Minimum-Impulse-Bit-Aware Allocation for Manipulator-Mounted Spacecraft Thrusters

William Schwend[*] *and Hanspeter Schaub*[**]
[*]*University of Colorado Boulder*
Boulder, CO 80303, USA; william.schwend@colorado.edu
[**]*University of Colorado Boulder*
Boulder, CO 80303, USA; hanspeter.schaub@colorado.edu

Abstract

Manipulator-mounted thrusters provide reconfigurable spacecraft control, but existing allocation methods do not directly enforce minimum impulse bit (MIB) constraints. This paper formulates coupled joint-and-thruster allocation as a mixed-integer nonlinear program (MINLP) with explicit MIB constraints. A staged procedure reduces computational cost through screening, feasible repair, and warm-started MINLP fallback. Numerical studies spanning minimum firing durations of 10–200 ms demonstrate improved executable control-wrench accuracy relative to post-allocation rounding, with greater computational savings at shorter durations. An 8 h proximity-operations simulation demonstrates improved position and pointing performance at the cost of increased applied impulse and allocation runtime.

1. Introduction

The need to service, assemble, upgrade, and eventually dispose of spacecraft on orbit has made in-space servicing, assembly, and manufacturing (ISAM) and active debris removal priorities for commercial and national space interests [1–3]. These capabilities are further motivated by the value of extending spacecraft lifetimes [4, 5] and the increasing difficulty of launching complex spacecraft in a fully integrated state [6]. Many such missions require close-proximity operations and robotic manipulation. Robotic arms have supported proximity operations since the Space Shuttle servicing missions to the Hubble Space Telescope [7]. They remain central to concepts such as NASA's former OSAM-1 mission and the proposed ClearSpace-1 debris-removal mission [8, 9].

Spacecraft proximity-operations control has traditionally relied on body-fixed thruster arrays, often augmented by reaction wheels, control moment gyroscopes, or variable-speed control moment gyroscopes for attitude control and momentum management [7, 10–13]. These arrays permit well-established allocation methods to map a desired control wrench into individual thruster commands, including lookup tables, Moore–Penrose-pseudoinverse methods, linear and quadratic programming, mixed-integer linear programming, and Lyapunov-based methods [13–16]. However, providing independent six-degree-of-freedom (6-DOF) authority with body-fixed actuators typically requires numerous thrusters. The additional actuators and propellant-feed hardware increase spacecraft mass, while nonideal geometry can require opposing firings that increase propellant consumption [17].

Manipulator-mounted propulsion instead makes the thruster geometry an allocation variable. Servicing concepts including Northrop Grumman's Mission Extension Vehicle (MEV) and Mission Extension Pod (MEP) [18], together with Astroscale's Life Extension In-orbit (LEXI) [19], illustrate architectures in which arm motion varies each thruster's direction and moment arm relative to the spacecraft. Adapting this geometry to the requested control wrench can reduce thrust cancellation and, in some designs, provide full 6-DOF control with fewer thrusters than a body-fixed array [20–25].

Arm-mounted thrusters have been investigated as a control source for slowly varying applications, including geostationary Earth orbit (GEO) station keeping [20–22] and geostationary transfer orbit (GTO)-to-GEO transfer [26], as well as for proximity operations [23–25, 27, 28]. Across these applications, most approaches determine the joint configuration and thruster commands through optimization. The joint-and-thruster decisions may be included directly in the guidance and control formulation or separated into a nonlinear allocation problem [20–22, 24, 25, 27]. Reference 23 instead

[*]https://news.northropgrumman.com/satellites/northrop-grummans-spacelogistics-continues-revolutionary-satellite-life-extension-work-with-sale-of-third-mission-extension-pod

[†]https://www.astroscale-us.com/en/news/astroscale-israel-completes-construction-of-new-computer-vision-robotics-lab-and-clean-room

Copyright © 2026 by William Schwend and Hanspeter Schaub. Published by the i-SAIRAS association with permission.



(a) MEV mission concept art*



(b) LEXI mission concept art†

Figure 1: Concept figures of spacecraft with thrusters mounted on robotic arms

uses numerical inverse kinematics to determine the joint configuration required to produce the desired wrench using fixed-magnitude thrusters.

None of these arm-mounted-thruster allocation approaches directly selects the joint configuration and firing commands subject to a minimum impulse bit (MIB) constraint. For a fixed-thrust actuator, the MIB is the smallest repeatable impulse and corresponds to a minimum nonzero firing duration. Over a finite allocation interval, it therefore creates a forbidden command region between zero and the minimum nonzero firing command. Existing arm-mounted-thruster formulations either omit the MIB constraint or enforce it after continuous allocation using rounding or firing logic [24, 25, 27, 29] or error-prediction-based quantization [21]. When the MIB is enforced after allocation, the joint configuration has already been selected for a continuous firing vector that may not be executable. The post-allocation method therefore produces a different applied firing vector, changing the realized control wrench and potentially reducing control accuracy.

Explicit MIB constraints have previously been incorporated into the allocation of conventional fixed thruster arrays. Reference 15, for example, formulates static MIB-aware allocation as a mixed-integer linear program (MILP) that jointly selects thruster activation and firing duration. Existing arm-mounted-thruster allocation formulations have not included an equivalent discrete firing model. In this case, the joint configuration changes both the thrust directions and moment arms through nonlinear kinematics. Directly enforcing the MIB while simultaneously selecting the joint configuration and firing commands therefore produces a mixed-integer nonlinear program (MINLP). Although this formulation addresses the MIB during allocation, repeatedly solving the resulting nonconvex MINLP can impose a substantial computational burden.

To address these formulation and computational challenges, this work makes four contributions. First, the coupled joint-and-thruster allocation problem is formulated as an MINLP that directly enforces the MIB while jointly selecting the arm configuration, thruster activation pattern, and firing commands. Second, a staged MIB-aware allocation procedure is developed to reduce the number and computational cost of MINLP solves through no-fire and continuous nonlinear-program (NLP) screening, finite MIB-feasible repair, and a warm-started MINLP fallback. Third, Monte Carlo studies quantify the allocation accuracy and computational performance of the direct and staged approaches across several minimum firing durations. Finally, the complete pipeline is compared with endpoint rounding in a multi-hour proximity-operations simulation to evaluate the resulting closed-loop accuracy, applied impulse, and allocation-runtime tradeoffs.

The remainder of this paper is organized as follows. Section 2 defines the coupled joint-and-thruster allocation problem and its baseline continuous-NLP formulation. Section 3 introduces the direct MINLP formulation and staged MIB-aware allocation procedure. Section 4 presents the allocation-level numerical studies and a closed-loop comparison between the staged allocation pipeline and post-allocation rounding. Finally, the conclusions are summarized in Section 5.

2. Manipulator-Mounted Thruster Allocation Problem

The coupled joint-and-thruster control-allocation process considered in this work occurs each time the spacecraft controller provides a new 6-DOF control wrench of the form

$$\boldsymbol{u} = \begin{bmatrix} \boldsymbol{F}_{\text{thr}} \\ \boldsymbol{L}_{\text{thr}} \end{bmatrix} \tag{1}$$

where $\boldsymbol{F}_{\text{thr}}$ and $\boldsymbol{L}_{\text{thr}}$ are the desired control-force and control-torque vectors, respectively. The allocation problem determines the joint configuration and thruster commands that best produce $\boldsymbol{u}$ while satisfying the geometric, actuator, and control constraints of the manipulator-mounted system. The actuators considered throughout this work are on–off, fixed-thrust devices whose delivered impulse is controlled through firing duration rather than thrust magnitude. The

allocation formulation treats $\boldsymbol{u}$ as an input and is independent of the upstream control law used to generate it. Accordingly, the discussion here focuses on mapping the requested wrench into joint and thruster commands.

2.1 Thruster-Wrench Mapping

In an arm-mounted-thruster system, joint reconfiguration changes both the force direction and the moment arm of each thruster relative to the instantaneous system center of mass (CoM). The allocation problem must therefore relate the joint configuration and equivalent interval-averaged force commands to the resulting spacecraft control wrench. Prior arm-mounted-thruster allocation formulations represent this relationship using a configuration-dependent thruster-wrench mapping [21, 24]. The formulation considered here accommodates an arbitrary number of articulated arms, revolute joints, and arm-mounted thrusters.

All arms are attached to a rigid spacecraft hub. Each joint is modeled as a single-DOF revolute joint with frame $\mathcal{J}_j$ centered at point J_j and unit spin axis $\hat{s}_j$. The joint angles are collected in the configuration vector $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_{N_J}]^T \in \mathbb{R}^{N_J}$, where N_J is the total number of joints in the system. Similarly, frame $\mathcal{T}_i$ is attached to the ith thruster and centered at its force-application point T_i. The force produced by the thruster acts along a fixed axis of $\mathcal{T}_i$, whose configuration-dependent unit direction is denoted by $\hat{f}_i(\boldsymbol{\theta})$.

An example spacecraft containing two 4-DOF arms with one thruster mounted on each arm is shown in Fig. 2. The instantaneous system CoM is located at point C. The vector from C to the force-application point of thruster i is denoted by $\boldsymbol{r}_{T_i/C}(\boldsymbol{\theta})$ and provides the corresponding moment arm.

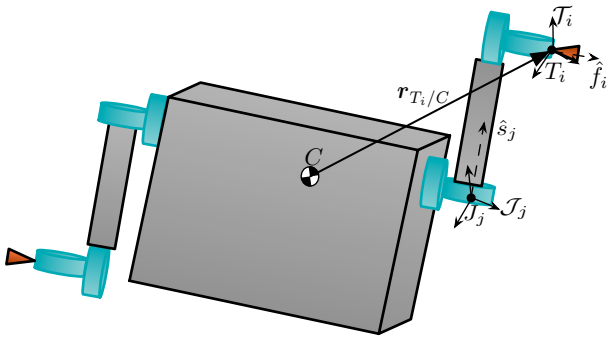


Figure 2: Example manipulator-mounted-thruster geometry and system-CoM moment arm used to construct the control wrench.

