doi: 10.82124/CEAS-GNC-2026-065 CEAS-GNC-2026-065

A unified framework for real-time optimal control allocation in space systems

Jesús Ramírez GNC Engineer, AOCS/GNC Department, SENER Aeroespacial y Defensa, C/Severo Ochoa, 4 – 28760 Tres Cantos, Madrid, Spain. jesus.ramirez@aeroespacial.sener (Corresponding author)

Joost Veenman Project Manager and robust control system expert, Navigation and Control Department, SENER Aeroespacial y Defensa, C/Severo Ochoa, 4 – 28760 Tres Cantos, Madrid, Spain. joost.veenman@aeroespacial.sener

Guillermo Alonso GNC Engineer, AOCS/GNC Department, SENER Aeroespacial y Defensa, C/Severo Ochoa, 4 – 28760 Tres Cantos, Madrid, Spain. guillermo.alonso@aeroespacial.sener

ABSTRACT

This paper tackles the challenge of managing multiple spacecraft actuators under complex and demanding mission conditions. We propose a flexible, convex optimization-based framework that efficiently allocates control commands across heterogeneous actuator sets while minimizing resource consumption, considering constraints, and performing secondary tasks autonomously. Unlike traditional methods, the algorithm ensures feasible solutions even in the presence of actuator failures, enabling robust operation in degraded scenarios without requiring mode-specific adaptations. To illustrate its versatility, we show how the algorithm can be tailored to a launcher-booster powered descent and landing scenario. Results on the allocation performance are provided for nominal and degraded scenarios and compared to a conventional approach. Altogether, this work establishes a scalable and reliable actuator management solution suitable for next-generation space missions.

Keywords: Real-time embedded (convex) optimization; control allocation; actuator management; powered descend and landing

Nomenclature

AOCS	=	Attitude and Orbit Control System
CMG	=	Control Momentum Gyro
DOF	=	Degree of Freedom
GNC	=	Guidance, Navigation, and Control
IPM	=	Interior Point Method
MIB	=	Minimum Impulse Bit
MPC	=	Model Predictive Control
MT	=	Magnetic torquer
PWM	=	Pulse Width Modulator
QCQP	=	Quadratically Constraint Quadratic Program

QP = Quadratic Program
RWL = Reaction Wheel
THR = Thruster
RCS = Reaction Control System
TVC = Thrust Vector Control

1 Introduction

Spacecraft often incorporate multiple actuators, enabling control actions to be achieved through different combinations. Managing such over-actuated systems requires distributing control commands while accounting for limitations and optimizing secondary objectives such as fuel efficiency or reaction-wheel unloading. Traditionally, engineers have favored simple, mission-specific solutions over numerical optimization due to risk aversion and limited on-board computational resources. As a result, suboptimal strategies such as lookup tables or pseudoinverse-based approaches have been widely used [1–3].

Recent advances in onboard computing capabilities have made real-time numerical optimization increasingly feasible, enabling more flexible and efficient control allocation strategies. Optimization-based approaches have also demonstrated strong fault-tolerance properties, supporting automatic reconfiguration and extending mission lifetimes [4, 5]. Although similar functionalities can be approximated with offline-generated lookup tables, online optimization consistently provides better adaptability and overall performance [3].

Within the optimization domain, convex optimization, particularly quadratic programming solved via interior-point methods (IPMs), has proven both effective and scalable [6–8]. In control allocation applications, IMPs have shown deterministic execution times that are compatible with space mission requirements outperforming traditional simplex-based alternatives[9].

Building on our previous work in [10], we leverage a convex optimization-based actuator management framework capable of allocating control commands across an arbitrary set of (heterogeneous) actuators. The formulation accommodates a wide range of possible objectives and constraints, including minimization of fuel/propellant/power consumption, reaction wheel null-space management, continuous or periodic desaturation via magnetic torquers or thrusters, automatic reconfiguration following actuator failures, and actuator operational domain enforcement.

In this paper we extend the framework to include Thrust Vector Control (TVC) actuators with variable throttle and gimbal angles in a reusable launcher landing scenario. We demonstrate that, with the addition of quadratic constraints in the control allocation problem, the framework from [10] can be adapted to optimally hybridize Reaction Control Systems (RCS) with TVC while preserving the convexity of the underlying optimization problem.

