15. Discretized internal re-configuration variables.

An analogous approach can be used to propagate the manipulator joint displacements:

$$\boldsymbol{\theta}_m^{[n+1]} = \boldsymbol{\Phi}_{\boldsymbol{\theta}}\boldsymbol{\theta}_m^{[n]} + \boldsymbol{\Theta}_{\boldsymbol{\theta}}^{[n]}\begin{bmatrix}\boldsymbol{\omega}_m^{[n]} \\ \dot{\boldsymbol{\omega}}_m^{[n]}\end{bmatrix} \tag{76a}$$

$$\boldsymbol{\Phi}_{\boldsymbol{\theta}} = \boldsymbol{I}_{n_{\mathrm{DoF}}} \tag{76b}$$

$$\boldsymbol{\Theta}_{\boldsymbol{\theta}}^{[n]} = \begin{bmatrix}\Delta t^{[n]}\boldsymbol{I}_{n_{\mathrm{DoF}}} & \frac{\left(\Delta t^{[n]}\right)^2}{2}\boldsymbol{I}_{n_{\mathrm{DoF}}}\end{bmatrix} \tag{76c}$$

The quaternion differential kinematics, shown in Equation (4), are nonlinear. As the angular velocity is piecewise linear, the following magnitudes can be defined

$$\boldsymbol{\omega}_0^{[n+1/2]} = \frac{\boldsymbol{\omega}_0^{[n+1]} + \boldsymbol{\omega}_0^{[n]}}{2} \tag{77a}$$

$$\hat{\boldsymbol{\omega}}_0^{[n+1/2]} = \frac{\boldsymbol{\omega}_0^{[n+1/2]}}{\left\|\boldsymbol{\omega}_0^{[n+1/2]}\right\|} \tag{77b}$$

$$\alpha_{\boldsymbol{\omega}_0}^{[n+1/2]} = \left\|\boldsymbol{\omega}_0^{[n+1/2]}\right\|\frac{\Delta t^{[n]}}{2} \tag{77c}$$

allowing to approximate the forward integration of the quaternion differential kinematics as follows:

$$\Delta\boldsymbol{q}^{[n]} \approx \begin{bmatrix}\hat{\boldsymbol{\omega}}_0^{[n+1/2]}\sin\alpha_{\boldsymbol{\omega}_0}^{[n+1/2]} \\ \cos\alpha_{\boldsymbol{\omega}_0}^{[n+1/2]}\end{bmatrix} \tag{78a}$$

$$\boldsymbol{q}_0^{[n+1]} \approx \Delta\boldsymbol{q}^{[n]} \otimes \boldsymbol{q}_0^{[n]} \tag{78b}$$

$$\boldsymbol{q}_0^{[n+1]} = f_{\boldsymbol{q}}^{[n]}\left(\boldsymbol{q}_0^{[n]}, \boldsymbol{\omega}_0^{[n+1/2]}\right) \tag{78c}$$

Despite the approximate nature of Equation (78), the quaternion propagation remains nonlinear.

Using a Taylor expansion around a particular trajectory $\tilde{\boldsymbol{\omega}}_0$, and truncating, only keeping the linear terms, a linear approximation of the function introduced in Equation (78c) is obtained:

$$f_{\boldsymbol{q}}^{[n]} \approx f_{\boldsymbol{q}}^{[n]}\left(\tilde{\boldsymbol{q}}_0^{[n]}, \tilde{\boldsymbol{\omega}}_0^{[n+1/2]}\right) + \\ + \nabla f_{\boldsymbol{q}}\left(\tilde{\boldsymbol{q}}_0^{[n]}, \tilde{\boldsymbol{\omega}}_0^{[n+1/2]}\right)\begin{bmatrix} \boldsymbol{q}_0^{[n]} - \tilde{\boldsymbol{q}}_0^{[n]} \\ \boldsymbol{\omega}_0^{[n+1/2]} - \tilde{\boldsymbol{\omega}}_0^{[n+1/2]} \end{bmatrix} \tag{79a}$$

$$\boldsymbol{q}_0^{[n+1]} \approx \tilde{\boldsymbol{q}}_0^{[n+1]} + \nabla f_{\boldsymbol{q}}\begin{bmatrix} \boldsymbol{q}_0^{[n]} - \tilde{\boldsymbol{q}}_0^{[n]} \\ \boldsymbol{\omega}_0^{[n+1/2]} - \tilde{\boldsymbol{\omega}}_0^{[n+1/2]} \end{bmatrix} \tag{79b}$$

An analogous procedure is used to approximate the nonlinear system dynamics. First, the dynamics are approximated using a Taylor expansion and then the expansion is truncated to keep only the linear terms:

$$\boldsymbol{H}\begin{bmatrix} \ddot{\boldsymbol{r}}_{c,0} \\ \dot{\bar{\boldsymbol{u}}} \end{bmatrix} + \boldsymbol{C}\begin{bmatrix} \dot{\boldsymbol{r}}_{c,0} \\ \bar{\boldsymbol{u}} \end{bmatrix} + \begin{bmatrix} 0 \\ \bar{\boldsymbol{J}}_c^{\mathrm{T}} \boldsymbol{f}_0^{\star} \end{bmatrix} = \begin{bmatrix} 0 \\ \bar{\boldsymbol{\tau}} \end{bmatrix} \tag{80a}$$

$$\bar{\boldsymbol{\tau}} = f_{\bar{\boldsymbol{\tau}}}\left(\boldsymbol{q}_0, \boldsymbol{\theta}, \bar{\boldsymbol{u}}, \dot{\bar{\boldsymbol{u}}}\right) = f_{\bar{\boldsymbol{\tau}}}\left(\dot{\bar{\boldsymbol{u}}}\right) \tag{80b}$$

$$\bar{\boldsymbol{\tau}} \approx \bar{\boldsymbol{\tau}} + \nabla f_{\bar{\boldsymbol{\tau}}}\left(\tilde{\dot{\bar{\boldsymbol{u}}}}\right)\left(\dot{\bar{\boldsymbol{u}}} - \tilde{\dot{\bar{\boldsymbol{u}}}}\right) \tag{80c}$$

Given the nonlinear nature of the inertia and convective inertia matrices, a complete linearization of the dynamics can be hard to obtain. The following partial linearization, omitting the linearization of the inertia matrices, is a simple to obtain—yet less accurate—alternative:

$$\bar{\boldsymbol{\tau}} \approx \tilde{\bar{\boldsymbol{\tau}}} + \tilde{\bar{\boldsymbol{H}}}\begin{bmatrix} \ddot{\boldsymbol{r}}_{c,0} - \tilde{\ddot{\boldsymbol{r}}}_{c,0} \\ \dot{\bar{\boldsymbol{u}}} - \tilde{\dot{\bar{\boldsymbol{u}}}} \end{bmatrix} + \tilde{\bar{\boldsymbol{C}}}\begin{bmatrix} \dot{\boldsymbol{r}}_{c,0} - \tilde{\dot{\boldsymbol{r}}}_{c,0} \\ \bar{\boldsymbol{u}} - \tilde{\bar{\boldsymbol{u}}} \end{bmatrix} \\ + \nabla\bar{\boldsymbol{J}}_c^{\mathrm{T}}\begin{bmatrix} \boldsymbol{q}_0 - \tilde{\boldsymbol{q}}_0 \\ \boldsymbol{\theta}_m - \tilde{\boldsymbol{\theta}}_m \end{bmatrix}\boldsymbol{f}_0^{\star} \tag{81a}$$

$$\dot{\boldsymbol{r}}_{c,0} \approx f\left(\tilde{\boldsymbol{H}}, \bar{\boldsymbol{u}}\right) \tag{81b}$$

$$\ddot{\boldsymbol{r}}_{c,0} \approx f\left(\tilde{\boldsymbol{H}}, \tilde{\boldsymbol{C}}, \dot{\bar{\boldsymbol{u}}}, \bar{\boldsymbol{u}}, \dot{\boldsymbol{r}}_{c,0}\right) \tag{81c}$$

with $\tilde{\boldsymbol{H}}$ and $\tilde{\boldsymbol{C}}$ denoting the inertia and convective inertia matrices constructed using the reference trajectory.

The linearization of the Jacobian can also be dropped to obtain an even simpler expression:

$$\bar{\boldsymbol{\tau}} \approx \tilde{\bar{\boldsymbol{\tau}}} + \tilde{\bar{\boldsymbol{H}}}\begin{bmatrix} \ddot{\boldsymbol{r}}_{c,0} - \tilde{\ddot{\boldsymbol{r}}}_{c,0} \\ \dot{\bar{\boldsymbol{u}}} - \tilde{\dot{\bar{\boldsymbol{u}}}} \end{bmatrix} + \tilde{\bar{\boldsymbol{C}}}\begin{bmatrix} \dot{\boldsymbol{r}}_{c,0} - \tilde{\dot{\boldsymbol{r}}}_{c,0} \\ \bar{\boldsymbol{u}} - \tilde{\bar{\boldsymbol{u}}} \end{bmatrix} \tag{82}$$

To ensure that the solution remains in a region where the linear approximations are valid, spherical trust regions around $\boldsymbol{\theta}_m$ and $\boldsymbol{\omega}_0$ are imposed:

$$\left\|\boldsymbol{\theta}_m^{[n]} - \tilde{\boldsymbol{\theta}}_m^{[n]}\right\|_1 \leqslant \rho_{\boldsymbol{\theta}_m} \tag{83a}$$

$$\left\|\boldsymbol{\omega}_0^{[n]} - \tilde{\boldsymbol{\omega}}_0^{[n]}\right\|_1 \leqslant \rho_{\boldsymbol{\omega}_0} \tag{83b}$$

with $\rho_{\boldsymbol{\theta}_m}$ and $\rho_{\boldsymbol{\omega}_0}$ denoting the radius of the trust region.

The resulting convex approximation of Problem 5 is shown in Problem 6.