For a specified joint configuration, forward kinematics determine the position and force direction of each thruster and the location of the instantaneous system CoM. Let $\boldsymbol{f} = [f_1, \ldots, f_{N_T}]^T \in \mathbb{R}^{N_T}$ contain the equivalent interval-averaged thruster-force commands, where N_T is the total number of thrusters. When active, thruster i produces the fixed force magnitude $f_{\max,i}$. Over an allocation-update interval of duration Δt_{alloc}, the continuous command f_i corresponds to the total impulse and firing duration

$$I_i = f_i \Delta t_{\text{alloc}}, \qquad t_{i,\text{fire}} = \frac{I_i}{f_{\max,i}} = \frac{f_i}{f_{\max,i}} \Delta t_{\text{alloc}}. \tag{2}$$

Thus, f_i represents the equivalent force obtained by modulating the firing duration; it is not a throttle setting for the physical thruster. The corresponding interval-averaged control wrench is

$$\boldsymbol{u}_{\text{ctl}}(\boldsymbol{\theta}, \boldsymbol{f}) = [T(\boldsymbol{\theta})]\, \boldsymbol{f} \tag{3}$$

where $[T(\boldsymbol{\theta})] \in \mathbb{R}^{6 \times N_T}$ is the configuration-dependent thruster-wrench mapping matrix defined by

$$[T(\boldsymbol{\theta})] = \begin{bmatrix} \hat{f}_1(\boldsymbol{\theta}) & \cdots & \hat{f}_{N_T}(\boldsymbol{\theta}) \\ \boldsymbol{r}_{T_1/C}(\boldsymbol{\theta}) \times \hat{f}_1(\boldsymbol{\theta}) & \cdots & \boldsymbol{r}_{T_{N_T}/C}(\boldsymbol{\theta}) \times \hat{f}_{N_T}(\boldsymbol{\theta}) \end{bmatrix} \tag{4}$$

Each column of $[T(\boldsymbol{\theta})]$ therefore gives the force and system-CoM moment produced per unit force by one thruster. Because both the force directions and moment arms vary with the joint configuration, reconfiguring the arms changes the wrench generated by a given set of equivalent force commands.

2.2 Continuous-NLP Allocation

The baseline allocator follows the continuous joint-and-thruster optimization introduced in Ref. [21] and subsequently adapted for 6-DOF arm-mounted-thruster proximity operations control in Refs. [24, 25]. Given a desired control wrench $\boldsymbol{u}$, the allocator selects the joint configuration and equivalent thruster-force commands by solving

$$\begin{aligned} \min_{\boldsymbol{\theta}^*, \boldsymbol{f}^*} \quad & J\left(\boldsymbol{\theta}^*, \boldsymbol{f}^*\right) \\ \text{subject to} \quad & \boldsymbol{\theta}_{\min} \leq \boldsymbol{\theta}^* \leq \boldsymbol{\theta}_{\max}, \\ & 0 \leq f_i^* \leq f_{\max,i}, \qquad \forall i \in \mathcal{I}_{\text{thr}} \end{aligned} \tag{5}$$

Here, $\mathcal{I}_{\text{thr}} \triangleq \{1, \ldots, N_T\}$ is the thruster index set, while $\boldsymbol{\theta}_{\min}$ and $\boldsymbol{\theta}_{\max}$ contain the joint-angle bounds. The upper force bound corresponds to firing thruster i for the complete allocation-update interval. The cost function $J\left(\boldsymbol{\theta}, \boldsymbol{f}\right)$ is defined as

$$J\left(\boldsymbol{\theta}, \boldsymbol{f}\right) = \left(\boldsymbol{u} - \boldsymbol{u}_{\text{ctl}}\left(\boldsymbol{\theta}, \boldsymbol{f}\right)\right)^T \left[W_u\right] \left(\boldsymbol{u} - \boldsymbol{u}_{\text{ctl}}\left(\boldsymbol{\theta}, \boldsymbol{f}\right)\right) + \boldsymbol{W}_f^T \boldsymbol{f} \tag{6}$$

The positive-definite matrix $[W_u] \in \mathbb{R}^{6\times 6}$ weights the control-wrench tracking error and sets the relative importance of its force and torque components. The vector $\boldsymbol{W}_f \in \mathbb{R}^{N_T}$ has strictly positive entries that penalize the force allocated to each thruster. Because $I_i = f_i \Delta t_{\text{alloc}}$ and the allocation interval is fixed, $\boldsymbol{W}_f^T \boldsymbol{f}$ is proportional to the corresponding weighted impulse expenditure; the constant factor Δt_{alloc} can be absorbed into $\boldsymbol{W}_f$.
Through Eq. (3), the joint configuration changes the force and moment contribution represented by each column of $[T(\boldsymbol{\theta})]$, while the entries of $\boldsymbol{f}$ scale those contributions. The allocation problem is therefore nonlinear in the joint angles but linear in the thruster-force commands for a fixed joint configuration. Solving Eq. (5) returns $\boldsymbol{\theta}^*$ and $\boldsymbol{f}^*$, which determine the allocated wrench and the nominal firing durations through Eq. (2).

3. MIB-Aware Allocation Framework

3.1 MIB Constraint and Allocation Implications

The continuous command-to-duration mapping in Eq. (2) permits any firing duration between zero and Δt_{alloc}. A physical fixed-thrust actuator instead has a minimum impulse bit $I_{\text{MIB},i}$, corresponding to the minimum nonzero firing duration

$$t_{\min,i} = \frac{I_{\text{MIB},i}}{f_{\max,i}}. \tag{7}$$

Expressed using the equivalent force variable employed by the allocator, this minimum firing duration produces the minimum nonzero command

$$f_{\min,i} \triangleq \frac{I_{\text{MIB},i}}{\Delta t_{\text{alloc}}} = \frac{f_{\max,i} t_{\min,i}}{\Delta t_{\text{alloc}}}. \tag{8}$$

An executable allocated-force command must therefore satisfy

$$f_i = 0 \qquad \text{or} \qquad f_i \geq f_{\min,i}, \qquad \forall i \in \mathcal{I}_{\text{thr}}. \tag{9}$$

A value in the interval $0 < f_i < f_{\min,i}$ would require a firing duration shorter than $t_{\min,i}$. Because the actuator is fixed thrust, this command cannot instead be realized by reducing the instantaneous force magnitude. The continuous NLP in Section 2.2 includes this forbidden interval within its force bounds and can therefore return commands that are not directly executable.
Post-allocation methods, including error-prediction-based quantization [21], rounding, and Schmitt firing triggers [29], convert these continuous commands into executable firings. However, modifying the firing commands after allocation can change the realized control wrench because the joint configuration and firing vector were selected without accounting for the MIB.
For arm-mounted thrusters, the reconfigurable geometry provides an alternative: the joint configuration and firing commands can be selected together from the MIB-feasible command set. This section first formulates this direct MIB-aware allocation as an MINLP. It then develops a staged solution framework that reduces how often the MINLP must be invoked and provides an MIB-feasible initial solution when the discrete solve is required. The direct formulation and staged framework therefore address distinct aspects of the problem: the former incorporates the MIB into allocation, while the latter addresses the computational cost of applying that formulation repeatedly.

3.2 Direct MINLP Formulation

The continuous NLP allows each thruster command to take any value between zero and its maximum magnitude. To enforce Eq. (9) within the allocation process, binary activation variables are introduced and these continuous firing bounds are replaced by binary-linked bounds. The resulting MINLP is

$$\begin{aligned} \min_{\boldsymbol{\theta}^*, \boldsymbol{f}^*, \boldsymbol{z}} \quad & J\left(\boldsymbol{\theta}^*, \boldsymbol{f}^*\right) \\ \text{subject to} \quad & \boldsymbol{\theta}_{\min} \leq \boldsymbol{\theta}^* \leq \boldsymbol{\theta}_{\max}, \\ & z_i f_{\min,i} \leq f_i^* \leq z_i f_{\max,i}, \qquad \forall i \in \mathcal{I}_{\text{thr}}, \\ & z_i \in \{0,1\}, \qquad \forall i \in \mathcal{I}_{\text{thr}}, \end{aligned} \tag{10}$$

where z_i indicates whether the ith thruster is active. When $z_i = 0$, the binary-linked bounds enforce $f_i^* = 0$. When $z_i = 1$, they instead require $f_{\min,i} \leq f_i^* \leq f_{\max,i}$. The original allocation objective and all other continuous constraints remain unchanged.

Directly incorporating these bounds ensures that the wrench-tracking and thruster-effort terms are evaluated using an MIB-feasible firing vector, rather than a continuous vector that must subsequently be modified for execution. This distinction is particularly important for arm-mounted thrusters. In addition to selecting which thrusters are active and their equivalent force commands, the allocator can modify the joint configuration to change each thruster's force direction and moment arm. The reconfigurable geometry therefore provides additional freedom to satisfy the requested wrench while requiring every commanded firing to be either zero or above its minimum value. This joint adjustment of geometry and firing commands can reduce the wrench error introduced by post-allocation rounding, although doing so may require additional thrust or force cancellation as captured by the original allocation objective. Equation (10) therefore defines the proposed direct MIB-aware allocation problem.

3.3 Staged MIB-Aware Allocation

The MINLP in Eq. (10) provides a direct MIB-aware allocation, but its binary activation variables can make it substantially more computationally expensive than the continuous NLP. The staged framework in Fig. 3 complements this formulation by reducing how often the discrete solve is required and by providing a feasible initial solution when it cannot be avoided. Each stage either returns an executable allocation or passes solution information forward, with the MINLP retained as the final fallback.

The staged procedure coordinates calls to existing NLP and MINLP solvers without modifying their internal optimization algorithms. The sections that follow derive the screening conditions, repair operation, and warm-start procedure used within this framework.

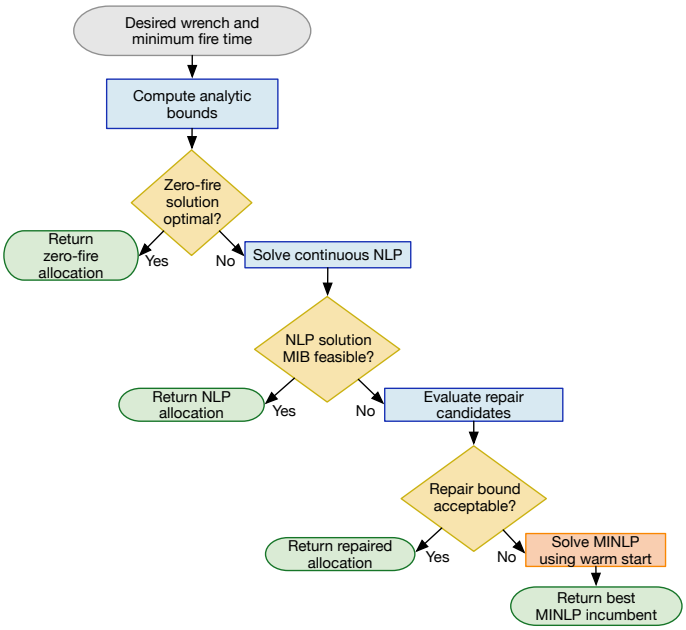


Figure 3: Staged MIB-aware allocation using no-fire and continuous-NLP screening, command repair, and a warm-started MINLP fallback.