The algorithm is implemented using the interior-point solver from the Sener Optimization Toolbox (SOTB) [11, 12], for which a customized version was previously validated through processor-in-the-loop (PIL) tests in [10]. Nevertheless, it should be noted that the proposed version of the solver supporting quadratic constraints has not yet been customized or validated through a process that would qualify it for on-board use.

The remainder of the paper is organized as follows: Section 2 introduces how a diversity of actuator management problems can be cast as QP. Section 3 then presents a unified formulation that subsumes these cases into a general algorithm and illustrates its adaptation to an RCS-TVC launcher configuration. Subsequently, Section. 4 details the powered descent and landing (PDL) benchmark used for performance evaluation. Section 5 then reports the simulations results, including a comparison with a conventional approach. The paper concludes with some final remarks in Section 6.

2 Actuator management for space applications: A concise overview

Before we present our results, let us first provide some further context by giving a concise overview of typical actuator management problems that appear in space applications. When focusing on satellites, one typically considers momentum-exchanging devices for attitude and orbit control [13]. Here, reaction wheels (RWLs) are the most widely adopted actuators for attitude control. In support of this, magnetic torquers (MTs) or thrusters (THRs) can be used to desaturate the wheels (i.e., RWLs wind up due to the absorption of external torque perturbations). In case of small- to medium-sized satellites in low-Earth orbit, MTs can be considered in a continuous desaturation scheme. Alternatively, thrusters must be adopted to periodically off-load the wheels. It is also not uncommon that spacecraft attitudes are controlled with thrusters as is done in missions such as Euclid [14], Proba-3 [15], and LISA [16].

For reusable space transportation vehicles, momentum absorption devices alone cannot provide enough torque to stabilize the system. In such cases, cold gas thrusters are typically used to enable torque control (in some scenarios hot gas is used instead), while throttable hot gas thrusters with thrust vector control (TVC) are employed to generate both force and torque. For Earth landings, for example, with reusable launcher stages, aerodynamic surfaces can also be utilized, primarily to compensate for aerodynamic torques but additionally to contribute to overall torque control.

From a mathematical point of view, given a control command $f \in \mathbb{R}^m$ on a spacecraft, the control allocation problem can be defined by finding actuator actions $U = \mathrm{col}(u_1, \cdots, u_n) \in \mathbb{R}^n$ by solving the equation

$$f = BU. \tag{1}$$

Here, $B \in \mathbb{R}^{m \times n}$ corresponds to the control or dispatching matrix which maps the actuator actions into forces and torques acting on the spacecraft. For a 6 DOF control problem, the matrix B must have rank 6 to ensure full controllability. One can show that at least 7 thrusters in an RCS are needed for realizing controllability in 6 DOF, or at least 3 RWLs for a 3-axis stabilized attitude control, all depending on the particular configuration. On the other hand, it is common practice to consider redundant schemes in which more actuators are available than strictly required. In such cases, the matrix B has a null-space which can be exploited to generate identical actions with different combinations of actuators (i.e., the solution, U^*, of (1) is non-unique). This opens up the possibility to impose additional constraints on the control allocation problem such as minimizing fuel consumption when using thrusters, avoid zero-crossings in RWL clusters, desaturate RWLs with MTs, among others.

Many of these problems can be formulated as convex optimization problems, or, more specifically, quadratic programs (QPs), which admit the form:

$$\min_{x \in \mathbb{R}^N} \quad \frac{1}{2} x^\mathsf{T} Q x + q^\mathsf{T} x \tag{2a}$$

$$\text{s.t.} \quad Ax \leq b \tag{2b}$$

Here $x \in \mathbb{R}^N$ is the optimization variable (typically but not limited to $x = U$), while $Q = Q^\mathsf{T} \in \mathbb{R}^{N \times N}$, $Q \succcurlyeq 0$ (i.e., Q is positive semi-definite), $q \in \mathbb{R}^N$, $A \in \mathbb{R}^{k \times N}$, and $b \in \mathbb{R}^k$ gather the control allocation problem structure.