Problem 6. Convex approximation of the internal reconfiguration optimal control problem.

$$\text{min}: \sum_{n=1}^{N_2-1} \frac{\bar{\boldsymbol{\tau}}^{[n+1]^{\mathrm{T}}} \boldsymbol{W}_2 \bar{\boldsymbol{\tau}}^{[n+1]} - \bar{\boldsymbol{\tau}}^{[n]^{\mathrm{T}}} \boldsymbol{W}_2 \bar{\boldsymbol{\tau}}^{[n]}}{2} \Delta t^{[n]} \tag{74}$$

$$\text{s.t.}: \bar{\boldsymbol{\tau}}^{[n]} = \tilde{\bar{\boldsymbol{\tau}}}^{[n]} + \nabla f_{\bar{\boldsymbol{\tau}}}\left(\tilde{\dot{\bar{\boldsymbol{u}}}}\right)\left(\dot{\bar{\boldsymbol{u}}} - \tilde{\dot{\bar{\boldsymbol{u}}}}\right) \qquad n = 1 \ldots N_2 \tag{80c}$$

$$\boldsymbol{q}_0^{[n+1]} = \tilde{\boldsymbol{q}}_0^{[n+1]} + \nabla f_{\boldsymbol{q}}\begin{bmatrix} \boldsymbol{q}_0^{[n]} - \tilde{\boldsymbol{q}}_0^{[n]} \\ \boldsymbol{\omega}_0^{[n+1/2]} - \tilde{\boldsymbol{\omega}}_0^{[n+1/2]} \end{bmatrix} \\ n = 1 \ldots N_2 - 1 \tag{79b}$$

$$\bar{\boldsymbol{u}}^{[n+1]} = \boldsymbol{\Phi}_{\boldsymbol{u}} \bar{\boldsymbol{u}}^{[n]} + \boldsymbol{\Theta}_{\boldsymbol{u}}^{[n]} \dot{\bar{\boldsymbol{u}}}^{[n]} \qquad n = 1 \ldots N_2 - 1 \tag{75a}$$

$$\boldsymbol{\theta}_m^{[n+1]} = \boldsymbol{\Phi}_{\boldsymbol{\theta}} \boldsymbol{\theta}_m^{[n]} + \boldsymbol{\Theta}_{\boldsymbol{\theta}}^{[n]}\begin{bmatrix} \boldsymbol{\omega}_m^{[n]} \\ \dot{\boldsymbol{\omega}}_m^{[n]} \end{bmatrix} \qquad n = 1 \ldots N_2 - 1 \tag{76a}$$

$$\boldsymbol{\theta}_{m\,\min} \leqslant \boldsymbol{\theta}_m^{[n]} \leqslant \boldsymbol{\theta}_{m\,\max} \qquad n = 1 \ldots N_2 \tag{6}$$

$$\boldsymbol{\tau}_{m\,\min} \leqslant \boldsymbol{\tau}_m^{[n]} \leqslant \boldsymbol{\tau}_{m\,\max} \qquad n = 1 \ldots N_2 \tag{7}$$

$$\|\boldsymbol{n}_0^{[n]}\| \leqslant n_{0\,\max} \qquad n = 1 \ldots N_2 \tag{9}$$

$$\boldsymbol{q}_0^{[n]T} \boldsymbol{A}_{\mathrm{LoS}}^{[n]+} \boldsymbol{q}_0^{[n]} + \|\boldsymbol{r}_d^{[n]}\|(\cos\phi - 1) \leqslant 0 \quad n = 1 \ldots N_2 \tag{72a}$$

$$\boldsymbol{\theta}_m^{[n]} = \boldsymbol{\theta}_m^{\mathrm{ps}[n]} \qquad n = n_{\mathrm{ps}} \ldots N_2 \tag{12}$$

$$\boldsymbol{q}_0^{[n]T} \boldsymbol{A}_{\mathrm{Ter}}^{[n]+} \boldsymbol{q}_0^{[n]} \leqslant 0 \qquad n = N_2 \tag{73a}$$

$$\boldsymbol{\omega}_0^{[n]} = \boldsymbol{\omega}_{\mathrm{RSO}}(t_f) \qquad n = N_2 \tag{15}$$

$$\left\|\boldsymbol{\theta}_m^{[n]} - \tilde{\boldsymbol{\theta}}_m^{[n]}\right\|_1 \leqslant \rho_{\boldsymbol{\theta}_m} \qquad n = 1 \ldots N_2 \tag{83a}$$

$$\left\|\boldsymbol{\omega}_0^{[n]} - \tilde{\boldsymbol{\omega}}_0^{[n]}\right\|_1 \leqslant \rho_{\boldsymbol{\omega}_0} \qquad n = 1 \ldots N_2 \tag{83b}$$

The solution to this convex problem is ${}^{[k]}\dot{\bar{\boldsymbol{u}}}^{\star}$, which is used to generate the new reference trajectory

$${}^{[k]}\dot{\bar{\boldsymbol{u}}}^{\star} \longrightarrow {}^{[k+1]}\tilde{\bar{\boldsymbol{u}}}, {}^{[k+1]}\tilde{\boldsymbol{\theta}}_m \tag{84}$$

The newly formed convex Problem 6 is repeatedly solved until the cost converges below a certain convergence threshold ϵ_2:

$$\left\|{}^{[k-1]}J_2^{\star} - {}^{[k]}J_2^{\star}\right\|_1 \leqslant \epsilon_2 \tag{85}$$

4.3. Initialization of the sequential convex programming procedure

The convex approximation of the problem relies on a linearization around a set trajectory $\tilde{\bar{\boldsymbol{u}}}, \tilde{\boldsymbol{\theta}}_m$. The first time the problem is solved there is no previous solution to rely on and an initial guess is required. To initialize the sequential convex programming procedure for system-wide translation, the non-convex constraints are dropped and the resulting convex problem solved, with its solution seeding the sequential procedure. Unfortunately, the non-convexities of the re-configuration sub-maneuver are in the kinematics and dynamics, which cannot be dropped. Another method to generate an initial guess is then required.

An initial guess for the base-spacecraft attitude can be generated by assuming that the chaser's attitude follows a trajectory that keeps the chaser's sensor perfectly pointing towards the grapple fixture, thus meeting the line-of-sight constraint. Starting from an initial orientation $\boldsymbol{q}_0^{[0]}$ that has the chaser's sensor pointed towards the grapple fixture, the rest of the attitude's initial guess can be constructed as follows:

$$\boldsymbol{q}_0^{[n+1]} = \Delta q^{[n]} \otimes q_0^{[n]} \tag{86a}$$

$$\hat{\boldsymbol{e}}^{[n]} = \frac{\left[\hat{\boldsymbol{n}}^{[n]}\right]^{\times} \hat{\boldsymbol{n}}^{[n+1]}}{\left\| \left[\hat{\boldsymbol{n}}^{[n]}\right]^{\times} \hat{\boldsymbol{n}}^{[n+1]} \right\|} \tag{86b}$$

$$\alpha^{[n]} = \arccos\left(\hat{\boldsymbol{n}}^{[n]T} \hat{\boldsymbol{n}}^{[n+1]}\right) \tag{86c}$$

$$\Delta \boldsymbol{q}^{[n]} = \begin{bmatrix} \hat{\boldsymbol{e}}^{[n]} \sin\left(\alpha^{[n]}/2\right) \\ \cos\left(\alpha^{[n]}/2\right) \end{bmatrix} \tag{86d}$$

Using this approach results in an attitude motion that may violate the terminal angular velocity constraint in Equation (15). This potential violation can be corrected during the first iteration of the sequential convex programming procedure, as long as the trust region on the angular velocity, $\rho_{\bar{u}}$, is large enough to allow a full correction (if $\rho_{\bar{u}}$ is not large enough, Equation (15) cannot be met and the convex programming problem becomes infeasible).

An initial guess for the manipulator's motion can be obtained assuming that the manipulator moves from its initial state to its final state following a linear trajectory in joint space (i.e., minimum deflection from initial pose to final pose) with two periods of constant acceleration:

$$\boldsymbol{\theta}_m(t) = \boldsymbol{\theta}_m(t_0) + \boldsymbol{\omega}_m(t_0) t + \dot{\boldsymbol{\omega}}_m \frac{t^2}{2} \tag{87a}$$

$$\begin{bmatrix} \dot{\omega}_{m,i} \text{ for } t = [t_0, \Delta t] \\ \dot{\omega}_{m,i} \text{ for } t = [\Delta t, t_{\mathrm{ps}}] \end{bmatrix} = \boldsymbol{A}^{-1} \boldsymbol{b}, \quad \text{for } i = 1 \ldots n_{\mathrm{DoF}} \tag{87b}$$

$$\boldsymbol{A} = \begin{bmatrix} \frac{3\Delta t^2}{2} & \frac{\Delta t^2}{2} \\ \Delta t & \Delta t \end{bmatrix} \tag{87c}$$

$$\boldsymbol{b} = \begin{bmatrix} \theta_{m,i}(t_{\mathrm{ps}}) - \theta_{m,i}(t_0) - \omega_{m,i}(t_0)\Delta t \\ \omega_{m,i}(t_{\mathrm{ps}}) - \omega_{m,i}(t_0) \end{bmatrix}, \quad \text{for } i = 1 \ldots n_{\mathrm{DoF}} \tag{87d}$$

$$\Delta t = \frac{t_{\mathrm{ps}} - t_0}{2} \tag{87e}$$

Although this procedure to obtain an initial guess is not the only possible one, it suffers from the same shortcomings as all other methods that do not explicitly enforce the problem constraints, namely: it may be infeasible (i.e., not meeting the control bounds).

4.4. Adjusting the trust regions

Adjusting the trust regions between iterations improves the robustness of the procedure while accelerating its convergence (Conn et al., 2000). As the kinematics and dynamics are nonlinear, the true cost of the trajectory ${}^{[k]}J^{\mathrm{true}}$ differs from the cost predicted by the optimization ${}^{[k]}J^{\star}$:

$$\left|{}^{[k]}J_2^{\mathrm{true}} - {}^{[k]}J_2^{\star}\right| \neq 0 \tag{88}$$

The difference between the two costs can be made arbitrarily small by tightening the trust region. A smaller trust region ensures that the linear approximation is more accurate, at the expense of a slower convergence rate.

To achieve a balance between robustness and speed of convergence, the trust region can be expanded or contracted as required. The following function is used to update the trust regions radius as a function of the cost difference:

$${}^{[k+1]}\rho = \kappa \frac{{}^{[k]}J_2^{\star}}{\left|{}^{[k]}J_2^{\mathrm{true}} - {}^{[k]}J_2^{\star}\right|} {}^{[k]}\rho \tag{89}$$

with κ denoting a tunable gain. The inflation/deflation rate is capped to avoid dramatic changes to the radius. A wide range of gains could be used or even other update laws conceived. As the gain or the update law may have a large influence on the speed of convergence a comprehensive analysis may be desired, yet such analysis falls outside the scope of this work.

Another issue with this sequential convex programming procedure is that the solution may violate the maximum torque, $\bar{\boldsymbol{\tau}}$, constraints (as $\bar{\boldsymbol{\tau}}$ is nonlinear). If that is the case, that particular solution can be disregarded, the trust region tightened and the convex problem re-solved. This approach is used by Mao et al. (2017), in conjunction with penalties and a trust region update law, to derive a sequential convex programming procedure with converge guarantees, yet potentially converging to an infeasible point.

Remark 11. The proposed sequential convex programming procedure to solve the internal re-configuration sub-maneuver does not guarantee convergence (no recursive feasibility guarantees) or that the solutions are admissible points of the original problem.