3.3.1 No-Fire Screening

The no-fire screen uses analytic objective bounds to determine whether any nonzero MIB-feasible allocation can improve upon the zero-fire command. If no thrusters are commanded, the generated control wrench is zero and the thruster-effort term vanishes. The corresponding objective value is therefore

$$J_{\text{zero}} = \boldsymbol{u}^T [W_u] \boldsymbol{u} \tag{11}$$

Because the zero-fire command is feasible for both the continuous NLP and the MIB-constrained formulation, it provides an upper bound on the globally optimal objective value of either problem. Moreover, every MIB-feasible allocation also satisfies the constraints of the continuous NLP, so

$$J^*_{\text{NLP}} \leq J^*_{\text{MIB}} \leq J_{\text{zero}} \tag{12}$$

where J^*_{NLP} and J^*_{MIB} denote the globally optimal objective values of the continuous and MIB-constrained allocation problems, respectively.

A complementary lower bound applies to the nonzero portion of the MIB-feasible solution space. Any such allocation activates at least one thruster $k \in \mathcal{I}_{\text{thr}}$, for which $f_k \geq f_{\min,k}$. Because the wrench-tracking term and the remaining thruster-effort contributions are nonnegative, its cost satisfies

$$\begin{aligned} J(\boldsymbol{\theta}, \boldsymbol{f}) &\geq \boldsymbol{W}_f^T \boldsymbol{f} \\ &\geq W_{f,k} f_k \\ &\geq W_{f,k} f_{\min,k} \\ &\geq \min_{i \in \mathcal{I}_{\text{thr}}} \left(W_{f,i} f_{\min,i} \right) \triangleq J_{\text{floor}} \end{aligned} \tag{13}$$

Here, $W_{f,k}$ is the kth entry of $\boldsymbol{W}_f$. Thus, J_{floor} is a theoretical lower bound on the cost of every nonzero MIB-feasible allocation, corresponding to zero wrench-tracking error and only a single thruster firing at its minimum command. Because such an allocation may not exist, J_{floor} need not be attainable.

If the following inequality is true

$$J_{\text{zero}} \leq J_{\text{floor}} \tag{14}$$

then no nonzero MIB-feasible allocation can improve upon the zero-fire solution. Consequently,

$$J^*_{\text{MIB}} = J_{\text{zero}} \tag{15}$$

and a globally optimal firing command is

$$\boldsymbol{f}^* = \boldsymbol{0} \tag{16}$$

The desired wrench can therefore be screened using Eq. (14) before either the continuous NLP or the MINLP is solved. When this condition is not satisfied, a nonzero allocation may provide a lower cost, and the procedure continues to the continuous-NLP screening stage.

3.3.2 Continuous-NLP Screening

When Eq. (14) is not satisfied, a nonzero MIB-feasible allocation may improve upon the zero-fire solution. Solving the MINLP in Eq. (10) directly would enforce the MIB constraints, but the binary activation variables make this formulation substantially more computationally expensive than the continuous NLP.

The continuous NLP can therefore serve as a computationally inexpensive preliminary solve. If the returned firing commands already satisfy the MIB constraints, the allocation is accepted and the MINLP stage is bypassed. The potential utility of this screen can be estimated retrospectively from the continuous-NLP commands generated by the spacecraft inspection simulations in Refs. [24, 25]. Although those studies did not impose MIB constraints within the allocation, they applied a 10 ms minimum-firing-duration rounding rule after allocation. The underlying continuous-NLP histories show that only 3–14% of the nonzero thruster commands fell within the corresponding forbidden MIB interval. These results indicate that sub-MIB commands were relatively infrequent under the operating conditions considered and motivate screening the continuous solution before invoking the MINLP.

Let the continuous NLP return the allocation $\left(\boldsymbol{\theta}^{\text{NLP}}, \boldsymbol{f}^{\text{NLP}}\right)$, with objective value

$$J_{\text{NLP}} = J\left(\boldsymbol{\theta}^{\text{NLP}}, \boldsymbol{f}^{\text{NLP}}\right) \tag{17}$$

The set of thrusters whose commands lie within the forbidden MIB interval is defined as

$$\mathcal{V} = \left\{ i \in \mathcal{I}_{\text{thr}} \,\middle|\, 0 < f_i^{\text{NLP}} < f_{\min,i} \right\} \tag{18}$$

If $\mathcal{V} = \varnothing$ then every firing command satisfies

$$f_i^{\text{NLP}} = 0 \quad \text{or} \quad f_i^{\text{NLP}} \geq f_{\min,i} \qquad \forall i \in \mathcal{I}_{\text{thr}} \tag{19}$$

Because the NLP solution also satisfies all constraints shared by the two formulations, it is MIB feasible and can be accepted without solving the MINLP. If $\mathcal{V} \neq \varnothing$, the continuous solution cannot be executed directly and is instead passed to the MIB-feasible repair stage. In implementation, the comparisons in Eq. (18) can be evaluated using a prescribed thruster-command feasibility tolerance.

3.3.3 MIB-Feasible Solution Repair

When $\mathcal{V} \neq \varnothing$, the continuous-NLP solution cannot be executed directly. Rather than immediately invoking the MINLP, an MIB-feasible solution is first constructed while retaining the joint configuration selected by the continuous NLP. Each violating thruster command is replaced by either zero or its minimum allowable value, whereas each nonviolating command remains unchanged. The resulting repair set is defined as

$$\mathcal{R} = \left\{ \bar{\boldsymbol{f}} \in \mathbb{R}^{N_T} \,\middle|\, \bar{f}_i \in \begin{cases} \{0, f_{\min,i}\}, & i \in \mathcal{V}, \\ \{f_i^{\text{NLP}}\}, & i \notin \mathcal{V} \end{cases} \quad \forall i \in \mathcal{I}_{\text{thr}} \right\} \tag{20}$$

This set contains $2^{|\mathcal{V}|}$ candidate firing vectors. The candidates are evaluated jointly because the wrench contributions from different thrusters are coupled through the tracking term in the objective; independently selecting the lower-cost repair for each thruster does not necessarily produce the lowest-cost combined allocation.
The joint configuration is fixed at the continuous-NLP solution:

$$\boldsymbol{\theta}^{\text{R}} = \boldsymbol{\theta}^{\text{NLP}} \tag{21}$$

No additional nonlinear optimization is performed during this repair. The existing allocation objective is evaluated directly for each candidate in $\mathcal{R}$, and the repaired firing command is selected as

$$\boldsymbol{f}^{\text{R}} = \underset{\bar{\boldsymbol{f}} \in \mathcal{R}}{\arg\min}\, J\left(\boldsymbol{\theta}^{\text{R}}, \bar{\boldsymbol{f}}\right) \tag{22}$$

with corresponding objective value

$$J_{\text{R}} = J\left(\boldsymbol{\theta}^{\text{R}}, \boldsymbol{f}^{\text{R}}\right) \tag{23}$$

Because the no-fire screen has already failed, the optimal MIB-constrained objective value satisfies

$$J_{\text{floor}} \leq J_{\text{MIB}}^{*} \leq J_{\text{R}} \tag{24}$$

where the upper bound follows from the MIB feasibility of the repaired solution. Consequently,

$$0 \leq J_{\text{R}} - J_{\text{MIB}}^{*} \leq J_{\text{R}} - J_{\text{floor}} \tag{25}$$

The conservative upper bound on the repaired solution's suboptimality is therefore defined as

$$\overline{\Delta J}_{\text{R}} \triangleq J_{\text{R}} - J_{\text{floor}}. \tag{26}$$

The repaired solution is accepted when

$$\overline{\Delta J}_{\text{R}} \leq \Delta J_{\max} \tag{27}$$

where $\Delta J_{\max}$ is the maximum permissible absolute objective-value suboptimality. Satisfying Eq. (27) guarantees that $J_{\text{R}} - J_{\text{MIB}}^{*} \leq \Delta J_{\max}$. If the condition is not satisfied, the repaired solution is instead used to initialize the MINLP.

3.3.4 Warm-Started MINLP Fallback

If the repaired allocation does not satisfy Eq. (27), the bound on its suboptimality is too large for the solution to be accepted directly. The repaired allocation remains MIB feasible, however, and can therefore be used to initialize the MINLP rather than being discarded. This differs from directly initializing the MINLP with the continuous-NLP solution, whose sub-MIB firing commands may violate the binary-linked thrust bounds in Eq. (10).
A binary activation value is assigned to each repaired thruster command according to

$$z_i^{\mathrm{R}} = \begin{cases} 0, & f_i^{\mathrm{R}} = 0, \\ 1, & f_i^{\mathrm{R}} \geq f_{\min,i}, \end{cases} \qquad \forall i \in \mathcal{I}_{\mathrm{thr}} \tag{28}$$

The complete MINLP initialization is then

$$\boldsymbol{\theta}^{(0)} = \boldsymbol{\theta}^{\mathrm{R}}, \qquad \boldsymbol{f}^{(0)} = \boldsymbol{f}^{\mathrm{R}}, \qquad \boldsymbol{z}^{(0)} = \boldsymbol{z}^{\mathrm{R}} \tag{29}$$

The values in Eq. (29) are supplied to the selected general-purpose MINLP solver as a complete MIB-feasible starting point. Once validated by the solver, this point provides an initial feasible incumbent with objective value J_{R}. The warm start supplies initial values only; it does not fix any decision variable, impose additional constraints, or restrict the final activation pattern. The solver then handles the remaining optimization of the unchanged MINLP in Eq. (10).
The acceptance test uses bounds already available from the preceding analysis and the selected optimizer. Specifically, J_{floor} is combined with any globally valid objective bound reported by the MINLP solver. The tighter available lower bound is defined as

$$J_{\mathrm{LB}} = \begin{cases} \max\left(J_{\mathrm{floor}}, J_{\mathrm{LB}}^{\mathrm{solver}}\right), & \text{if a valid solver bound is reported,} \\ J_{\mathrm{floor}}, & \text{otherwise} \end{cases} \tag{30}$$

where $J_{\mathrm{LB}}^{\mathrm{solver}}$ denotes the globally valid lower bound reported directly by the selected MINLP solver. The proposed allocation procedure does not otherwise modify how this bound or the feasible incumbent is generated. Let J_{inc} denote the objective value of the best MIB-feasible incumbent currently available to the solver. Because

$$J_{\mathrm{LB}} \leq J_{\mathrm{MIB}}^{*} \leq J_{\mathrm{inc}} \tag{31}$$

the incumbent's unknown suboptimality satisfies

$$0 \leq J_{\mathrm{inc}} - J_{\mathrm{MIB}}^{*} \leq J_{\mathrm{inc}} - J_{\mathrm{LB}} \tag{32}$$

The allocation procedure accepts the best feasible solution reported by the MINLP solver once

$$J_{\mathrm{inc}} - J_{\mathrm{LB}} \leq \Delta J_{\max} \tag{33}$$

Satisfying Eq. (33) guarantees that the accepted solution is within $\Delta J_{\max}$ of the globally optimal MIB-constrained objective value. If a prescribed solution-time limit is reached first, the best MIB-feasible solution reported by the solver is returned. Together, the preceding stages reserve the MINLP for allocation updates that cannot be resolved by the zero-fire, continuous-NLP, or repair tests while also providing a feasible solution to initialize the solver.