Several toolkits capable of efficiently solving (2b) are readily available. Well-known (open-source) examples include `ACADOS` [17] and `CVX` [18], which, through user-selectable solvers, not only solve the problem but can also generate autocoded `C` implementations. Alternatively, `quadprog` from `MATLAB` can be employed to reproduce the experiments presented in this work. Nonetheless, due to software memory constraints and hardware limitations, tailored solvers with fully accessible code are generally preferred for maximizing performance, especially in space applications. This motivated the development of the Sener Optimization Toolbox (SOTB) as further discussed in Section 3.2.

The simplest and most common solution to this problem is for the special case in which $x = U$, $A = 0$, $b = 0$, $q = 0$, and $Q = I$. This corresponds to the (unconstrained) least-square problem where we minimize the l_2-norm (i.e., $|U|_2 \coloneqq \sqrt{\Sigma_{i=1}^{n} u_i^2}$) subject to $f = BU$. This problem has the following analytical solution:

$$U^* = B^+ f. \tag{3}$$

Here $B^+ \coloneqq B^T \left(BB^T\right)^{-1}$ denotes the Moore–Penrose or pseudoinverse of B. Although convenient, this solution has severe limitations since actuator constraints are not accounted for, which could result in commands that exceed the actuator limits. Indeed, this solution is generally unsuitable for thrust dispatching or any other actuator which cannot provide negative actuation. Furthermore, minimization of the l_2-norm, which tends to equilibrate actuation within actuators, is not appropriate for propellant minimization.

In Table 1, we present a selection of features that can be achieved by solving the QP in (2), but are not attainable through direct pseudo-inverse application[1]. In the following section, we formalize the terms in (2) to ensure that all these features are jointly addressed within the control allocation process.

Table 1 Selection of optimal control allocation features

Classical Approach limitation	**Optimization feature**
Suboptimal solutions arising from the pseudoinverse application whenever $B^+ f$ saturates an actuator.	Actuation limits consideration (including not only torque or force constraints but, e.g., RWLs momentum saturation envelopes).
Least-square l_2-norm unsuitable for minimizing fuel consumption.	Thrust dispatching minimizing l_1-norm (i.e., $\|u\|_1 \coloneqq \Sigma_{i=1}^{n} \|u_i\|$) which corresponds to minimizing the fuel consumption.
Complex/heuristic null-space management.	Flexibility in how the null space is managed to: prioritize a working regime of the actuators (e.g., RWLs around a speed inducing low micro-vibrations), avoid or deal with saturation, and minimize overall consumption. Null space management adapted to situation.
Conservative solutions to avoid saturation.	Full actuators' set exploitation enhancing system agility.
Dedicated Fault Detection Isolation and Recovery (FDIR) solutions for failure scenarios.	Adaptation to available actuators without requiring FDIR specific algorithms or tasks, allowing continuation of operations or smooth transitions to safe modes with minimum degraded performance.
Generally limited to allocation.	Autonomous execution of secondary tasks (e.g., continuous RWLs off-loading with MTs).
Difficulties in handling unfeasible commands.	Prioritization of certain tasks of the control allocation under unfeasible commands with minimum errors.
Difficulties in actuating different types of actuators simultaneously.	Hybridization of different actuators even working at different frequencies.
Challenges in managing certain actuators, e.g., Control Moment Gyroscopes (CMGs).	Flexibility to cover multiple actuators, including CMGs (These are not addressed in this paper, but the application can be found in previous work [11]).

[1]Dedicated solutions or suboptimal add-ons, such as look-up tables (LUTs), may enable some of these features individually. In contrast, the optimization-based approach allows for their simultaneous and comprehensive treatment.