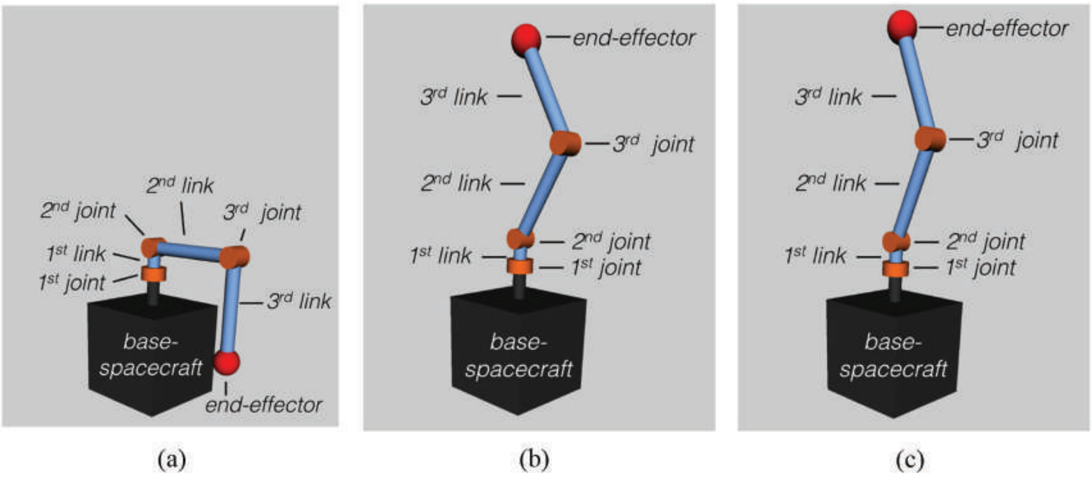


Fig. 16. Chaser model used in the numerical simulations: (a) stowed configuration; (b) configuration at $t = t_{\text{ps}}$; (c) capture configuration.

4.5. Overview of Step 2

The Step 2 (**S2**) optimization procedure consists of the following steps.

- **S2.**1 Generate an initial guess to serve as a reference trajectory.
- **S2.**2 Using the reference trajectory generate and solve the convex programming Problem 6.
- **S2.**3 With the solution obtained from solving Problem 6 then:
  - (a) check whether the solution meets the manipulator's torque constraints; if not, disregard this solution, tighten the trust regions, and repeat step **S2.**2;
  - (b) evaluate the solution's cost difference and adjust the trust region for the next iteration (see Section 4.4);
  - (c) proceed to the next iteration, repeat steps **S2.**2–**S2.**3.

Remark 12. If the Step 2 sequential convex programming procedure converges, the converged solution is a local optimal solution to the original non-convex problem. At the convergence point, the dynamics, linearized around the converged solution, are an exact representation.

A solution to the original guidance problem is obtained when the results of the two consecutive optimization steps **S2** and **S1** are combined.

5. Numerical simulations

To illustrate the proposed approach, a numerical simulation case study is provided. For added insight, a Monte Carlo analysis was conducted for 21 maneuver times (t_f) and 21 initial RSO angular velocity magnitudes ($\omega_{RSO}(t=0)\|$). For each pair of t_f and $\|\omega_{RSO}(t=0)\|$, 100 simulations with randomized RSO initial orientations (as proposed by Shoemake (1995)) and angular velocity directions was conducted. A total of $21 \times 21 \times 100 = 44,100$ capture maneuvers were simulated, allowing the robustness of the proposed guidance to be assessed.

The chaser and target RSO used in these simulations are shown in Figures 16 and 17. The chaser has a three-degree-of-freedom manipulator and a base-spacecraft mass fraction of 77% with respect to the total mass of the system. The target RSO convex hull, $\text{conv}(S_{\text{RSO}})$, is shown in Figure 17(b) and its convex decomposition into three convex sets ($n_C = 3$) in Figure 17(a). The parameters used in the simulations are provided in Table 1.

To solve the convex optimization problems CVX (Grant and Boyd, 2008, 2014) with the SDPT3 solver (Toh et al., 1999) was used. The 44,100 cases were solved on the Naval Postgraduate School high performance computing facility "Hamming."

In these simulations, the chaser starts the maneuver at rest, at a distance of 10 from its target, and with the manipulator in a stowed configuration (see Figure 16(a)). To initially meet the line-of-sight constraint, the chaser is oriented to have its sensor's boresight pointing towards the center of mass of the target. During the final part of the maneuver, $t = [t_{\text{ps}}, t_f]$, the manipulator's motion is set to move from $\boldsymbol{\theta}_m(t_{\text{ps}})$ to $\boldsymbol{\theta}_m(t_f)$ in a linear fashion and the keep-out radius is kept at a constant $R_{\text{KO}}^{\text{ps}}(t) = R_f$ for $t \in [t_{\text{ps}}, t_f]$.

5.1. Step 1 simulation results

During the first optimization step, 15 of the 12,100 capture maneuvers (0.03%) were found infeasible—the proposed guidance could not find a feasible solution—and 992 of them (2.25%) required the use of penalties to obtain an admissible solution (see **S1.**3b). These results suggest that the proposed approach is robust and able to handle a wide range of initial RSO states. For the converged cases, Figures 18–20 show the results of the first optimization step.

In particular, Figure 18 shows the mean system-wide translation sub-maneuver cost with respect to the different

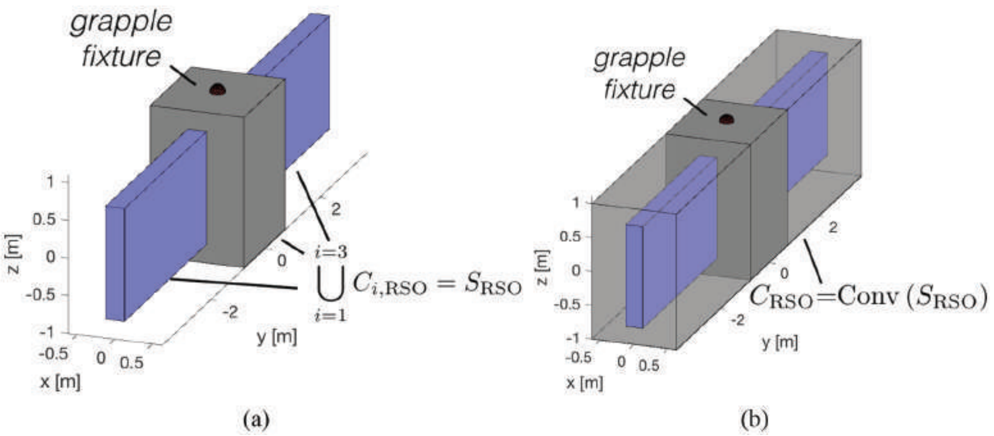


Fig. 17. RSO model used in the numerical simulations: (a) RSO's convex decomposition; (b) RSO's convex hull.

Table 1. Numerical simulation parameters.

Parameter	Value
Chaser	
Initial distance from RSO	10 m
Initial velocity	0 m s^{-1} & $0^{\circ}\mathrm{s}^{-1}$
Initial orientation	Sensor's boresight pointing towards target
Sensor's boresight (for line-of-sight)	Aligned with the first link
Mass of the base	105 kg
Inertia of the base	$J_{xx,yy,zz} = 9.3$ kg m^2
Mass of the links	$m_1 = 5$, $m_2 = 10$, and $m_3 = 10$ kg
Length of the links	$l_1 = 0.2$, $l_2 = 0.75$, and $l_3 = 0.75$ m
Initial configuration $\boldsymbol{\theta}_m(t_0)$	$\theta_1 = \pi$, $\theta_2 = \frac{\pi}{2}$, and $\theta_3 = \frac{\pi}{2}$ (see Figure 16(a))
Initial pre-set configuration $\boldsymbol{\theta}_m\left(t_{\mathrm{ps}}\right)$	$\theta_1 = -\pi$, $\theta_2 = \frac{\pi}{8}$, and $\theta_3 = -\frac{\pi}{4}$ (see Figure 16(b))
Grasping configuration $\boldsymbol{\theta}_m\left(t_f\right)$	$\theta_1 = -\pi$, $\theta_2 = \frac{\pi}{6}$, and $\theta_3 = -\frac{\pi}{3}$ (see Figure 16(c))
Grasping pose reach	$R_f = 1.85$ m
Keep-out sphere radius	$R_{\mathrm{KO}} = 2.04$ m
Maximum force	$f_{0\ \max} = 6.25$ N
Maximum base torque	$\tau_{0\ \max} = 1$ N m
Maximum joint torque	$\tau_{m\ \max} = 5$ N m
Maximum joint deflections	$\theta_1 = \pm\pi$, $\theta_2 = \pm\frac{\pi}{2}$, and $\theta_3 = \pm 1.75$
Line-of-sight cone half-angle	$\phi = 15^{\circ}$
RSO	
Initial attitude	Random
Initial angular velocity	Magnitude 0-10 $^{\circ}\mathrm{s}^{-1}$. Random initial direction
Mass	1200 kg
Inertia	$J_{xx} = 933$, $J_{yy} = 502$, and $J_{zz} = 693$ kg m^2
Optimization	
Number of nodes	$N_{1,2} = 101$
Pre-set time t_{ps}	$t_{\mathrm{ps}} = t_f - 20$ s
S1 convergence criteria	$\epsilon_1 = 0.01\%$ of $^{[k-1]}J_1^{\star}$
S2 convergence criteria	$\epsilon_2 = 2\%$ of $^{[k-1]}J_2^{\star}$
S2 initial trust regions	$\rho_{\boldsymbol{\theta}_m} = 0.5$, $\rho_{\boldsymbol{\omega}_0} \geqslant 0.5$
S2 trust regions tunable gain κ	0.001
S2 linearization of the dynamics	using Equation (81a)
Monte Carlo	
Maneuver time t_f	from 80 to 100 s in 1-second steps
RSO's initial angular velocity magnitude $\parallel \boldsymbol{\omega}_{\mathrm{RSO}}(t=0) \parallel$	from 0 to $10^{\circ}\mathrm{s}^{-1}$ in $0.5^{\circ}\mathrm{s}^{-1}$ steps
Number of samples per t_f, $\parallel \boldsymbol{\omega}_{\mathrm{RSO}}(t=0) \parallel$ combination	100

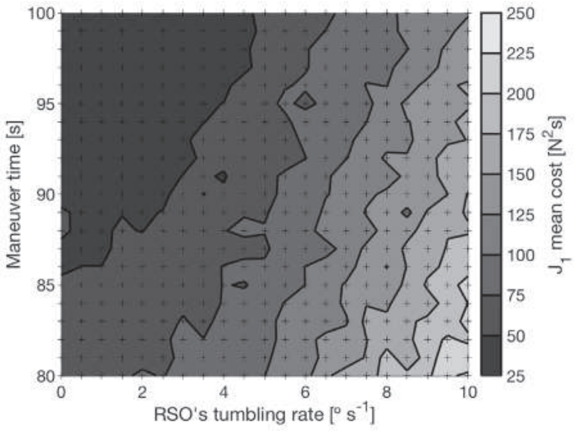


Fig. 18. Simulation results: system-wide translation mean cost.