4. Numerical Results

All numerical studies use the same spacecraft illustrated in Figure 2, consisting of a rigid rectangular hub and two identical 4-DOF arms attached to opposing hub faces at $[\pm 0.79, 0, 0]^T$ m. Each arm comprises two proximal revolute joints, a 1 m rigid boom, and two distal revolute joints. The thruster force-application point is located 0.1 m from the final joint. When active, each thruster produces a fixed force of 0.25 N along its local $[-1, 0, 0]^T$ axis. The component mass and inertia properties are listed in Table 1, and the local joint spin axes are listed in Table 2. The arm-component properties in Table 1 apply separately to each of the two arms and the inertia matrices are evaluated about the corresponding component CoM and expressed along its local principal axes.
All allocation calls used an update interval of $\Delta t_{\mathrm{alloc}} = 10$ s. The eight joint angles were constrained by

$$-\pi \leq \theta_j \leq \pi, \qquad j = 1, \ldots, 8 \tag{34}$$

The allocation objective used the weighting parameters

$$[W_u] = [I_{6\times 6}], \qquad \boldsymbol{W}_f = 10^{-4}\mathbf{1}_{N_T}, \tag{35}$$

Table 1: Spacecraft mass and inertia properties.

Body	**Mass [kg]**	**Dimensions [m]**	**Inertia [kg·m^2]**
Hub	980	$1.58 \times 1.38 \times 1.04$	diag(243.929, 292.667, 359.007)
Joints 1–2 and boom	8	1	diag(0.040, 0.687, 0.687)
Joints 3–4 and thruster	2	0.1	diag(0.0025, 0.021, 0.021)

Table 2: Local joint spin axes for each manipulator arm

Joint 1	**Joint 2**	**Joint 3**	**Joint 4**
$[1, 0, 0]^T$	$[0, 1, 0]^T$	$[0, 0, 1]^T$	$[0, 1, 0]^T$

where $[I_{6\times6}]$ is the six-dimensional identity matrix and $\mathbf{1}_{N_T}$ is an N_T-element vector of ones.
A common set of 15,000 desired control wrenches was generated using six-dimensional Latin hypercube sampling. Each force component was sampled over $[-0.005, 0.005]$ N, and each torque component was sampled over $[-0.005, 0.005]$ N·m. The same wrench set was evaluated using the continuous NLP and five MIB-constrained MINLP families with minimum firing durations

$$t_{\min} \in \{10,\ 20,\ 50,\ 100,\ 200\}\ \text{ms} \tag{36}$$

With $\Delta t_{\text{alloc}} = 10$ s and $f_{\max,i} = 0.25$ N, Eq. (8) maps these minimum firing durations to the equivalent minimum nonzero force commands

$$f_{\min,i} \in \left\{2.5 \times 10^{-4},\ 5.0 \times 10^{-4},\ 1.25 \times 10^{-3},\ 2.5 \times 10^{-3},\ 5.0 \times 10^{-3}\right\}\ \text{N} \tag{37}$$

respectively. The continuous-NLP baseline does not impose a minimum nonzero force command.
Using a common wrench set enables paired comparisons between the continuous baseline and each MIB family. The sampled wrench magnitudes were deliberately kept small to increase the frequency of sub-MIB commands. Consequently, this ensemble is intended to stress the MIB-aware allocation methods and is not representative of the wrench history expected from the closed-loop controller. None of the 75,000 MIB-constrained cases satisfied $J_{\text{zero}} \le J_{\text{floor}}$. The allocation studies therefore evaluate the continuous-NLP screening, repair, and MINLP stages but do not provide an empirical evaluation of the no-fire screen.
All NLP and MINLP cases were solved using the global nonlinear optimization capabilities of Gurobi Optimizer version 13.0.2*, with four threads assigned to each solve. The global algorithm uses spatial branch-and-bound and dynamic outer approximations to maintain valid objective bounds. The Gurobi absolute-gap parameter `MIPGapAbs` was set to the value of $\Delta J_{\max}$ defined below. Each solve was terminated when either the absolute objective-gap condition in Eq. (33) was satisfied or 15 s had elapsed following the completion of presolve. At termination, the best feasible incumbent and associated globally valid objective bound were retained.
The reported per-case solution time is Gurobi's `Model.Runtime` value. This measurement includes presolve and the subsequent optimization process. Consequently, the reported runtime can exceed 15 s when presolve requires nonnegligible time. It excludes model construction, which occurs before the call to `optimize()`, and Python-side result extraction and storage, which occur after the runtime is recorded.
A common absolute objective-gap tolerance of $\Delta J_{\max} = 10^{-6}$ was used for all allocation studies. To scale this value against the objective range relevant to the MIB-constrained problem, define

$$B_{m,q} \triangleq J_{\text{zero},m} - J_{\text{floor},m,q} \tag{38}$$

where m denotes the Monte Carlo trial and q denotes the MIB family. The quantity $B_{m,q}$ is the conservative objective range between the zero-fire cost and the theoretical nonzero MIB lower bound. Its fifth percentile over all sampled trials and MIB families is

$$B_{5\%} \triangleq \text{quantile}_{0.05}\left(\{B_{m,q}\}\right) \tag{39}$$

The tolerance was selected to satisfy

$$\Delta J_{\max} \le 0.05 B_{5\%} \tag{40}$$

Because $B_{m,q} \ge B_{5\%}$ for 95% of the pooled cases, any solution satisfying Eq. (33) has an unresolved objective improvement no greater than 5% of $B_{m,q}$ for at least 95% of the sampled MIB cases.

*`https://docs.gurobi.com/projects/optimizer/en/current/`

The 15 s post-presolve optimization window was selected using a separate preliminary study of 1,000 desired wrenches across representative 10, 100, and 200 ms MIB families. Extending the post-presolve optimization window from 15 to 25 s reduced the incumbent objective by less than $0.02B_{m,q}$ for 95% of the preliminary cases. Thus, the additional solution time yielded limited improvement relative to each case's conservative objective-reduction range while substantially increasing the computational cost of the Monte Carlo study.

4.1 Direct MINLP Results

The computational performance of direct MIB-aware allocation is first evaluated by solving the MINLP for every requested wrench. Figures 4 and 5 compare its termination behavior with that of the continuous NLP. The termination-time cumulative distribution functions (CDFs) in Fig. 4 show a clear separation between the two formulations. The prescribed absolute optimality-gap criterion is reached in 99.26% of the continuous-NLP cases but only 62.18–66.15% of the MINLP cases, with approximately one-third of each MINLP family instead reaching the runtime limit. The terminal increases in the CDFs correspond to these time-limit terminations rather than satisfaction of the gap criterion. The MINLP distributions are otherwise broadly similar, although the 200 ms family produces fewer rapid terminations.
Figure 5 considers only the time-limited cases and reports the remaining gap ratio $(J_{\text{inc}} - J_{\text{LB}})/\Delta J_{\max}$, for which a value of one corresponds to the prescribed acceptance threshold. The continuous-NLP distribution remains concentrated near this threshold, whereas the MINLP families have median ratios between 2.22 and 2.90 and substantially broader upper tails. Nevertheless, every NLP and MINLP solve returns a feasible incumbent before termination. The principal computational difficulty is therefore reducing and certifying the remaining optimality gap within the available runtime, rather than locating an MIB-feasible allocation.

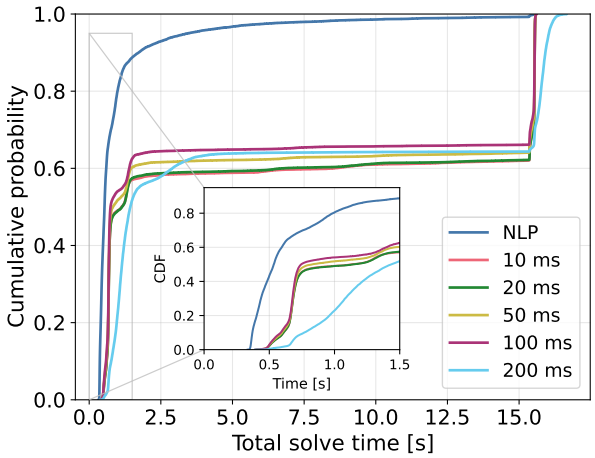


Figure 4: Termination-time CDFs for the continuous NLP and direct-MINLP MIB families

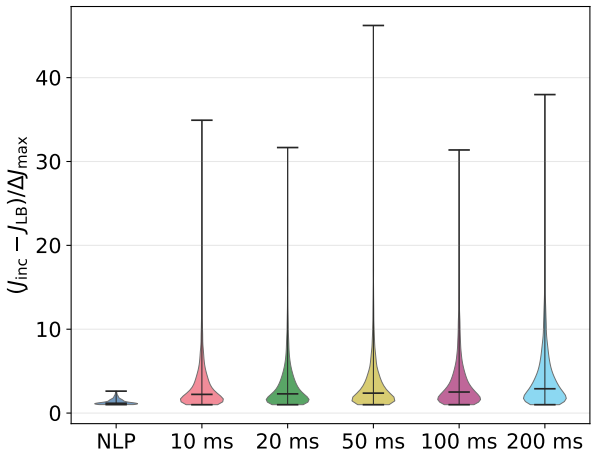


Figure 5: Remaining optimality-gap ratios for cases terminated by the runtime limit

Table 3 reports median and 95th-percentile (P95) runtime, node-count, and terminal-gap statistics. These results show that the effect of minimum firing duration on the global search is not strictly monotonic. The 10 and 20 ms families require the largest median runtimes and substantially greater branch-and-bound exploration, whereas the 50 and 100 ms families generally terminate more quickly. Conversely, the remaining gap ratios tend to increase with minimum firing duration. Within this data set, shorter minimum firing durations therefore produce larger search trees, while longer durations tend to produce weaker final optimality certificates. Every direct-MINLP family is nonetheless substantially more computationally demanding than the continuous NLP, motivating the staged methods introduced in Section 3.3.
The allocation-quality comparison is restricted to continuous-NLP solutions that violate the corresponding MIB requirement. For the rounded baseline, each sub-MIB firing duration is independently replaced by either zero or the minimum firing duration, whichever is closer, while the remaining firing commands and NLP-selected joint configuration are retained. The resulting command is MIB feasible but is not reoptimized after rounding.
For an allocation method $a \in \{\text{round}, \text{MINLP}\}$, let $\boldsymbol{f}^a$ denote its executable firing-command vector, $\boldsymbol{u}^a_{\text{ctl}}$ the corresponding executable control wrench, and J^a the objective reconstructed from that allocation. Its wrench-tracking error is defined as

$$e^a \triangleq \left\| \boldsymbol{u} - \boldsymbol{u}^a_{\text{ctl}} \right\| \tag{41}$$

Table 3: Continuous-NLP and direct-MINLP termination and computational performance

Criterion	Continuous NLP	MINLP MIB family				
		10 ms	20 ms	50 ms	100 ms	200 ms
Gap criterion reached [%]	99.26	62.18	62.27	64.17	66.15	64.43
Time limit reached [%]	0.74	37.82	37.73	35.83	33.85	35.57
Median runtime [s]	0.101	0.724	0.736	0.346	0.244	0.457
P95 runtime [s]	3.05	15	15	15	15	15
Median node count	1	1,180	1,180	487	135	27.5
P95 node count	13,400	59,000	56,000	53,000	50,000	34,800
Median gap ratio at time limit	1.16	2.22	2.29	2.37	2.51	2.90
P95 gap ratio at time limit	2.02	8.43	8.53	8.93	9.52	10.9

The paired performance measures reported in Table 4 are

$$R_e \triangleq \frac{e^{\text{MINLP}}}{e^{\text{round}}}, \qquad R_J \triangleq \frac{J^{\text{MINLP}}}{J^{\text{round}}}, \qquad \Delta I \triangleq \Delta t_{\text{alloc}} \mathbf{1}_{N_T}^T \left(\boldsymbol{f}^{\text{MINLP}} - \boldsymbol{f}^{\text{round}} \right) \tag{42}$$

Ratios below one favor the direct MINLP, while $\Delta I > 0$ indicates that it uses more impulse.