3 A generic optimization-based algorithm for actuator management

Now that we have given ample motivation for optimization-based control allocation, let us present our main contribution of the paper: a unifying approach that covers the features described in the previous section and many others. In this algorithm, the full range of each actuator and the null space of the entire hybrid cluster can be managed while minimizing a cost function that gathers the stated objectives by means of a convex formulation. Convex optimization is a key ingredient due to benefits such as guarantees for convergence and convergence rates, the absence of local optima, and the availability of efficient solvers. The algorithm is designed to cover generic satellite cases with typical actuator suites, yet with the flexibility to cover special cases. The default set of actuators considered in the sequel are reaction wheels, magnetic torquers, and thrusters. The algorithm also allows changing the configuration (i.e., which subset of actuators is used) without requiring any modification at the algorithm level. This facilitates the adaptation to different GNC modes, to hardware failure scenarios, and even to different missions with similar architectures.

The key to translating control allocation problems into convex optimization algorithms is the formulation of high-level objectives by means of mathematical constraints or cost penalties compatible with (2). This has resulted in the following QP:

$$\min_{U \in \mathbb{R}^n,\, h \in \mathbb{R}^{n_R}} \quad \frac{1}{2}U^\mathsf{T} R U + r^\mathsf{T} U + \frac{1}{2}\left(h - h_{\text{ref}}\right)^\mathsf{T} W \left(h - h_{\text{ref}}\right) + \frac{1}{2}\left(f - BU\right)^\mathsf{T} L \left(f - BU\right) \tag{4a}$$

$$\text{s.t.} \quad u_i^{\min} \le u_i \le u_i^{\max}, \quad i \in \{1, \cdots, n\}, \tag{4b}$$

$$h_j = h_{0j} + \Delta t \cdot u_j, \quad j \in \{1, \cdots, n_R\}, \tag{4c}$$

$$h_j^{\min} \le h_j \le h_j^{\max}, \quad j \in \{1, \cdots, n_R\},, \tag{4d}$$

$$\frac{1}{2}u^T Q_k u + q_k^T u \le \alpha_k \quad k \in \{1, \cdots, n_k\}. \tag{4e}$$

As before, B denotes the dispatching matrix, f the commanded force/torque vector, and $U = \text{col}(u_1, \cdots, u_n)$ with $u_i \in [u_i^{\min}, u_i^{\max}]$ the to-be-computed vector containing the reaction wheel, magnetic torquer, and thruster actions respectively, i.e., $n = n_R + n_M + n_T$. Then $h = \text{col}(h_1, \cdots, h_{n_R}) \in \mathbb{R}^{n_R}$ with $h_j \in [h_j^{\min}, h_j^{\max}]$ denotes the stored angular momentum in each wheel, h_{ref} the reference objective value, h_{0j} the current state, and Δt the time of action of U, i.e., the control allocation step. Convex cone constraints are allowed through (4e), required for, e.g., TVC actuation limitations. Here $Q_k \in \mathbb{R}^{n \times n}$ and $\alpha_k \in \mathbb{R}^1$ represent n_k second-order cone or Euclidean ellipsoids [19], which might be non-centered in zero through vector $q_k \in \mathbb{R}^{\varkappa}$. Finally, $R \succcurlyeq 0$, $W \succcurlyeq 0$, and $L \succcurlyeq 0$ are (symmetric) positive semi-definite (diagonal) weighting matrices of compatible dimensions, while r weights the linear part of the cost function.

The structure of (4) enables simultaneous consideration of all the features listed in Tab. 1 within a single optimization framework. By tuning the cost penalties, different objectives can be weighted and traded off. A high value of L prioritizes minimization of allocation error, while R and r promote command realization with minimal actuation effort. In our experience, this formulation has consistently achieved allocation errors below actuator tolerances, without requiring ad-hoc mechanisms such as command saturation. When an exact realization is infeasible due to hardware limitations, the optimizer naturally converges to the closest feasible actuation. Moreover, this structure inherently supports null-space exploitation, actuator hybridization, and secondary objectives such as autonomous RWL unloading. By selecting the penalty W sufficiently smaller than R, r, and L, desaturation is triggered only when appropriate —specifically, when RWLs approach saturation (reflected by the increase of $(h - h_{ref})$), command demands are low, and a "cheap" actuator such as an MT is available. The same unmodified formulation (except for parameter retuning) can also operate in degraded configurations, delivering the best possible actuation with the remaining hardware. This reduces the need for multiple FDIR modes

or dedicated fallback algorithms. Finally, as demonstrated in [10] and in this work, (4) can be easily tailored to mission-specific constraints. Overall, it provides a generic yet highly adaptable framework for real-time optimal control allocation in space systems.