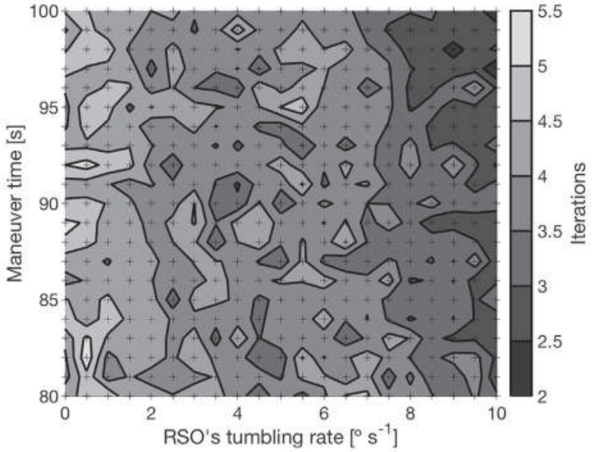


Fig. 19. Simulation results: system-wide translation mean iterations to converge.

maneuver times (t_f), and tumbling rate magnitudes ($\|\boldsymbol{\omega}_{RSO}(t=0)\|$). As expected, the cost decreases as the maneuver time t_f increases. With respect to the tumbling rate magnitude, the sub-maneuver cost increases as the tumbling rate increases.

Figure 19 shows the mean number of iterations required to converge, showing that, on average, fewer than five iterations are required. Figure 20 shows the convergence rate statistics for all cases. Note that the sequential convex programming procedure for the first optimization step is considered to converge when the cost between iterations decreases less than 0.01%.

For these simulations, the steps described in Section 3.4 and illustrated in Figure 11 are followed. During the first time that Problem 4 is solved the keep-out zone constraints are not enforced (see **S1.**1). Figure 20 shows that solving the optimization problem without enforcing the keep-out zone constraints is sufficient to solve nearly half the cases (optimum found). If the solution to the first iteration violates the keep-out zone constraints, more iterations are required. During the second time Problem 4 is solved, the S_{RSO} is approximated by its convex hull C_{RSO} (see **S1.**3). This second problem, if feasible, provides an admissible solution to the non-convex Problem 3, and from the third iteration onwards Problem 4 is solved using the non-convex S_{RSO} with guarantees of convergence (see **S1.**4). The solution to the second problem increases, by definition, the cost with respect to the initial solution as shown in Figure 20. If Problem 4 with the convex-hull is infeasible then, Problem 4 is re-solved converting the keep-out zones into penalties (see **S1.**3b), with the prospect to eventually find a feasible solution and resume the sequential convex programming procedure.

With this procedure, more than 92% of the cases converged after 8 iterations. The cases that after 8 iterations have not converged see little cost improvements in subsequent iterations (around 0.1%). These observations suggest that a good strategy would be to limit the number of iterations to 8, bounding the total computational time and ensuring convergence on most cases without incurring in excessive cost penalties for the non-converged cases.

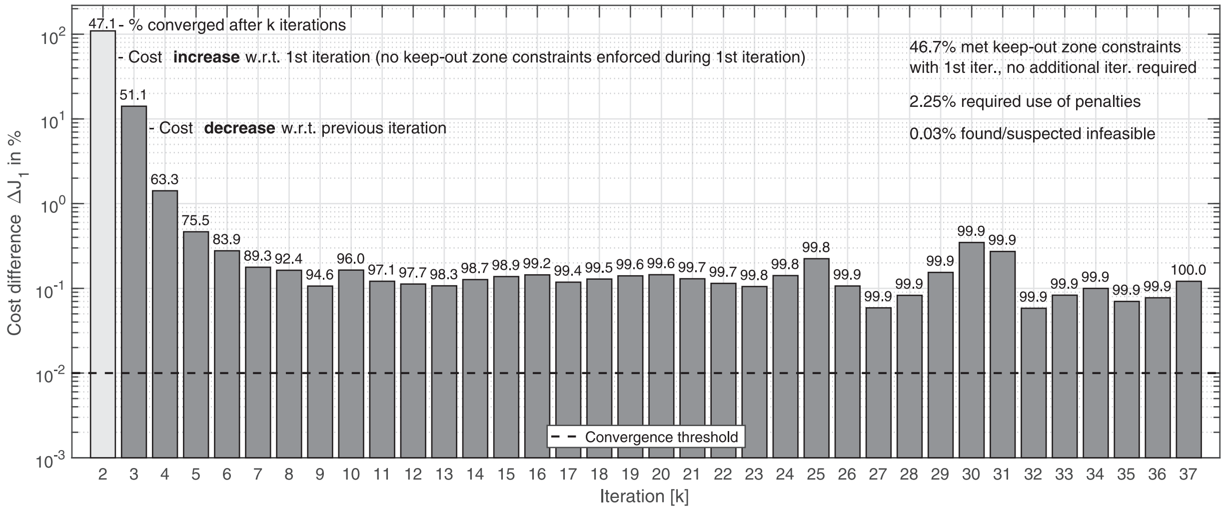


Fig. 20. Simulation results: System-wide translation convergence rate. Dashed line shows the 0.01% convergence threshold.

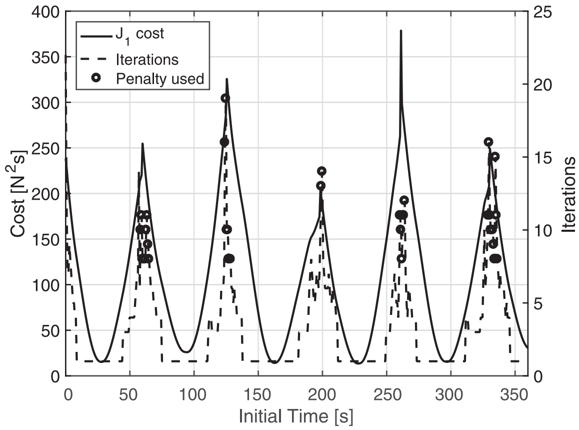


Fig. 21. Simulation results: system-wide translation cost and iterations with respect to initial time.

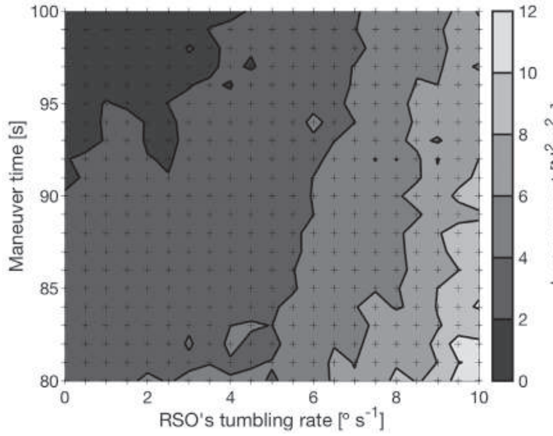


Fig. 22. Simulation results: internal re-configuration mean cost.

Finally, Figure 21 shows how the cost and required iterations evolve depending on the maneuver's start time. The RSO's orientation with respect to the chaser changes with time as the RSO is tumbling at a rate of $5^{\circ}\mathrm{s}^{-1}$. From Figure 21 it is clear that the RSO's orientation at the start of the maneuver—tied to the starting time—has a dramatic impact on the maneuver's cost and computational complexity. The cost and the number iterations, oscillating with a period $P \approx \frac{2\pi}{\|\omega_{\mathrm{RSO}}\|}$, present local minima when the geometry between the RSO and chaser is favorable. Waiting for such conditions to occur appears to be a good strategy to lower the maneuver's cost and computational load (see Section 3.7).

5.2. Step 2 simulation results

During the second optimization step admissible solutions where found for all of the internal re-configuration sub-maneuvers, suggesting that, despite the lack of convergence proofs, the proposed method is robust. Figures 22–24 show the results of the second optimization step.

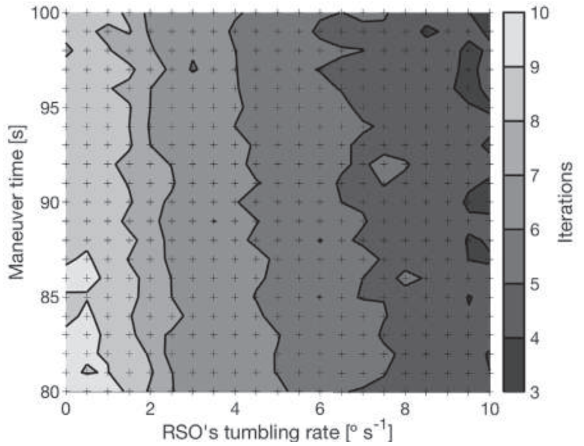


Fig. 23. Simulation results: internal re-configuration mean iterations to converge.

As expected, and as shown in Figure 22, the cost of the second sub-maneuver decreases as the maneuver time, t_f, increases and RSO's tumbling rate decreases. As shown in Figures 23–24, the second optimization step requires a larger number of iterations to converge. Note that in this case the convergence is achieved when the cost decrease is below 2%. Interestingly, the cost decrease plateaus at around 3% for $k \geqslant 9$ iterations. This behavior is induced by the trust region, which is tuned via κ (see Equation (89)). As the trust region stabilizes, the optimization convergence rate also stabilizes. A larger trust region allows a faster convergence rate but increases the risk of constraint violations or cost increases between consecutive iterations. As in the first optimization step, it may be a good strategy to limit the number of maximum iterations to bound the maximum computational time.

5.3. Combined maneuver

A video showing the different optimization steps and the final maneuver is shown in Extension 1.

6. Hardware-in-the-loop experiments

The simulation results illustrate the use and performance of the proposed guidance approach under a wide range of initial conditions. To demonstrate that the proposed guidance approach is suitable for onboard implementation and real-time execution, the algorithm has been implemented on an embedded onboard computer and used on the POSEIDYN hardware-in-the-loop dynamic test bed (Zappulla II et al., 2017b) to guide, in real-time, a robotic capture maneuver. As these experiments are performed on a planar environment, the guidance approach has been adapted to this environment and the line-of-sight constraint only enforced during the $t \in \left[t_{\mathrm{ps}}, t_f\right]$, yet with $\phi = 0$. More details on the experimental implementation can be found in Virgili-Llop et al. (2017c).

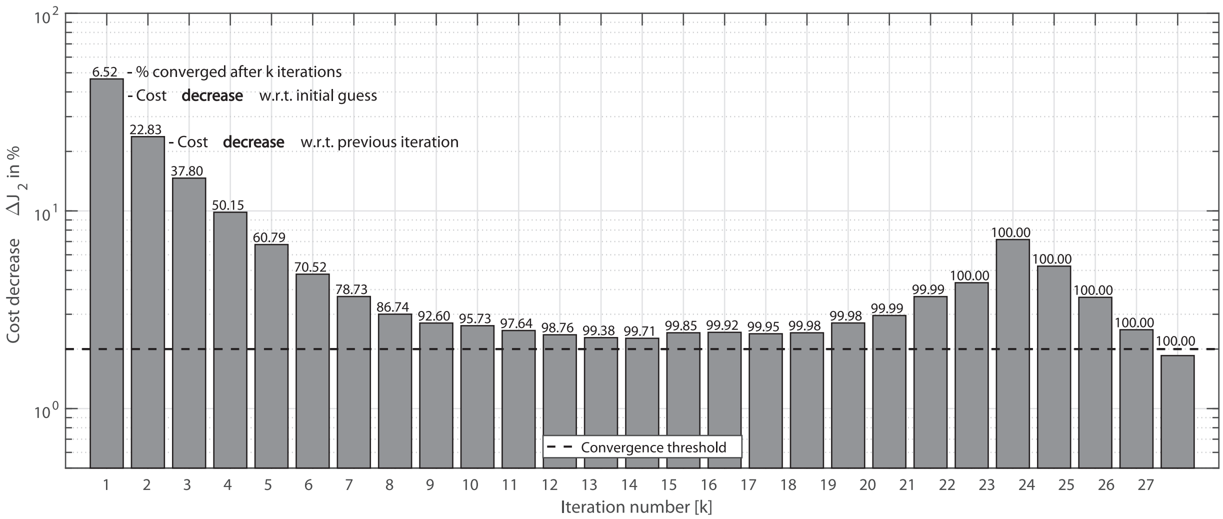


Fig. 24. Simulation results: internal re-configuration convergence rate. Dashed line shows the 2% convergence threshold.