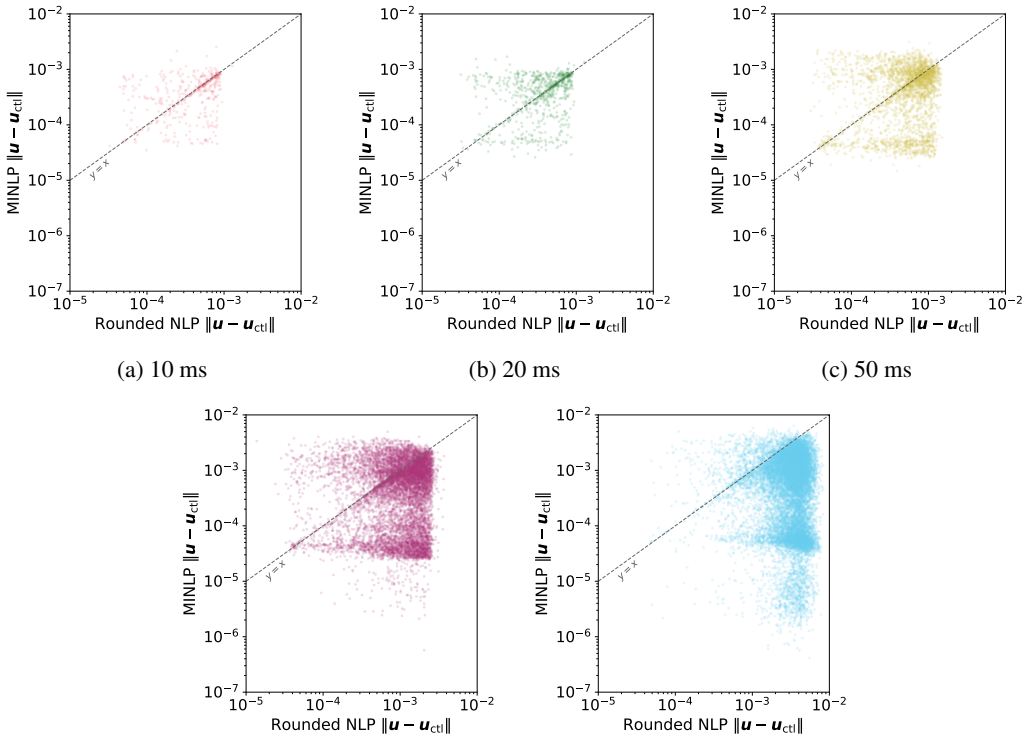
Figure 6 compares the tracking errors for the violating cases. Points below the $y = x$ line favor the direct MINLP, whereas points above it favor rounding. Because the panels contain only MIB-violating allocations, their increasing density also reflects the growing frequency of MIB violations. The sampled wrench magnitudes were deliberately kept small to increase the occurrence of sub-MIB commands, so the frequencies in Table 4 characterize this MIB-stress-test data set rather than expected closed-loop rates. Even under these conditions, the violation frequency ranges from only 2.97% for the 10 ms family to 99.22% for the 200 ms family.

The progression across the panels shows that the benefit of MIB-aware optimization increases with minimum firing duration. The 10 and 20 ms point clouds remain distributed around the equality line, the 50 ms family exhibits an intermediate transition, and the 100 and 200 ms clouds lie predominantly below the line. Table 4 quantifies this behavior: the direct MINLP produces lower tracking error in approximately half of the shortest-MIB cases but in 94.22% of the 200 ms cases, for which the median error ratio is 0.196. The complete allocation objective follows the same general trend.

The improvement is not uniform, as points remain above the equality line in every panel. The impulse statistics also reveal an accuracy–effort tradeoff: median impulse changes remain small through the 100 ms family, whereas the 200 ms MINLP uses more impulse in 79.30% of the violating cases. Moreover, P95 objective ratios greater than one show that some direct solves terminate before improving upon the available rounded solution, even though that solution is MIB feasible. This upper-tail behavior illustrates the benefit of using an MIB-feasible repaired allocation as the initial incumbent for the warm-started MINLP described in Section 3.3.4.

Table 4: Rounded-NLP and direct-MINLP performance for MIB-violating allocations.

Metric	MIB family				
	10 ms	20 ms	50 ms	100 ms	200 ms
MIB-violating NLP cases	445 (2.97%)	731 (4.87%)	2,798 (18.65%)	9,184 (61.23%)	14,883 (99.22%)
MINLP lower tracking error [%]	47.42	47.06	60.61	74.83	94.22
Median error ratio R_e	1.00	1.00	0.885	0.534	0.196
P95 error ratio R_e	5.02	5.57	6.40	4.55	1.06
MINLP greater impulse [%]	51.24	41.04	41.78	50.58	79.30
Median impulse change ΔI [N·s]	2.72×10^{-6}	-6.03×10^{-5}	-4.66×10^{-4}	1.27×10^{-5}	1.76×10^{-2}
P95 impulse change ΔI [N·s]	8.43×10^{-3}	6.56×10^{-3}	1.67×10^{-2}	3.83×10^{-2}	0.105
MINLP lower objective [%]	52.81	50.07	60.44	74.09	93.91
Median objective ratio R_J	1.00	1.00	0.949	0.628	0.162
P95 objective ratio R_J	2.18	2.05	3.71	3.99	1.12



(a) 10 ms (b) 20 ms (c) 50 ms

(d) 100 ms (e) 200 ms

Figure 6: Rounded-NLP and direct-MINLP tracking errors for MIB-violating allocations

4.2 Staged Allocation Results

The staged MIB-aware allocation procedure from Section 3.3 is evaluated by cumulatively adding each screening, repair, and warm-start operation to the direct MINLP baseline. Figure 7 compares the total-runtime distributions of four implementations: direct MINLP allocation, continuous-NLP screening with a cold-started MINLP fallback, screening and repair with a cold-started MINLP fallback, and the complete procedure with a warm-started MINLP fallback. The effect of each added operation is isolated below by comparing adjacent curves. For each implementation, the total runtime includes the recorded runtime of every invoked optimization stage and, where applicable, the repair evaluation time. The negligible computational cost of the logical screening checks is not included.

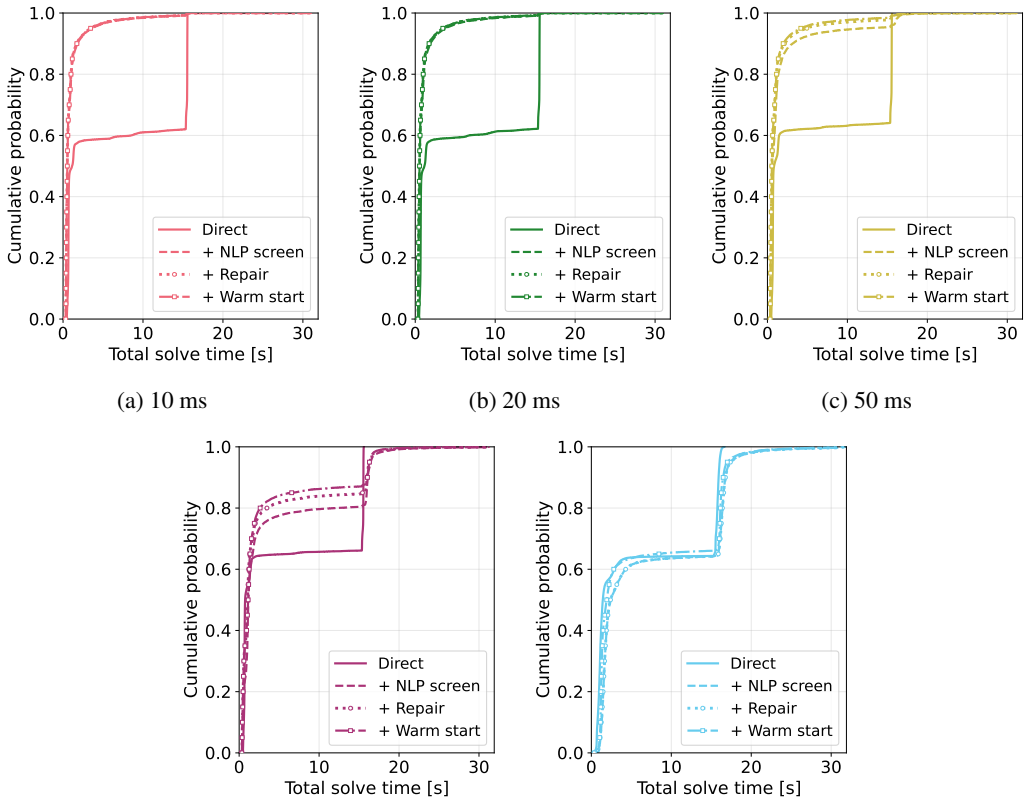
The no-fire screen is applied within each staged allocation call, but none of the sampled wrenches satisfy its acceptance condition. Consequently, the analytical benefit established in Section 3.3 cannot be assessed empirically using this Monte Carlo data set, and the numerical comparisons begin with continuous-NLP screening.

4.2.1 Continuous-NLP Screening

The computational benefit of continuous-NLP screening is governed by how frequently the NLP solution satisfies the MIB requirement. The Monte Carlo data set was deliberately constructed using small desired control wrenches to increase the occurrence of sub-MIB commands. The resulting acceptance rates therefore characterize an MIB-stressing data set rather than the screening performance expected for every operational command history. As shown in Fig. 7 and Table 5, the screen is highly effective for the shorter minimum firing durations. More than 95% of the 10 and 20 ms cases are accepted without invoking the MINLP, and the screened procedure is faster than direct MINLP allocation in more than 84% of the trials. A substantial runtime benefit also remains for the 50 ms family.