3.1 Adaptation for RCS and TVC thrust dispatching

Let us now demonstrate how the optimization problem (4e) can be utilized for a reusable launch vehicle actuated by thrusters, i.e., a RCS configuration in combination with a TVC with variable throttle. For another use case related to the control of satellites, we refer the reader to [10].

Let $n_T = n_{RCS} + n_{TVC}$ with n_{RCS} being the number of cold gas thrusters in the RCS and n_{TVC} the number of thrusters in the TVC. On the one hand, RCS actuation is modeled as a thrust component along the thruster direction, bounded by a maximum value and modulated in time via a Pulse Width Modulator (PWM). The thrust is constrained to be nonnegative, with the Minimum Impulse Bit (MIB) neglected and instead handled within the PWM to avoid introducing nonconvexities:

$$0 \leq u_i \leq T_{\max}^{RCS}, \quad i \in \{1, \cdots, n_{RCS}\}. \tag{5}$$

On the other hand, to avoid nonlinearities in the dispatching matrix, TVC thrusters are not modeled in terms of deflection angles and thrust levels,as this would introduce sine and cosine dependencies. Instead, each TVC is represented directly as a force vector in its local reference frame, $v_j = [v_j^x, v_j^y, v_j^z]^\mathsf{T}$, so that the total number of control variables is $n_{TVC} = 3 \cdot N_{TVC}$, where N_{TVC} denotes the number of TVC thrusters. Eq. (6) expresses the second-order cone constraint for the force vectors v_j, limited by a nonzero minimum and a maximum thrust magnitude, as illustrated in Fig. 3b.

$$\left(T_{\min,j}^{TVC}\right)^2 \leq v_j^T v_j \leq \left(T_{\max,j}^{TVC}\right)^2, \quad v_j^T Q_j v_j \leq 0 \quad \text{for all} \quad j \in \{1, \cdots, N_{TVC}\}. \tag{6}$$

$$Q_j = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -\tan(\beta_j) \end{bmatrix}. \tag{7}$$

Here, Q_j is defined under the assumption that the axis of symmetry of the thrust cones of each TVC thruster is aligned with the z-axis of the body. Moreover, β_j denotes the maximum deflection angle of the each TVC thruster. For simplicity, the remainder of the paper assumes that all TVC thrusters share identical constraints (i.e., $T_{\max,j}^{TVC} = T_{\max}^{TVC}$, $T_{\min,j}^{TVC} = T_{\min}^{TVC}$, $Q_j = Q \; \forall j \in \{1, \cdots, N_{TVC}\}$).

The minimum bound in (6) introduces a non-convexity in an otherwise convex constraint. To fix this, lossless convexification techniques are available in [6], which keep the problem convex without requiring any approximation or linearization. In this work, however, assuming that the cone angle β is sufficiently small, we employ a simple convex approximation of the thrust envelope to reduce computational effort, thereby replacing (6) by:

$$T_{\min}^{TVC} \leq v_j^z \leq \cos(\beta)\, T_{\max}^{TVC}, \quad v_j^T Q\, v_j \leq 0 \quad \text{for all} \quad j \in \{1, \cdots, N_{TVC}\} \tag{8}$$

Note that this approximation introduces only minimal conservatism. For typical values of $\beta \leq 5°$, at most 0.42% of the actual maximum achievable thrust may be unexploited in the upper-bound, and at most 0.38% in the lower-bound, as sketched in Fig. 1.