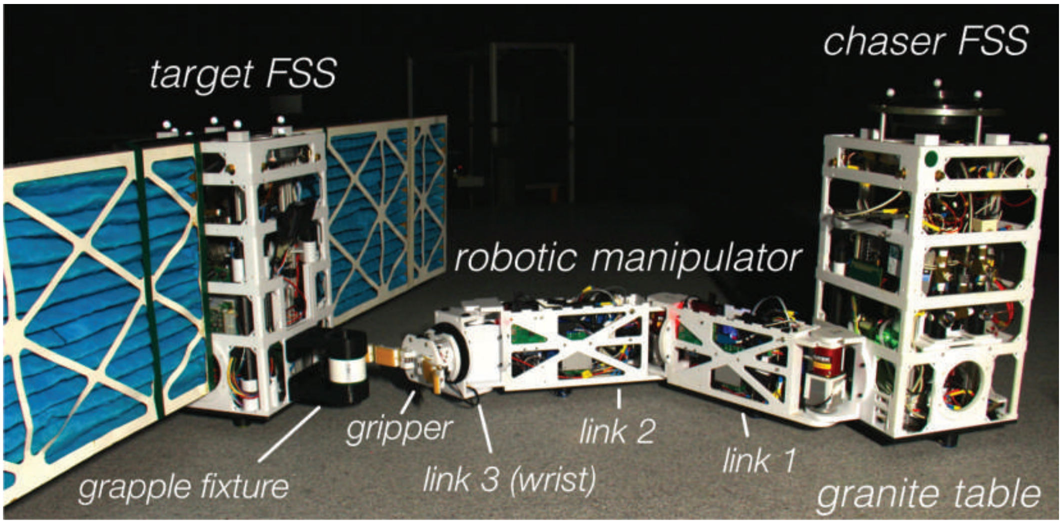


Fig. 25. Floating Spacecraft Simulators used during the experiments at the Spacecraft Robotics Laboratory POSEIDYN planar air bearing test bed at the Naval Postgraduate School.

6.1. Experimental set-up

The POSEIDYN air bearing test bed consists of a smooth and horizontally leveled 4×4 m^2 granite table and multiple Floating Spacecraft Simulators (FSS) (see Zappulla II et al., 2017b). A FSS, equipped with a three-link robotic manipulator, is used as the chaser spacecraft while a second FSS, with mock solar panels, simulates the tumbling object to be captured. An overview of the experimental set up is shown in Figure 25.

Three planar air bearings mounted on the FSS greatly reduce its friction with the granite table. This quasi-frictionless dynamics combined with the horizontally leveled table produce a low residual acceleration environment in two translation and one rotation degree of freedom (planar motion). Eight cold-gas thrusters (Lugini and Romano, 2009), modulated using a Delta-Sigma modulator (Ciarcià et al., 2017; Zappulla II et al., 2017a), provide the required control forces and torque. Selected parameters of the FSS are provided in Table 2.

A three-link manipulator is mounted on the chaser FSS (Virgili-Llop et al., 2016a, 2017a). The manipulator joints are all revolute. The first two links are identical and host an additional air bearing to support their weight. The third link of the manipulator is a minimalistic joint that functions like a “wrist,” allowing the gripper to adjust its orientation. The gripper is based on the open-source OpenHand Model T42 (Ma et al., 2013). Selected parameters of the robotic manipulator are provided in Table 3.

Table 2. Selected Floating Spacecraft Simulator parameters (Zappulla II et al., 2017b).

Parameter	Value
Mass	13 kg
Inertia	0.28 kg m^2
Dimensions (length × width)	0.27 × 0.27 m
Force per thruster	$\sim$0.15 N (inlet pressure dependent)
Air tank capacity	1.868 cm^3 (14 ci)
Air tank nominal pressure	20.7 Mpa (3000 psi)
Air bearings and thrusters inlet pressure	413.7 Pa (60 psi)
Onboard computer	Intel® Atom$^{\text{TM}}$ 1.6 GHz Z530 with 2 GB of RAM
Onboard computer performance	$\sim$1900 MIPS (Dhrystone v2.1)
Real-time operating System	Linux 2.6 with the `RT_PREEMPT` patch (Arthur et al., 2007)
Fiber-optic rate-gyroscope	KVH® DSP-3000
Test bed residual linear acceleration	$\sim$1.871 × 10^{-4} m s^{-2}(or $\sim$19.1 μg)
Test bed residual angular acceleration	$\sim$7.56 × 10^{-2}°s^{-2}

Table 3. Selected manipulator parameters.

Parameter	Value
Mass per modular link	2.9 kg
Inertia per modular link	≈ 0.0364 kg m^2
Third link and gripper mass	1.128 kg
Third link and gripper inertia	≈ 0.012 kg m^2
Modular link's length (axis-to-axis)	0.38 m
Link's width	0.08 m
Motor maximum torque	±1.8 N m (2.5 for the third joint)
Encoder resolution	150″ (317″ for the third joint)
Maximum joint angular displacement	± 90°
Gripper	OpenHand Model T42 (flexure–flexure) (Ma et al., 2013)
Control and telemetry rate	50 Hz

An overhead motion capture system (VICON) provides position and orientation measurements of the different FSS, at a 100 Hz rate. The chaser's base-spacecraft and RSO's position and orientation measurements are made available to the chaser FSS, effectively solving the relative navigation problem. The chaser's navigation data is augmented by an onboard fiber optic gyroscope (FOG).

6.2. Experimental concept

The nominal initial conditions of the maneuver are shown in Figure 26. These initial conditions present a "hard" guidance problem, requiring multiple iterations on both sequential convex programming procedures. The initial conditions were selected to obtain a trajectory that remains within the 4 × 4 m^2 granite table. The RSO rotation rate was set to 5°s^{-1} (above this rate, simulations show the chaser moving beyond the limits of the granite table). Note that the RSO resembles the one used for the numerical simulations (Section 5), with a similar convex hull (C_{RSO}), and also being decomposed with three convex sets.

The chaser first moves to its prescribed initial position while acquiring the initial manipulator's configuration. The capture maneuver starts when the RSO achieves a predetermined initial attitude (seen in Figure 26). Then, the two-step optimization problem is solved onboard the chaser FSS every 5 seconds. The solutions $\boldsymbol{f}_0^\star$ and $\ddot{\boldsymbol{u}}^\star$ are applied in a feed-forward arrangement with a linear-quadratic

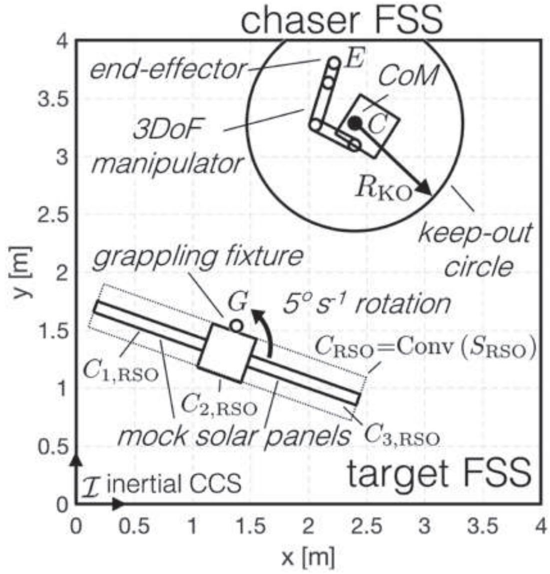


Fig. 26. Experiment initial conditions on the POSEIDYN test bed.

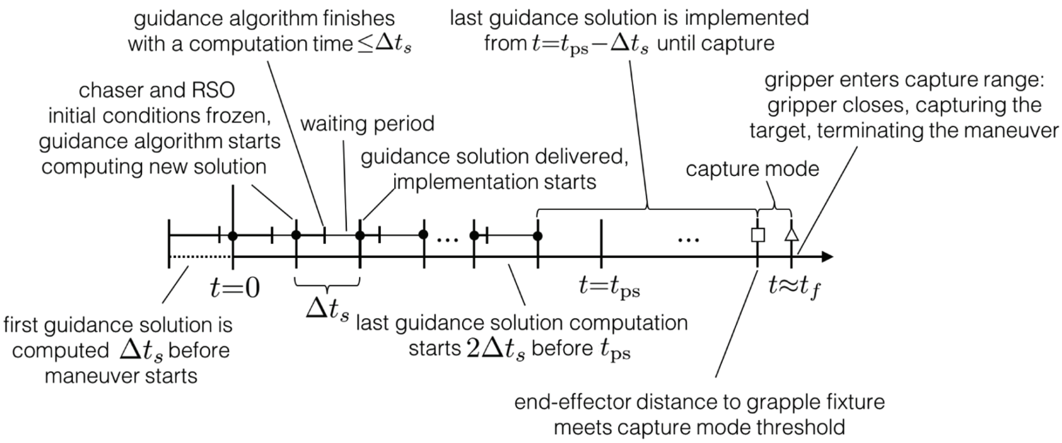


Fig. 27. Experimental timeline.

Table 4. Selected guidance parameters used during the experiments.