As the minimum firing duration increases, fewer NLP solutions pass the screen and the cost of performing the preliminary NLP solve is incurred before most cold-started MINLP calls. The 100 ms family illustrates this transition. Figure 7d shows that screening increases the cumulative fraction of cases completed over part of the intermediate-runtime range, but the added NLP solve shifts the fallback cases farther into the upper tail. Consequently, the screened procedure is faster in only 32.25% of the 100 ms cases, and both its median and P95 runtimes exceed those of direct MINLP allocation. At 200 ms, only 0.78% of the NLP solutions pass the screen, so the added NLP solve increases the runtime for nearly every allocation call in this data set. This result does not imply that screening is generally ineffective for longer MIBs. During closed-loop operation, larger requested control wrenches can produce firing durations that naturally exceed the MIB, potentially increasing the screening acceptance rate.

The runtime statistics in Table 5 include all 15,000 trials, whereas the solution-quality statistics include only cases



(a) 10 ms (b) 20 ms (c) 50 ms

(d) 100 ms (e) 200 ms

Figure 7: Total-runtime CDFs as NLP screening, repair, and warm starting are added to direct MINLP allocation

accepted by the NLP screen. Over this accepted subset, the NLP solution has lower tracking error than the matched direct-MINLP incumbent in 61–70% of the cases, and the median tracking-error and objective ratios remain below one for every MIB family. The accepted NLP objective is also no greater than the direct-MINLP objective in at least 71% of the cases.

These differences arise from finite-time solver performance rather than from a superior continuous formulation. If a globally optimal continuous-NLP solution satisfies the MIB requirement, it is feasible for the MINLP and therefore also attains the globally optimal MINLP objective value. The observed advantage therefore indicates that the easier continuous problem often returns a higher-quality MIB-feasible solution than the direct MINLP identifies within its termination limits. This advantage is not uniform, however: the P95 tracking-error and objective ratios exceed one for every MIB family, showing that the direct MINLP incumbent performs better in the upper tail of the accepted-case comparisons. The 200 ms quality statistics should also be interpreted in view of the small accepted subset, which contains only 117 cases.

Table 5: Continuous-NLP screening performance and accepted-solution quality.

Metric	**MIB family**				
	10 ms	**20 ms**	**50 ms**	**100 ms**	**200 ms**
NLP screen accepted [%]	97.03	95.13	81.35	38.77	0.78
Screened pipeline faster [%]	86.31	84.93	71.13	32.25	0.58
Median runtime [s], direct/screened	1.13 / 0.542	1.14 / 0.546	0.830 / 0.620	0.745 / 1.16	1.44 / 2.42
P95 runtime [s], direct/screened	15.6 / 3.59	15.6 / 3.80	15.6 / 11.8	15.6 / 16.5	16.0 / 17.4
NLP lower tracking error [%]	70.40	68.87	65.09	61.43	68.38
Median error ratio, NLP/direct	0.540	0.534	0.533	0.458	0.243
P95 error ratio, NLP/direct	2.14	2.40	2.48	3.45	7.06
NLP objective no greater [%]	71.98	71.89	71.49	71.91	81.20
Median objective ratio, NLP/direct	0.849	0.853	0.844	0.809	0.724
P95 objective ratio, NLP/direct	1.31	1.33	1.35	1.42	1.22

4.2.2 MIB-Feasible Repair

The repair stage determines whether the lowest-cost MIB-feasible point in the finite repair set satisfies the prescribed objective-gap acceptance criterion. Figure 7 and Table 6 compare continuous-NLP screening with a cold-started MINLP fallback against the same procedure augmented by repair. Evaluating the repair candidates requires a median of approximately 3.6×10^{-4} s for every MIB family, which is negligible relative to the NLP and MINLP runtimes.

The computational benefit therefore depends primarily on how frequently repair avoids the subsequent MINLP. For the 10 and 20 ms families, the adjacent screening and repair curves in Fig. 7 nearly overlap because the preceding NLP screen already accepts most allocations. At the opposite extreme, repair accepts only 4.64% of the 200 ms screen failures, producing little change in the upper-tail runtime. The greatest benefits occur for the intermediate MIB families. Adding repair reduces the median runtime by approximately 12% for both the 50 and 100 ms families, while the 50 ms P95 runtime decreases from 11.8 to 4.52 s, a reduction of approximately 62%. The cumulative fraction of cases avoiding the MINLP consequently reaches 94.31% for the 50 ms family and 59.62% for the 100 ms family, up from 81.35% and 38.77%, respectively, with screening alone.

The allocation-quality statistics in Table 6 include only cases for which the repaired solution is accepted and compare that solution with the cold-started MINLP incumbent that would otherwise have been returned. The median tracking-error ratios range from 0.999 to 1.05, while every median objective ratio rounds to 1.00. Thus, the typical accepted repair provides allocation quality comparable to the substantially more expensive cold-started MINLP solve.

The upper-tail ratios show greater variation. The P95 tracking-error ratios range from 5.60 to 17.7, whereas the P95 objective ratios remain between 1.51 and 1.86. These quantities should be interpreted in view of the repair criterion. Repair is accepted using an absolute bound on the complete allocation objective, not a relative bound on tracking error alone. The objective also permits a tradeoff between wrench accuracy and thruster effort, and ratios can become large when the corresponding cold-started MINLP error or objective is very small. Consequently, the upper-tail ratios do not violate the prescribed repair guarantee, but they show that an accepted repair can produce a larger tracking error than the cold-started MINLP incumbent in some cases.

Table 6: MIB-feasible repair acceptance, runtime, and allocation quality.

Metric	**MIB family**				
	10 ms	**20 ms**	**50 ms**	**100 ms**	**200 ms**
Repair accepted [% of screen failures]	91.24	93.98	69.51	34.05	4.64
Cumulative MINLP avoidance [%]	99.74	99.71	94.31	59.62	5.38
Median repair time [s]	3.64×10^{-4}	3.60×10^{-4}	3.58×10^{-4}	3.67×10^{-4}	3.68×10^{-4}
P95 repair time [s]	3.77×10^{-4}	3.70×10^{-4}	3.70×10^{-4}	3.83×10^{-4}	3.79×10^{-4}
Median runtime [s], without/with repair	0.542 / 0.535	0.546 / 0.535	0.620 / 0.548	1.16 / 1.02	2.42 / 2.30
P95 runtime [s], without/with repair	3.59 / 3.47	3.80 / 3.49	11.8 / 4.52	16.5 / 16.2	17.4 / 17.3
Median error ratio, repair/cold MINLP	0.999	0.999	1.00	1.02	1.05
P95 error ratio, repair/cold MINLP	5.60	5.67	12.2	17.7	16.4
Median objective ratio, repair/cold MINLP	1.00	1.00	1.00	1.00	1.00
P95 objective ratio, repair/cold MINLP	1.64	1.51	1.77	1.86	1.63

4.2.3 Warm-Start Results

The final stage uses the repaired MIB-feasible solution to initialize the MINLP. Figure 7 shows the effect of warm starting on the total runtime of the complete allocation pipeline, whereas Table 7 isolates the MINLP-stage performance for cases not accepted by either screening or repair. Because only 39 and 44 cases reach the MINLP for the 10 and 20 ms families, respectively, warm starting has little visible effect on their all-case runtime distributions. Nevertheless, among those remaining cases, warm starting reduces the median MINLP runtime by 23.7% and 32.3%, respectively. The 200 ms family similarly experiences a 41.4% reduction in median MINLP runtime.

More generally, warm starting improves finite-time convergence reliability rather than uniformly reducing runtime. The warm-started MINLP satisfies the absolute-gap criterion more frequently and requires fewer median branch-and-bound nodes for every MIB family. For the 50 and 100 ms families, however, the median runtime increases despite the higher gap-criterion satisfaction rate and lower median node count. Their P95 node counts also increase slightly, demonstrating that node count alone does not characterize the computational effort associated with the altered branch-and-bound search. Among the 50–200 ms cases that still reach the cutoff, both the median and P95 remaining-gap ratios are lower with warm starting. The corresponding statistics for the 10 and 20 ms families are not interpreted because only one warm-started solve reaches the cutoff in each family. For the 50–200 ms families, the final warm-started incumbent has an objective

value no greater than the matched cold-started incumbent in 68.11–78.36% of cases, including objectives considered equivalent using absolute and relative tolerances of 10^{-12} and 10^{-9}, respectively. These results indicate that the repaired solution provides useful initialization information, particularly for the larger sets of allocations that proceed to the MINLP stage.

Table 7: Cold- and warm-started MINLP performance for cases not accepted by screening or repair.

Metric	**MIB family**				
	10 ms	**20 ms**	**50 ms**	**100 ms**	**200 ms**
Cases requiring MINLP N [% of all trials]	39 [0.26]	44 [0.29]	853 [5.69]	6,057 [40.38]	14,193 [94.62]
Gap criterion reached [%] cold/warm	89.74 / 97.44	88.64 / 97.73	78.66 / 85.23	67.54 / 69.69	63.71 / 64.77
Median runtime [s] cold/warm	0.675 / 0.515	0.690 / 0.467	0.691 / 0.780	0.728 / 0.833	1.46 / 0.855
P95 runtime [s] cold/warm	15.5 / 0.589	15.5 / 0.659	15.5 / 15.7	15.5 / 15.7	16.0 / 15.7
Median node count cold/warm	7 / 1	7 / 1	7 / 6	46 / 7	47 / 27
P95 node count cold/warm	54,652 / 1	54,465 / 6	54,272 / 56,325	50,387 / 51,081	35,162 / 43,498
Median timeout gap ratio cold/warm	1.63 / 1.81[†]	1.21 / 1.78[†]	1.35 / 1.10	2.09 / 1.64	2.88 / 2.70
P95 timeout gap ratio cold/warm	5.99 / 1.81[†]	2.04 / 1.78[†]	2.77 / 1.64	5.97 / 3.63	10.7 / 8.91
Warm-start objective no greater [%]	23.08	36.36	68.11	73.82	78.36

[†] Based on one warm-started runtime-cutoff case.

Overall, the staged procedure is most computationally effective when screening or repair accepts a sufficient fraction of the allocations before the MINLP is invoked. These early stages provide substantial runtime reductions for the shorter MIB families, whereas their added computational cost becomes more influential as the fraction of cases requiring the MINLP increases. Warm starting improves the finite-time convergence of the remaining MINLP solves but does not uniformly reduce their runtime or incumbent objective. The staged procedure therefore trades a larger worst-case runtime, resulting from the preliminary NLP and repair stages, for improved typical performance when those stages successfully avoid or accelerate the MINLP.