Parameter	Value
Guidance Algorithm	
Guidance re-compute rate Δt_s	5 s
Maneuver time t_f	70 s
Pre-set time t_{ps}	60 s
Number of nodes	$N_1 = 26$ and $N_2 = 20$
Convergence threshold	$\epsilon_{1,2}$ are set at 10% of $^{[k-1]}J_{1,2}$
S2 linearization of the dynamics	using Equation (118)
End-Effector error to start "capture mode"	$\leqslant 20$ cm
End-Effector error to trigger capture	$\leqslant 5$ cm
Chaser	
Chaser base-spacecraft initial position	$x = 2.5$ m, $y = 3.5$ m
Manipulator initial configuration	$\boldsymbol{\theta}_m(t=0) = [-80°,-80°,0°]$
Manipulator final configuration	$\boldsymbol{\theta}_m(t_f) = [-30°,60°,-30°]$
Manipulator configuration at t_{ps}	$\boldsymbol{\theta}_{ps}(t_{ps}) = [-36°,72°,-36°]$
RSO	
RSO position	$x = 1.3$ m, $y = 1.3$ m
RSO angular velocity	$5°\mathrm{s}^{-1}$
Nominal RSO orientation at start of maneuver	45°
Mock solar panel dimensions	1×0.1 m
Main body dimensions	0.4×0.4 m

regulator in a feedback loop, correcting any deviations from the nominal trajectory (e.g., resulting from actuator noise, errors in kinematic/dynamic models, or other unmodeled effects). The implemented linear-quadratic regulator is equivalent to that implemented in Virgili-Llop et al. (2017a). The open-source nonlinear programming solver IPOPT (Wächter and Biegler, 2005) is used to solve the convex programming problems onboard the chaser FSS and the open-source SPART toolkit (Virgili-Llop, 2017) is used to obtain all the manipulator-related kinematic and dynamic quantities.

To generate the initial guess of the second step optimization, a linear manipulator motion from initial to the final configuration is used, while the chaser's attitude is set to face the RSO's grappling fixture. Selected guidance parameters are provided in Table 4.

When the chaser reaches $t > t_{\mathrm{ps}}$ no new guidance updates are produced and the chaser follows the latest available solution. Finally, when the manipulator's end-effector is within a certain distance with respect to the grappling fixture, the chaser transitions to the capture mode. In this mode, the chaser uses a linear-quadratic regulator to acquire and maintain the terminal position of the last guidance solution. The third joint of the manipulator is steered to point the end-effector towards the grappling fixture. The end-effector distance with respect to the grappling fixture

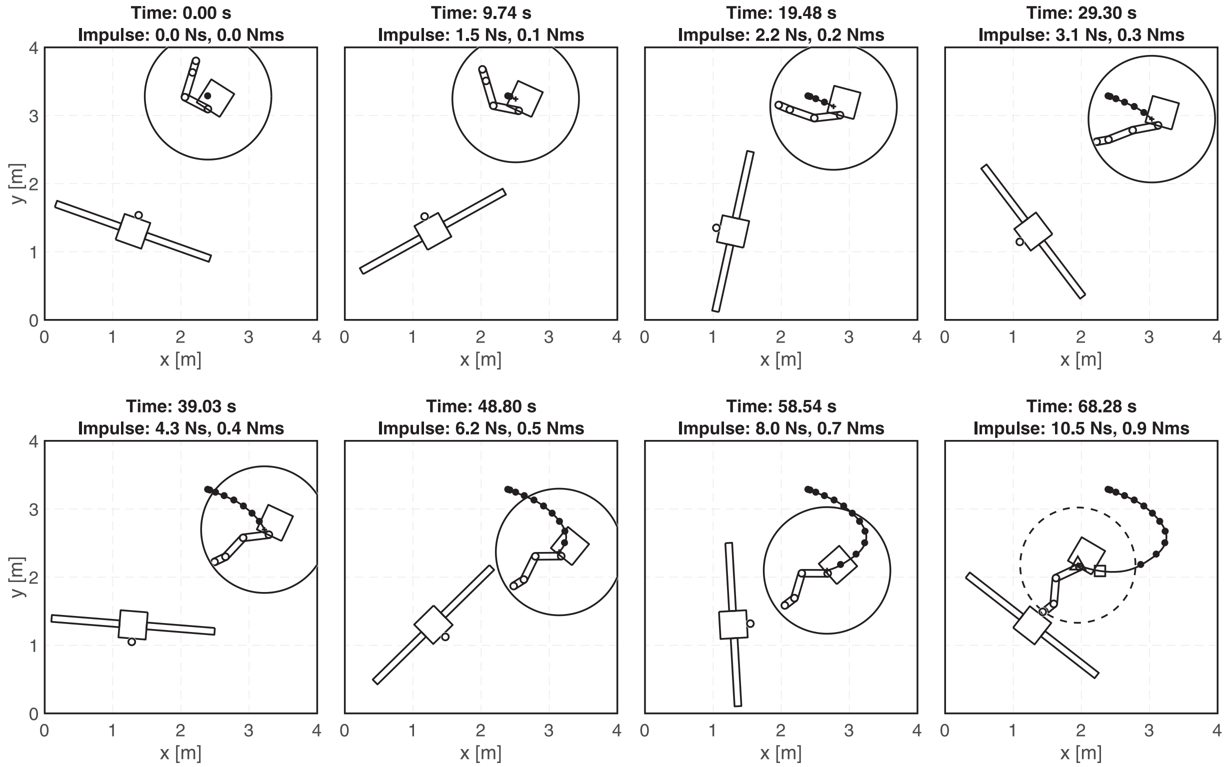
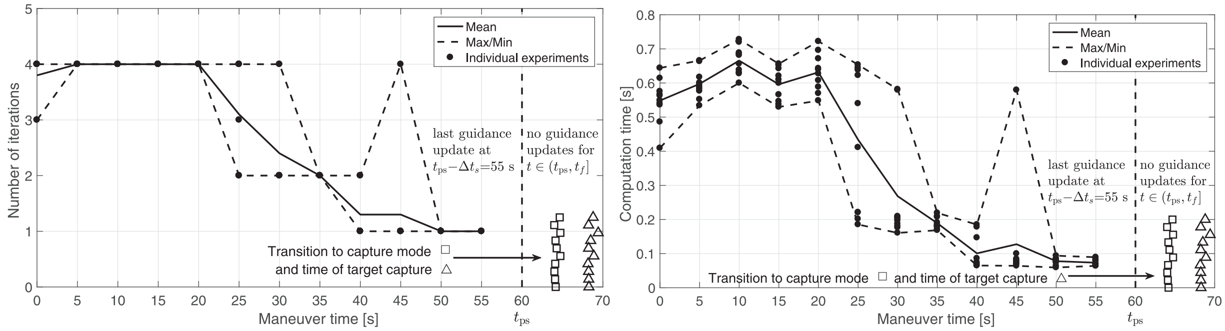


Fig. 28. Experimental results: telemetry snapshots of the first experiment.



(a) (b)

Fig. 29. Experimental results: Step 1 computational effort. (a) Iterations. (b) Computation time.

continues to decrease until a pre-determined threshold is reached, triggering the gripper to close, capturing the rotating RSO. The experimental timeline is notionally illustrated in Figure 27.

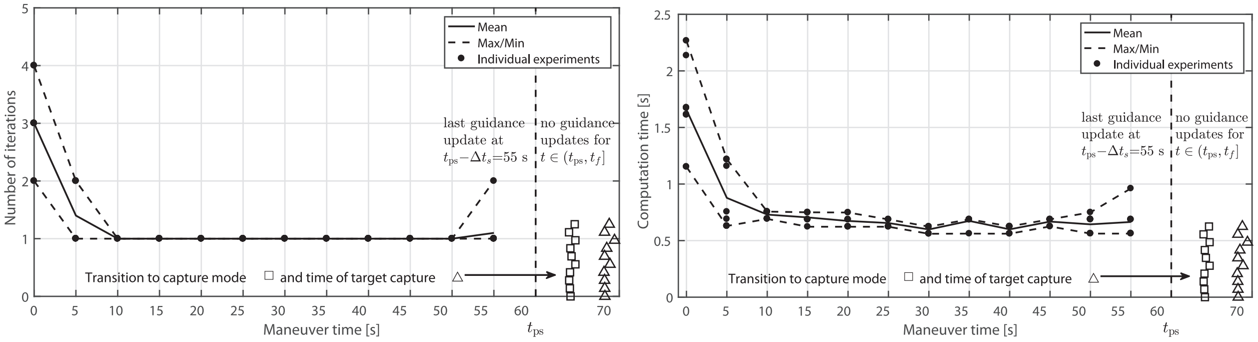
The FSS emulating the RSO to be captured is also controlled up to the capture instant. Its position and orientation are controlled to follow the prescribed rotation rate. When the RSO is captured this control stops, allowing the combined RSO-chaser system to drift freely.

6.3. Experimental results

Ten experiments were performed sequentially in a single session and all were successful, showing that the proposed guidance is able to produce repeatable results. A video showing an example maneuver and the telemetry replays for all 10 experiments is shown in Extension 2.

Despite the prescribed initial conditions, the RSO's attitude at the beginning of the maneuver is variable due to the guidance solution being computed every 5 seconds in a synchronous manner. The capture maneuver only starts when the RSO reaches its initial orientation, but the computation of the trajectory starts when the chaser's internal clock reaches the next 5-second mark. Therefore, a variability on the initial RSO's attitude of 25° is introduced.

Eight snapshots of the first experiment are shown in Figure 28. The bold line denotes the trajectory of the chaser's center of mass, the bold circles, ●, denote the locations



(a) (b)

Fig. 30. Experimental results: Step 2 computational effort. (a) Iterations. (b) Computation time.

Table 5. Experimental and simulation results.

Parameter	Simulation	Experiments mean	Experiments standard deviation
Maneuver time	66.85 s	68.50 s	0.41 s
Linear impulse	6.8 N s	11.09 N s	0.48 N s
Angular impulse	0.6 N m s	0.90 N m s	0.06 N m s

where the guidance updates are delivered, the square, □, denotes the point where the chaser enters into the capture mode, and the triangle, △, denotes when the gripper closes and the capture is achieved (corresponding to the last figure of the series).

Figures 29 and 30 show the number of iterations required to converge and the total amount of computational time used.

What can be observed from these figures is that the optimization problem is harder at the beginning, taking several iterations to converge. As the maneuver advances, the chaser, given its position and velocity, becomes “committed” to that particular trajectory. Making changes to the trajectory of a moving chaser gradually becomes more expensive, making the guidance solution settle on the already planned trajectory. It is also very relevant to note that the cumulative computation time is within the allotted 5 s interval.

Figure 31 shows the evolution of the manipulator’s configuration, the commanded base-spacecraft forces and torque, as well as the position and orientation of the base-spacecraft. The mean time when the transition to capture mode occurs and when the chaser captures the target is clearly marked. It is noteworthy to point out that new guidance solutions, delivered every 5 seconds, cause in some instances discontinuities in the commanded forces and torque, as the trajectory is suddenly altered.

Table 5 compares the experimental results obtained with the results predicted by numerical simulations of the same experiments. A noticeable discrepancy is found on the maneuver duration, which is fixed at 70 s. The gripper closure, capturing the grappling fixture, is triggered when the end-effector is within 5 cm from the grapple fixture, thus cutting the maneuver short.

As expected, the numerical simulation, representing an idealized environment, underestimates the amount of impulse required to complete the capture maneuver. Thruster misalignments, variability, and modeling errors make the chaser deviate from the nominal trajectory during the experiments, which, when compensated by the linear-quadratic regulator, result in higher impulses. In addition, and given that all experiments succeeded, the proposed guidance and control exhibits a certain degree of robustness. Although a rigorous study on the guidance robustness has yet to be conducted, the experimental results demonstrate that the proposed guidance is able to cope with the imperfect relative state estimates, the effects of unmodeled dynamics, and the chaser’s actuation uncertainties inherently present in hardware-in-the-loop experimental demonstrations.

7. Conclusions

An approach to guide the capture maneuver of a tumbling RSO by a chaser spacecraft equipped with a robotic manipulator has been proposed. The original guidance problem is made more tractable by dividing the capture maneuver into two sub-maneuvers: a system-wide translation and an internal re-configuration. These two sub-maneuvers are optimized with two consecutive optimizations steps. By combining the results of the two optimization steps a solution to the original guidance problem is obtained. To optimize the two sub-maneuvers in two consecutive steps, they are decoupled, prioritizing the minimization of the system-wide translation at the expense of the internal re-

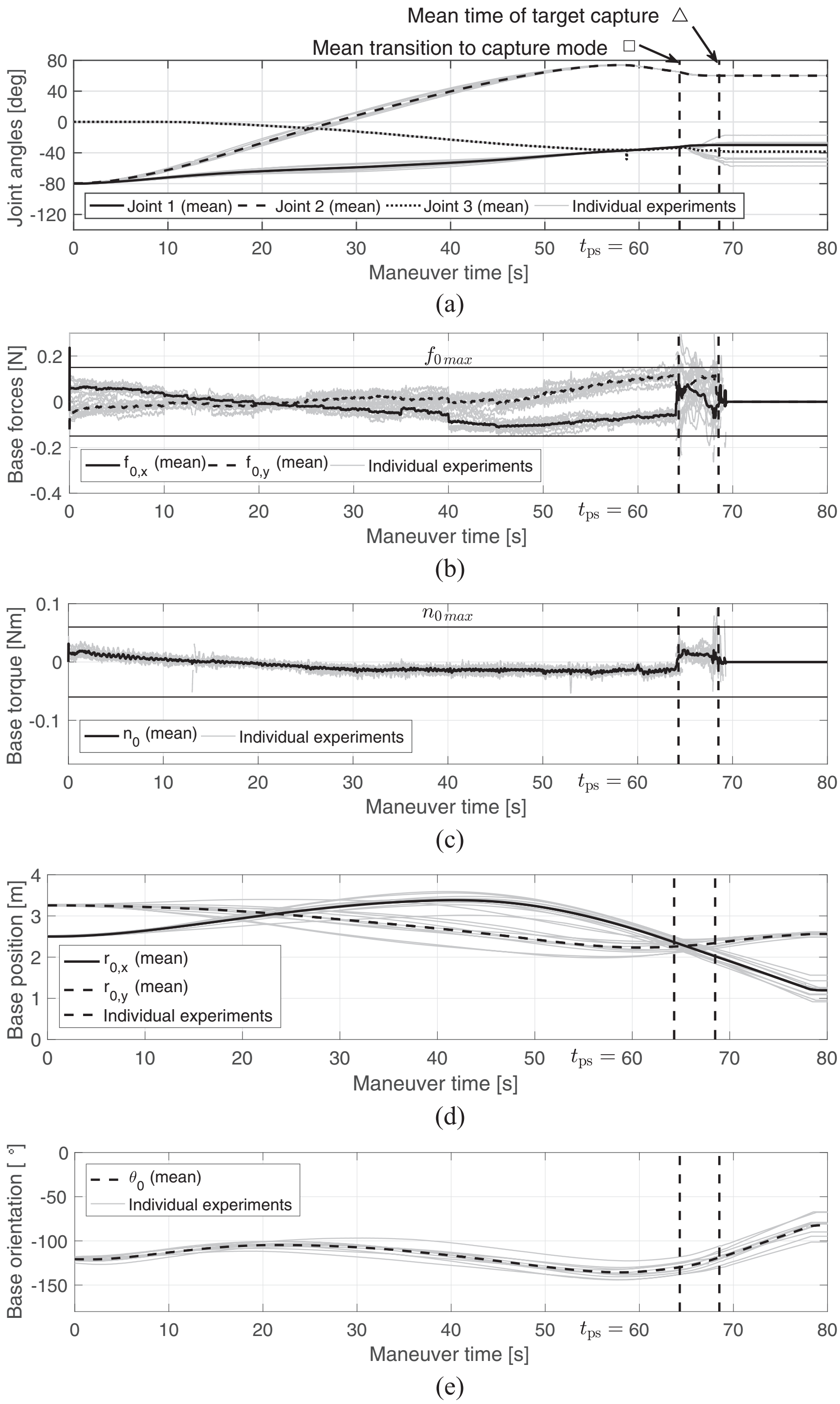


Fig. 31. Experimental results: (a) chaser's manipulator configuration, (b) chaser's thrust forces in components in the inertial frame, (c) chaser's thrust torque, (d) base-spacecraft position in the inertial frame, and (d) base-spacecraft orientation.

configuration cost. The two consecutive optimization steps are transcribed into two non-convex programming problems. A sequential convex programming procedure is used on both instances to overcome the non-convex constraints emerging from the keep-out zones on the system-wide translation and the nonlinear kinematic and dynamics on the internal re-configuration. In summary, the proposed guidance approach casts the original guidance control problem into a collection of convex programming problems. A proof of convergence has been offered for the sequential convex programming used for system-wide translational, expanding previous results. The procedure used to optimize the internal re-configuration does not offer guaranteed convergence and a set of heuristics—trust regions—is used. The results of an extensive numerical simulation campaign provide insight into the performance and robustness of the proposed approach, suggesting that the method can be reliably applied to a wide range of maneuvers. As convex programming offers deterministic convergence properties, the proposed guidance approach is suitable for onboard implementation. This has been confirmed in a set of hardware-in-the-loop experiments where the proposed guidance was implemented on an onboard computer and used, in real-time, to guide a capture maneuver. Although these experiments have not been conducted in a full six-degree-of-freedom environment, they significantly advanced the previously demonstrated state-of-the-art and can be used as evidence to the suitability of the proposed method for onboard implementation.

Future work can be directed to demonstrate maneuvers with more complex chaser or target geometries as well as with more elaborate initial guesses, pre-set manipulator motions, and trust region update laws. Finally, a full three-dimensional guidance implementation case could be demonstrated on-orbit or using a processor-in-the-loop test bed.

Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: Marcello Romano kindly acknowledges the support of the sponsors of this research program at the NPS Spacecraft Robotics Lab.

ORCID iD

Josep Virgili-Llop https://orcid.org/0000-0002-2477-0005

Note

1. The original equations labels are repeated for convenience.

References

Açıkmeşe B, Carson JM and Blackmore L (2013) Lossless convexification of nonconvex control bound and pointing constraints of the soft landing optimal control problem. *IEEE Transactions on Control Systems Technology* 21(6): 2104–2113.

Aghili F (2008) Optimal control for robotic capturing and passivation of a tumbling satellite with unknown dynamics. In: *AIAA Guidance, Navigation, and Control Conference and Exhibit*, volume 21, paper 2008-7274.

Alexander HL and Cannon RH (1989) *Experiments on the Control of a Satellite Manipulator*. Berlin: Springer.

Arthur S, Emde C and Mc Guire N (2007) Assessment of the real-time preemption patches (RT-PREEMPT) and their impact on the general purpose performance of the system. In: *Proceedings of the 9th Real-Time Linux Workshop*.

Bergen Gvd (1999) A fast and robust GJK implementation for collision detection of convex objects. *Journal of Graphics Tools* 4(2): 7–25.

Boyd S and Vandenberghe L (2004) *Convex Optimization*. Cambridge: Cambridge University Press.

Chen VW and Cannon RH (1994) Experiments in nonlinear adaptive control of multi-manipulator free-flying robots. In: *Proceedings 1994 IEEE International Conference on Robotics and Automation*, vol. 3, pp. 2213–2220.

Ciarcià M, Cristi R and Romano MM (2017) Emulating scaled Clohessy–Wiltshire dynamics on an air-bearing spacecraft simulation testbed. *Journal of Guidance, Control, and Dynamics* 40(10): 2496–2510.

Conn AR, Gould NI and Toint PL (2000) *Trust Region Methods*. Philadelphia, PA: SIAM.

Conway BA (2012) A survey of methods available for the numerical optimization of continuous dynamic systems. *Journal of Optimization Theory and Applications* 152(2): 271–306.

Flores-Abad A, Ma O, Pham K and Ulrich S (2014) A review of space robotics technologies for on-orbit servicing. *Progress in Aerospace Sciences* 68: 1–26.

Flores-Abad A, Zhang L, Wei Z and Ma O (2016) Optimal capture of a tumbling object in orbit using a space manipulator. *Journal of Intelligent & Robotic Systems*. DOI:10.1007/s10846-016-0417-1 .

Gilbert EG, Johnson DW and Keerthi SS (1988) A fast procedure for computing the distance between complex objects in three-dimensional space. *IEEE Journal on Robotics and Automation* 4(2): 193–203.

Grant M and Boyd S (2008) Graph implementations for non-smooth convex programs. In: Blondel V, Boyd S and Kimura H (eds.) *Recent Advances in Learning and Control* (*Lecture Notes in Control and Information Sciences*). Berlin: Springer-Verlag, pp. 95–110.

Grant M and Boyd S (2014) CVX: Matlab software for disciplined convex programming, version 2.1. Available at: http://cvxr.com/cvx .

Hanßmann H (1999) Quasi-periodic motion of a rigid body under weak forces. In: *Hamiltonian Systems with Three or More Degrees of Freedom*. Dordrecht: Springer, pp. 398–402.

Hull DG (1997) Conversion of optimal control problems into parameter optimization problems. *Journal of Guidance, Control, and Dynamics* 20(1): 57–60.

Jacobsen S, Lee C, Zhu C and Dubowsky S (2002) Planning of safe kinematic trajectories for free flying robots approaching an uncontrolled spinning satellite. In: *ASME 2002 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. American Society of Mechanical Engineers, pp. 1145–1151.

Karaman S and Frazzoli E (2011) Sampling-based algorithms for optimal motion planning. *The International Journal of Robotics Research* 30(7): 846–894.

Kim Y, Mesbahi M, Singh G and Hadaegh FY (2010) On the convex parameterization of constrained spacecraft reorientation. *IEEE Transactions on Aerospace and Electronic Systems* 46(3): 1097–1109.

Lampariello R (2010) Motion planning for the on-orbit grasping of a non-cooperative target satellite with collision avoidance. In: *International Symposium on Artificial Intelligence, Robotics, and Automation in Space*, vol. 1, pp. 636–643.

Lampariello R and Hirzinger G (2013) Generating feasible trajectories for autonomous on-orbit grasping of spinning debris in a useful time. In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, pp. 5652–5659.

Lee U and Mesbahi M (2016) Constrained autonomous precision landing via dual quaternions and model predictive control. *Journal of Guidance, Control, and Dynamics* 40(2): 292–308.

Lipp T and Boyd S (2016) Variations and extension of the convex–concave procedure. *Optimization and Engineering* 17(2): 263–287.

Liu X, Lu P and Pan B (2017) Survey of convex optimization for aerospace applications. *Astrodynamics* 1(1): 23–40.