4.3 Closed-Loop Performance

To determine whether the improved allocation accuracy provided by MIB-aware allocation translates into improved closed-loop performance, two simulations are compared using a 100 ms minimum firing duration. The baseline solves the continuous NLP and rounds each sub-MIB command to the nearer admissible value of zero or the minimum firing command, whereas the proposed case uses the complete staged MIB-aware allocation pipeline. The spacecraft dynamics, initial conditions, reference trajectory, feedback-control architecture, and lower-level joint and thruster models are identical in both simulations; only the treatment of the MIB during allocation is changed. Both cases use the spacecraft model defined in Tables 1 and 2 as well as the proximity-inspection scenario and control gains reported in the *Simulation Setup and Scenario Definition* section and Table 3, respectively, of Ref. 24. The desired control wrench is generated using the proximity-operations control architecture of Refs. 24, 25 and the joint motion follows the 9.5 s cubic reference profile introduced in Ref. 25. These guidance and feedback-control elements remain unchanged in the present study. Both simulations use the same 10 ms pulse-resolution increment when converting the allocated firing durations into executable commands as the prior studies.

Relative to the prior closed-loop studies, the present simulations increase the minimum firing duration from 10 to 100 ms and use the global nonlinear optimization capabilities of Gurobi Optimizer for both the continuous NLP and MINLP, rather than the multistart sequential least-squares programming implementation in SciPy. The same 15 s post-presolve runtime cutoff used in the preceding allocation studies was applied in both simulations. For allocation step k, the conservative cost-reduction range is

$$B_k = J_{\mathrm{zero},k} - J_{\mathrm{floor},k}. \tag{43}$$

The value of $\Delta J_{\max,k}$ was selected using

$$\Delta J_{\max,k} = \begin{cases} \Delta J_{\text{base}}, & \epsilon_B B_k \leq \Delta J_{\text{base}}, \\ \epsilon_B B_k, & \Delta J_{\text{base}} < \epsilon_B B_k < \Delta J_{\text{cap}}, \\ \Delta J_{\text{cap}}, & \epsilon_B B_k \geq \Delta J_{\text{cap}} \end{cases} \tag{44}$$

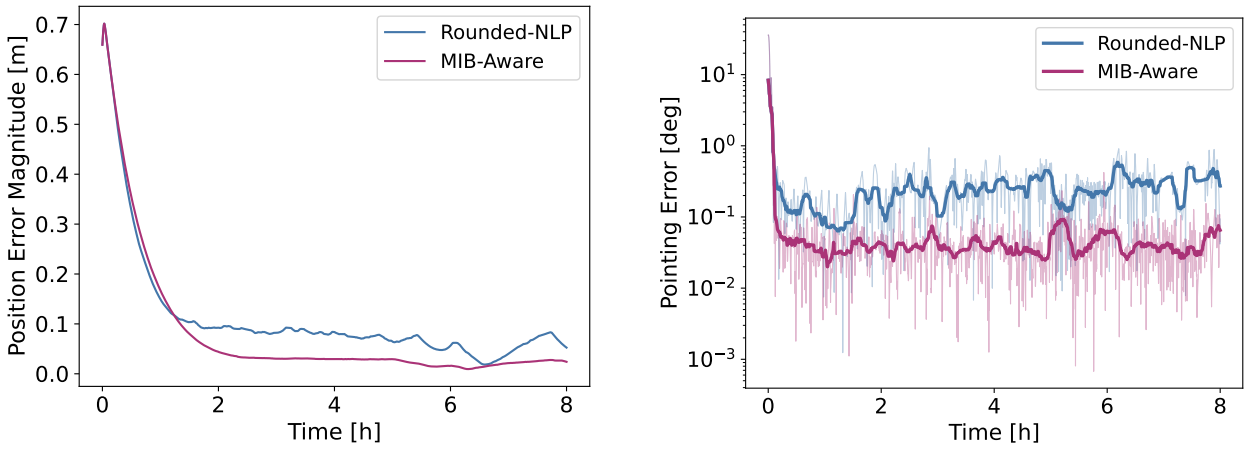
Here, $\epsilon_B = 0.05$ limits the unresolved objective improvement to 5% of the case-specific conservative cost-reduction range when neither bound is active. The lower limit, $\Delta J_{\text{base}} = 1.0 \times 10^{-6}$, is the fixed tolerance established for the small-wrench allocation study through Eqs. (38)–(40). The upper limit is selected relative to the objective variation produced by representative thruster-force uncertainty.

A separate set of 15,000 nominal allocations was generated with desired force components sampled from $\mathcal{U}[-0.025, 0.025]$ N and desired torque components sampled from $\mathcal{U}[-0.25, 0.25]$ N·m where $\mathcal{U}[a, b]$ denotes a continuous uniform distribution over $[a, b]$. These bounds were selected to encompass the largest absolute force and torque components requested during the closed-loop inspection simulation, thereby spanning the relevant control-wrench envelope rather than the deliberately small commands used in the preceding allocation study. For each allocation, the optimized joint configuration and commanded firing vector were held fixed while the delivered force of each thruster was independently perturbed by ±5%, based on reported performance data for broadly comparable small chemical thrusters which indicates delivered-thrust variations of approximately 5-11% [30–32]. The allocation objective was recomputed for all four uncertainty combinations, and the largest increase relative to the nominal objective was retained as $\Delta J^{5\%}_{\text{unc},m}$.

$$\Delta J_{\text{cap}} = \text{quantile}_{0.025}\left(\left\{\Delta J^{5\%}_{\text{unc},m}\right\}_{m=1}^{15{,}000}\right) \approx 2.25 \times 10^{-5}. \tag{45}$$

Thus, for approximately 97.5% of the sampled allocations, the modeled 5% thrust variation produces an objective increase at least as large as ΔJ_{cap}. Capping the solver tolerance at this value prevents the permitted unresolved optimization improvement from exceeding a conservative lower-tail estimate of the objective variation associated with implementation uncertainty.

Figure 8 shows the closed-loop position and pointing errors throughout the 8 h simulation. The two trajectories initially coincide because the continuous-NLP commands are naturally MIB feasible and therefore require no rounding. They begin to diverge after approximately 5 min, once the initial attitude transient has largely been removed and the smaller requested commands become more sensitive to the treatment of the MIB. Both errors settle into their representative operating ranges within the first 3 h. Table 8 therefore reports the root-mean-squared (RMS), P95, and maximum tracking errors over the final 5 h of the simulation. Figure 8 and Table 8 show that the MIB-aware pipeline reduces the RMS position and pointing errors by 63.3% and 82.8%, respectively, relative to rounding.



(a) Position error

(b) Pointing error

Figure 8: Closed-loop position and pointing errors; thick pointing curves show 10-min rolling medians

The improved closed-loop tracking is consistent with the executable allocation errors reported in Table 8. These errors are evaluated from the final MIB-feasible command returned by each allocation approach, after rounding for the continuous-NLP baseline, but before lower-level joint and pulse-execution effects. Over the full trajectory, the MIB-aware pipeline reduces the RMS force and torque allocation errors by 25.8% and 65.3%, respectively. The corresponding P95 reductions are 32.5% and 64.9%.

The improved allocation and tracking accuracy come at the cost of greater applied impulse, as shown in Fig. 9 and Table 8. The MIB-aware pipeline commands 85.66 N·s and applies 85.59 N·s, such that 99.91% of the commanded impulse is realized. The remaining difference results from the common 10 ms pulse resolution rather than from sub-MIB commands. By comparison, the continuous NLP commands 109.2 N·s, but endpoint rounding reduces the applied impulse to 44.26 N·s, or 40.53% of the commanded value. The MIB-aware simulation therefore applies 93.4% more impulse than the rounded simulation despite producing a smaller nominal command.

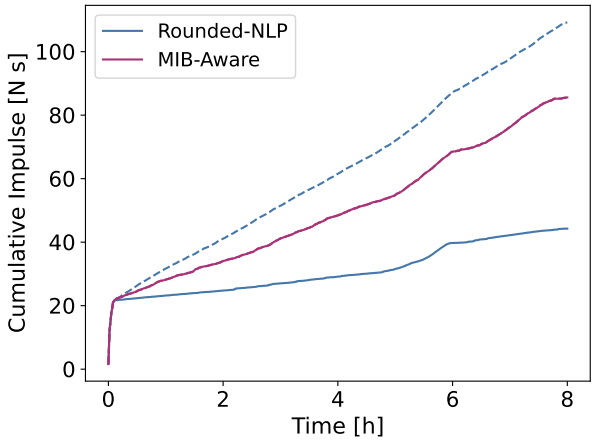


Figure 9: Cumulative commanded (dashed) and applied (solid) thruster impulse.

This improvement also requires greater allocation runtime. The median runtime increases from 0.419 s for the rounded continuous NLP to 0.863 s for the MIB-aware pipeline, while the P95 runtime increases from 0.503 to 15.84 s. Table 9 shows that the staged procedure nevertheless bypasses the MINLP for 43.7% of the allocation calls: 2.7% terminate at the no-fire screen, 1.2% return the MIB-feasible continuous-NLP solution, and 39.8% accept the repaired solution. The remaining 56.3% invoke the MINLP.

Only 35 continuous-NLP solutions are accepted directly, all within the first 36 allocation updates when the requested force and torque components are largest. As the tracking errors decrease, the requested control wrenches become smaller and the continuous solution more frequently contains sub-MIB commands. The resulting stage distribution is therefore specific to the selected 100 ms MIB and fixed-thrust level. A shorter MIB or lower fixed-thrust level would reduce the minimum nonzero command and could increase the continuous-NLP screening rate, although changing the thrust level would also alter the available control authority.

Table 8: Closed-loop tracking, allocation, impulse, and runtime performance.

Metric	**Rounded NLP**	**MIB-aware**
Tracking errors for $t \geq 3$ hr		
Position error [m], RMS/P95/max	0.06719 / 0.08823 / 0.09258	0.02466 / 0.03064 / 0.03109
Pointing error [deg], RMS/P95/max	0.3276 / 0.5921 / 0.9112	0.05639 / 0.1013 / 0.4221
Executable allocation errors over the full trajectory		
Force error [N], RMS/P95	1.040×10^{-3} / 1.763×10^{-3}	7.715×10^{-4} / 1.189×10^{-3}
Torque error [N·m], RMS/P95	1.322×10^{-3} / 2.336×10^{-3}	4.585×10^{-4} / 8.208×10^{-4}
Full-trajectory impulse and allocation runtime		
Commanded impulse [N·s]	109.2	85.66
Applied impulse [N·s]	44.26	85.59
Applied/commanded impulse [%]	40.53	99.91
Allocation runtime [s], median/P95	0.4193 / 0.5034	0.8630 / 15.84

Table 9: MIB-aware pipeline utilization over the closed-loop trajectory.