Lu P and Liu X (2013) Autonomous trajectory planning for rendezvous and proximity operations by conic optimization. *Journal of Guidance, Control, and Dynamics* 36(2): 375–389.

Lugini C and Romano M (2009) A ballistic-pendulum test stand to characterize small cold-gas thruster nozzles. *Acta Astronautica* 64(5): 615–625.

Ma RR, Odhner LU and Dollar AM (2013) A modular, open-source 3D printed underactuated hand. In: *2013 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, pp. 2737–2743.

Mao Y, Dueri D, Szmuk M and Açkmeşe B (2017) Successive convexification of non-convex optimal control problems with state constraints. *IFAC-PapersOnLine* 50(1): 4063–4069.

Menon C, Busolo S, Cocuzza S, et al. (2007) Issues and solutions for testing free-flying robots. *Acta Astronautica* 60(12): 957–965.

Misra G and Bai X (2017) Optimal path planning for free-flying space manipulators via sequential convex programming. *Journal of Guidance, Control, and Dynamics* 40(11): 3026–3033.

Morgan D, Subramanian GP, Chung SJ and Hadaegh FY (2016) Swarm assignment and trajectory optimization using variable-swarm, distributed auction assignment and sequential convex programming. *The International Journal of Robotics Research* 35(10): 1261–1285.

Nahon M, Damaren C, Bergen A and Goncalves J (1995) A test facility for multi-armed space-based manipulators. *Canadian Aeronautics and Space Journal* 41(4): 150–162.

Nanjangud A, Blacker PC, Bandyopadhyay S and Gao Y (2018) Robotics and AI-enabled on-orbit operations with future generation of small satellites. *Proceedings of the IEEE* 106(3): 429–439.

Nesterov Y and Nemirovskii A (1994) *Interior-Point Polynomial Algorithms in Convex Programming*. Philadelphia, PA: SIAM.

Oda M, Kibe K and Yamagata F (1996) Ets-vii, space robot in-orbit experiment satellite. In: *Proceedings 1996 IEEE International Conference on Robotics and Automation*, vol. 1, pp. 739–744.

Ogilvie A, Allport J, Hannah M and Lymer J (2008) Autonomous satellite servicing using the orbital express demonstration manipulator system. In: *Proceedings of the 9th International Symposium on Artificial Intelligence, Robotics and Automation in Space (i-SAIRAS'08)*, pp. 25–29.

Opromolla R, Fasano G, Rufino G and Grassi M (2017) A review of cooperative and uncooperative spacecraft pose determination techniques for close-proximity operations. *Progress in Aerospace Sciences* 93: 53–72.

Park H, Romano M, Zagaris C, Virgili-Llop J and Zappulla RI (2017) Nonlinear model predictive control for spacecraft rendezvous and docking with a rotating target. In: *27th AAS/AIAA Spaceflight Mechanics Meeting*, San Antonio, TX, 6–9 February 2017.

Persson SM and Sharf I (2015) Ground-based experiments towards the interception of non-cooperative space debris with a robotic manipulator. In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5441–5446.

Pillo GD and Grippo L (1989) Exact penalty functions in constrained optimization. *SIAM Journal on Control and Optimization* 27(6): 1333–1360.

Pinson R and Lu P (2016) Trajectory design employing convex optimization for landing on irregularly shaped asteroids. In: *AIAA/AAS Astrodynamics Specialist Conference*, Long Beach, CA, 13–16 September 2016. American Institute of Aeronautics and Astronautics.

Russakow J, Khatib O and Rock SM (1995) Extended operational space formulation for serial-to-parallel chain (branching) manipulators. In: *Proceedings 1995 IEEE International Conference on Robotics and Automation*, vol. 1, pp. 1056–1061.

Sabatini M, Gasbarri P and Palmerini GB (2017) Coordinated control of a space manipulator tested by means of an air bearing free floating platform. *Acta Astronautica* 139: 296–305.

Schulman J, Duan Y, Ho J, et al. (2014) Motion planning with sequential convex optimization and convex collision checking. *The International Journal of Robotics Research* 33(9): 1251–1270.

Schwartz JL, Peck MA and Hall CD (2003) Historical review of air-bearing spacecraft simulators. *Journal of Guidance, Control, and Dynamics* 26(4): 513–522.

Shan M, Guo J and Gill E (2016) Review and comparison of active space debris capturing and removal methods. *Progress in Aerospace Sciences* 80: 18–32.

Shoemake K (1995) Uniform random rotations. In: *Graphics Gems III*. AP Professional, pp. 124–132.

Siciliano B, Sciavicco L, Villani L and Oriolo G (2009) *Robotics Modelling, Planning and Control*. New York: Springer.

Starek JA, Schmerling E, Maher GD, Barbee BW and Pavone M (2016) Fast, safe, propellant-efficient spacecraft motion planning under Clohessy–Wiltshire–Hill dynamics. *Journal of Guidance, Control, and Dynamics* 40(2): 418–438.

Sternberg DC and Miller D (2018) Parameterization of fuel-optimal synchronous approach trajectories to tumbling targets. *Frontiers in Robotics and AI* 5: 33.

Szmuk M and Acikmese B (2018) *Successive Convexification for 6-DoF Mars Rocket Powered Landing with Free-Final-Time*. American Institute of Aeronautics and Astronautics.

Toglia C, Kennedy F and Dubowsky S (2011) Cooperative control of modular space robots. *Autonomous Robots* 31(2–3): 209–221.

Toh KC, Todd MJ and Tütüncü RH (1999) SDPT3—a MATLAB software package for semidefinite programming, version 1.3. *Optimization Methods and Software* 11(1-4): 545–581.

Ullman MA and Cannon RH (1993) *Experiments in Autonomous Navigation and Control of a Multi-Manipulator, Free-Flying Space Robot*. Boston, MA: Springer, pp. 269–284.

Umetani Y and Yoshida K (1989) Experimental study on two-dimensional free-flying robot satellite model. In: *Proceedings of NASA Conference on Space Telerobotics*, vol. 5, pp. 215–224.

Umetani Y and Yoshida K (2001) Workspace and manipulability analysis of space manipulator. *Transactions of the Society of Instrument and Control Engineers* E-1(1): 116–123.

Ventura J, Romano M and Walter U (2015) Performance evaluation of the inverse dynamics method for optimal spacecraft reorientation. *Acta Astronautica* 110: 266–278.

Verscheure D, Demeulenaere B, Swevers J, Schutter JD and Diehl M (2009) Time-optimal path tracking for robots: A convex optimization approach. *IEEE Transactions on Automatic Control* 54(10): 2318–2327.

Virgili-Llop J (2017) SPART: Spacecraft robotics toolkit. Available at: https://github.com/NPS-SRL/SPART .

Virgili-Llop J, Drew J and Romano M (2016a) Design and parameter identification by laboratory experiments of a prototype modular robotic arm for orbiting spacecraft applications. In: *6th International Conference on Astrodynamics Tools and Techniques (ICATT)*, 14–17 March 2016, Darmstadt, Germany.

Virgili-Llop J, Drew J, Zappulla R II and Romano M (2016b) Autonomous capture of a resident space object by a spacecraft with a robotic manipulator: Analysis, simulation and experiments. In: *AIAA/AAS Astrodynamics Specialist Conference*, Long Beach, CA, 13–16 September 2016. American Institute of Aeronautics and Astronautics.

Virgili-Llop J, Drew JV, Zappulla R and Romano M (2017a) Laboratory experiments of resident space object capture by a spacecraft–manipulator system. *Aerospace Science and Technology* 71(C): 530–545.

Virgili-Llop J, Zagaris C, Park H, Zappulla R and Romano M (2018) Experimental evaluation of model predictive control and inverse dynamics control for spacecraft proximity and docking maneuvers. *CEAS Space Journal* 10(1): 37–49.

Virgili-Llop J, Zagaris C, Zappulla R II, Bradstreet A and Romano M (2017b) Convex optimization for proximity maneuvering of a spacecraft with a robotic manipulator. In: *27th AAS/AIAA Space Flight Mechanics Meeting*, San Antonio, TX, 5–9 February 2017.

Virgili-Llop J, Zagaris C, Zappulla R II, Bradstreet A and Romano M (2017c) Laboratory experiments on the capture of a tumbling object by a spacecraft–manipulator system using a convex-programming-based guidance. In: *AAS/AIAA Astrodynamics Specialist Conference*, Stevenson, WA, 21–24 August (AAS 17-734).

Wächter A and Biegler TL (2005) On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming* 106(1): 25–57.

Watterson M, Smith T and Kumar V (2016) Smooth trajectory generation on SE(3) for a free flying space robot. In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5459–5466.

Wilde M, Ciarcià M, Grompone A and Romano M (2016) Experimental characterization of inverse dynamics guidance in docking with a rotating target. *Journal of Guidance, Control, and Dynamics* 39(6): 1173–1187.

Xu W, Liang B, Li C, Liu Y and Xu Y (2009) Autonomous target capturing of free-floating space robot: Theory and experiments. *Robotica* 27(03): 425–445.

Xu W, Liang B and Xu Y (2011) Survey of modeling, planning, and ground verification of space robotic systems. *Acta Astronautica* 68(11–12): 1629–1649.

Yoshida K (1994) Experimental study on the dynamics and control of a space robot with experimental free-floating robot satellite. *Advanced Robotics* 9(6): 583–602.

Yoshida K, Dimitrov D and Nakanishi H (2006) On the capture of tumbling satellite by a space robot. In: *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4127–4132.

Yuille AL and Rangarajan A (2003) The concave–convex procedure. *Neural Computation* 15(4): 915–936.

Zagaris C, Park H, Virgili-Llop J, Zappulla R II, Romano M and Kolmanovsky I (2018) Model predictive control of spacecraft relative motion with convexified keep-out-zone constraints. *Journal of Guidance, Control, and Dynamics*, in press.

Zappulla RI, Virgili-Llop J and Romano M (2017) Near-optimal real-time spacecraft guidance and control using harmonic potential functions and a modified RRT. In: *27th AAS/AIAA Space Flight Mechanics Meeting*, San Antonio, TX.

Zappulla R II, Virgili-Llop J and Romano M (2017a) Spacecraft thruster control via sigma–delta modulation. *Journal of Guidance, Control, and Dynamics* 40(11): 2928–2933.

Zappulla R II, Virgili-Llop J, Zagaris C, Park H and Romano M (2017b) Dynamic air-bearing hardware-in-the-loop testbed to experimentally evaluate autonomous spacecraft proximity maneuvers. *Journal of Spacecraft and Rockets* 54(4): 825–839.

Appendix. Index to multimedia extensions

Archives of IJRR multimedia extensions published prior to 2014 can be found at http://www.ijrr.org, after 2014 all videos are available on the IJRR YouTube channel at http://www.youtube.com/user/ijrrmultimedia

Table of Multimedia Extensions

Extension	Media type	Description
1	Video	Example of a simulated maneuver
2	Video	Experimental demonstration on the POSEIDYN test bed