Allocation path	Calls	Fraction [%]
No-fire screen	77	2.7
Continuous-NLP screen	35	1.2
MIB-feasible repair	1,147	39.8
MINLP invoked	1,622	56.3
Total	2,881	100.0

5. Conclusion

This paper developed an MIB-aware formulation of the coupled joint-and-thruster allocation problem for spacecraft with reconfigurable, arm-mounted fixed-thrust thrusters. The direct MINLP formulation jointly selects the manipulator configuration, thruster activation pattern, and equivalent force commands, allowing the thruster geometry to adapt to the MIB rather than modifying an incompatible continuous allocation after optimization. A staged allocation pipeline was also introduced to reduce the computational burden of this formulation through no-fire screening, continuous-NLP screening, MIB-feasible solution repair, and warm starting of the MINLP when a discrete solve remains necessary.
The numerical studies demonstrate both the benefit and computational cost of explicitly enforcing the MIB during allocation. The accuracy benefit of direct MINLP allocation relative to rounding increases with minimum firing duration, but requires substantially greater solution time than the continuous NLP. The staged procedure reduces the frequency with which the MINLP must be invoked, although its effectiveness depends on the MIB and the distribution of requested control wrenches. In the closed-loop inspection simulation, the complete MIB-aware pipeline reduces the RMS position and pointing errors by 63.3% and 82.8%, respectively, relative to endpoint rounding. These improvements are accompanied by a 93.4% increase in applied impulse and higher allocation runtime, demonstrating a direct tradeoff among executable control accuracy, thruster usage, and computational effort.
Future work will focus on reducing the computational cost of MIB-aware allocation while preserving MIB feasibility and solution quality. Learning-based methods could provide rapid estimates of promising manipulator configurations and thruster activation patterns, which could either be accepted after verification or used to initialize the optimization procedure.

Disclaimer

The views expressed are those of the authors and do not reflect the official guidance or position of the United States Government, the Department of Defense, the United States Air Force, or the United States Space Force

References

[1] Arney, D. D., J. Mulvaney, C. Williams, W. Ruperto-Hernández, J. Friz, C. Stockdale, J. Nelson, and R. Rivera-Vargas. 2024. In-Space Servicing, Assembly, and Manufacturing (ISAM) State of Play.

[2] Lawless, J., S. Patane, R. Rollins, M. Ledbetter, and R. Cook. 2022 Oct. Future ISAM Architectures for National Security Space. In: *ASCEND 2022*, American Institute of Aeronautics and Astronautics, Las Vegas, Nevada & Online. https://doi.org/10.2514/6.2022-4299.

[3] Opiela, J. N. 2024 Apr. Orbital Debris Charts, April 2024.

[4] Fox, G., R. Salazar, H. Habib-Agahi, and G. F. Dubos. 2013. A Satellite Mortality Study to Support Space Systems Lifetime Prediction. In: *2013 IEEE Aerospace Conference*, IEEE, Big Sky, Montana, pp. 1–9. https://doi.org/10.1109/AERO.2013.6497352.

[5] Ferrone, K. L., T. Tran, M. Cavanaugh, and N. Brown. 2019. 2019 Satellite Lifetime Study. Tech. Rep. TOR-2019-02620, The Aerospace Corporation.

[6] Mukherjee, R. 2023 Oct. Survey Of Select Recent In-Space Servicing Assembly and Manufacturing Related Robotics Projects at The Jet Propulsion Laboratory. In: *ASCEND 2023*, American Institute of Aeronautics and Astronautics, Las Vegas, Nevada. https://doi.org/10.2514/6.2023-4700.

[7] Goodman, J. L. 2006 Sep. History of Space Shuttle Rendezvous and Proximity Operations. *Journal of Spacecraft and Rockets*, 43(5):944–959. https://doi.org/10.2514/1.19653.

[8] National Aeronautics and Space Administration. n.d. OSAM-1: On-Orbit Servicing, Assembly, and Manufacturing 1. NASA Goddard Space Flight Center. URL https://nexis.gsfc.nasa.gov/OSAM-1.html, accessed: 2026-08-11.

[9] Biesbroek, R., S. Aziz, A. Wolahan, S. Cipolla, M. Richard-noca, and L. Piguet. 2021. The ClearSpace-1 mission: ESA and ClearSpace team up to remove debris. In: *8th European Conference on Space Debris*.

[10] Pasand, M., A. Hassani, and M. Ghorbani. 2017 Jul. A Study of Spacecraft Reaction Thruster Configurations for Attitude Control System. *IEEE Aerospace and Electronic Systems Magazine*, 32(7):22–39. https://doi.org/10.1109/MAES.2017.160104.

[11] Meub, J. H. 2011. Continuous Low Thrust for Small Spacecraft Proximity Operations. Master's thesis, Missouri University of Science and Technology.

[12] Schulte, P. Z., and D. A. Spencer. 2016 Jan. Development of an Integrated Spacecraft Guidance, Navigation, & Control Subsystem for Automated Proximity Operations. *Acta Astronautica*, 118:168–186. https://doi.org/10.1016/j.actaastro.2015.10.010.

[13] Curti, F., M. Romano, and R. Bevilacqua. 2010 Jul. Lyapunov-Based Thrusters' Selection for Spacecraft Control: Analysis and Experimentation. *Journal of Guidance, Control, and Dynamics*, 33(4):1143–1160. https://doi.org/10.2514/1.47296.

[14] Wang, M., and Y. Xie. 2009 Dec. Design of the Optimal Thruster Combinations Table for the Real Time Control Allocation of Spacecraft Thrusters. In: *Proceedings of the 48th IEEE Conference on Decision and Control (CDC) Held Jointly with 2009 28th Chinese Control Conference*, pp. 5063–5068. https://doi.org/10.1109/CDC.2009.5400664.

[15] Schmucker, F. 2020. Development and Validation of an Efficient Thruster-Based Control Strategy for Orbital Robots. Master's thesis, Technical University of Munich.

[16] Yang, S.-z., L. Wang, and P. Sun. 2011 Feb. Optimal Thrust Allocation Logic Design of Dynamic Positioning with Pseudo-Inverse Method. *Journal of Shanghai Jiaotong University (Science)*, 16(1):118–123. https://doi.org/10.1007/s12204-011-1104-9.

[17] Fauré, M., D. Henry, J. Cieslak, P. Lachevre, and F. Ankersen. 2023 Oct. Optimization of Spacecraft Thrusters Configuration Under Fault Diagnosability and Recoverability Constraints. *IEEE Transactions on Aerospace and Electronic Systems*, 59(5):5275–5286. https://doi.org/10.1109/TAES.2023.3256976.

[18] Northrop Grumman. n.d. SpaceLogistics: Making Satellite Servicing Routine. Northrop Grumman. URL https://www.northropgrumman.com/what-we-do/space/space-logistics-services, accessed: 2026-08-11.

[19] Astroscale. n.d. LEXI-P Servicer. Astroscale Missions. URL https://www.astroscale.com/en/missions/lexi-p, accessed: 2026-08-11.

[20] Zlotnik, D., S. Di Cairano, and A. Weiss. 2017 Aug. MPC for Coupled Station Keeping, Attitude Control, and Momentum Management of GEO Satellites Using on-off Electric Propulsion. In: *2017 IEEE Conference on Control Technology and Applications (CCTA)*, pp. 1835–1840. https://doi.org/10.1109/CCTA.2017.8062723.

[21] Caverly, R., S. Di Cairano, and A. Weiss. 2020 Jan. Control Allocation and Quantization of a GEO Satellite with 4DOF Gimbaled Thruster Booms. In: *AIAA Scitech 2020 Forum*, American Institute of Aeronautics and Astronautics, Orlando, FL. https://doi.org/10.2514/6.2020-1687.

[22] Wei, Z., T. Long, R. Shi, and X. Song. 2022 May. Electric Thruster Configuration Design Optimization for Geostationary Satellites with Robotic Manipulators. *Advances in Space Research*, 69(10):3798–3813. https://doi.org/10.1016/j.asr.2022.02.048.

[23] Prakash, A., and D. K. Giri. 2025 Jan. Design and Dynamical Analysis of a Novel 6-DOF Spacecraft Maneuvering System Using Robotic Manipulator-Based Thruster System. In: *AIAA SCITECH 2025 Forum*, American Institute of Aeronautics and Astronautics, Orlando, FL. https://doi.org/10.2514/6.2025-0408.

[24] Schwend, W., and H. Schaub. 2026. Cascaded Control Architecture for Spacecraft with Arm-Mounted Thrusters using Reaction Torque Compensation. In: *AAS/AIAA Astrodynamics Specialist Conference*. Paper No. AAS 26-749.

[25] Schwend, W., A. Rogers, and H. Schaub. 2026. Hub Reaction Torque Reduction For Spacecraft Control Using Robotic Arm-Mounted Thrusters. In: *AAS/AIAA Astrodynamics Specialist Conference*. Paper No. AAS 26-693.

[26] Ma, G., and X. Kong. 2023 Jan. Planning Allocation for GTO-GEO Transfer Spacecraft with Triple Orthogonal Gimbaled Thruster Boom. *Mathematics*, 11(13):2844. https://doi.org/10.3390/math11132844.

[27] Schwend, W., and H. Schaub. 2025 Aug. Planar Spacecraft Control Through One Degree of Freedom Time-varying Thruster Configurations. In: *AAS/AIAA Astrodynamics Specialist Conference*. Paper No. AAS 25-786.

[28] Sena, F., R. Vijayan, H. Mishra, and M. De Stefano. 2026 Jun. Center-of-Mass and Attitude Control of an Orbital Manipulator: A Novel Control Strategy. *IEEE Robotics and Automation Letters*, 11(6):6871–6878. https://doi.org/10.1109/LRA.2026.3682621.

[29] Alcorn, J., H. Schaub, and S. Piggott. 2017 Sep. Steady-State Attitude and Control Effort Sensitivity Analysis of Discretized Thruster Implementations. *Journal of Spacecraft and Rockets*, 54(5):1161–1169. https://doi.org/10.2514/1.A33709.

[30] Watts, A. 2021. MHT-1N Monopropellant Hydrazine Thruster. Tech. rep., Nammo (U.K.) Limited, 47 Westcott Venture Park, Wescott, Buckinghamshire, United Kingdom. URL https://www.nammo.com/wp-content/uploads/2021/04/2021-Nammo-Westcott-MHT-1N-Hydrazine-Thruster.pdf, product datasheet.

[31] ArianeGroup. 2021 Nov. Spacecraft Propulsion: Mono Propellant System PM25. Tech. rep., ArianeGroup, Lampoldshausen, Germany. URL https://www.space-propulsion.com/brochures/propulsion-systems/orbital-propulsion-expertise.pdf, product datasheet.

[32] McKevitt, F. X., R. F. Eggers, and C. W. Bolz. 1971 Jun. Planetary Explorer Liquid Propulsion Study. Tech. Rep. NASA-CR-123369, Rocket Research Corporation, Redmond, Washington. URL https://ntrs.nasa.gov/citations/19720004053, Rocket Research Corporation report RRC-71-R-249, prepared under NASA Contract NAS5-11296.