Once the required constraints are defined, we can proceed with normalizing the variables as $u_i = u_i^{\max} \cdot \bar{u}_i$ for $i \in \{1, \cdots, n_{RCS}\}$ and $v_j = T_{\max}^{TVC} \cdot [\tan(\beta) \cdot \bar{v}_j^x, \tan(\beta) \cdot \bar{v}_j^y, \bar{v}_j^z]^\mathsf{T}$ for convenience and to improve the solver's numerical performance. Grouping penalty terms in a single matrix $H \succcurlyeq 0$ and vector c, and ignoring constant offsets of the cost function we finally obtain:

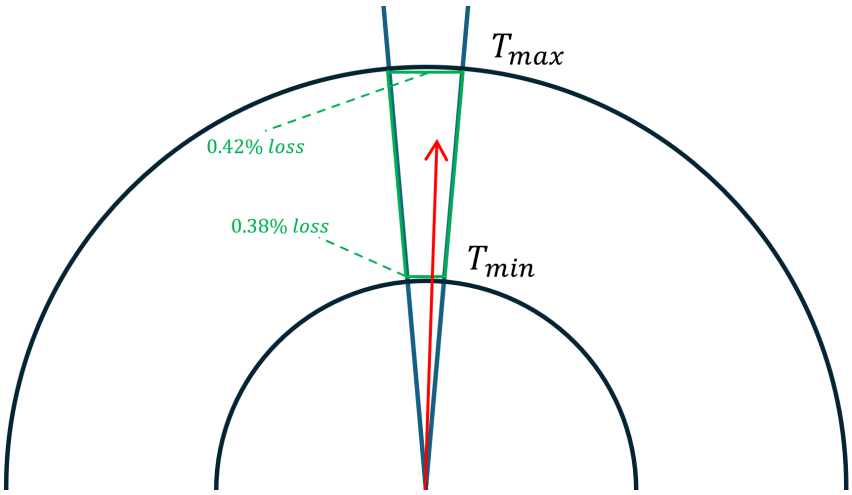


Fig. 1 Convex constraint approximation effect example sketch

$$\min_{\bar{U} \in \mathbb{R}^{n_T}} \quad \frac{1}{2}\bar{U}^\mathsf{T} H \bar{U} + c^\mathsf{T}\bar{U} \tag{9a}$$

$$\text{s.t.} \quad 0 \leq \bar{u}_i \leq 1, \qquad i \in \{1, \cdots, n_{RCS}\}, \tag{9b}$$

$$\frac{T^{TVC}_{\min}}{T^{TVC}_{\max}} \leq \bar{v}^z_j \leq \cos(\beta), \quad j \in \{1, \cdots, N_{TVC}\}, \tag{9c}$$

$$\bar{v}_j^\mathsf{T} \bar{Q} \bar{v}_j \leq 0, \qquad j \in \{1, \cdots, N_{TVC}\}. \tag{9d}$$

Here H is structured as

$$H = R + \bar{B}^\mathsf{T} L \bar{B} \tag{10}$$

with $\bar{U} = \text{col}(\bar{u}_1, \cdots, \bar{u}_{n_{RCS}}, \bar{v}^x_1, \bar{v}^y_1, \bar{v}^z_1, \cdots, \bar{v}^x_{N_{TVC}}, \bar{v}^y_{N_{TVC}}, \bar{v}^z_{N_{TVC}})$, and where $\bar{B}$ is the dispatching matrix for the normalized variables, i.e., $\bar{B} = B \cdot \text{diag}([u_1/\bar{u}_1, \cdots, u_{n_{RCS}}/\bar{u}_{n_{RCS}}, v^x_1/\bar{v}^x_1, \cdots, v^z_{N_{TVC}}/\bar{v}^z_{N_{TVC}}])$. In addition, the linear part of the cost function, c, is given by:

$$c = r - f^\mathsf{T} L \bar{B} \tag{11}$$

Finally, the conic constraints after normalization are given by:

$$\bar{Q} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & -1 \end{bmatrix} \tag{12}$$

Note here the slight abuse of notation by keeping the same weighting variables (R, L, r) as in the non-normalized case.

3.2 Implementation

The optimal control allocation problem can be complemented with an upper-level software layer, as illustrated in Figure 2, which updates its internal state based on the current system status. This state consists of the parameters that characterize the instantaneous capabilities of the control allocation algorithm and, in particular, accounts for actuator availability. The latter can be seamlessly incorporated into the optimization problem by setting to zero the corresponding columns of B associated with unavailable actuators. It is further noted that, by design, the optimization problem is always guaranteed to admit a feasible solution (see Remark 1).

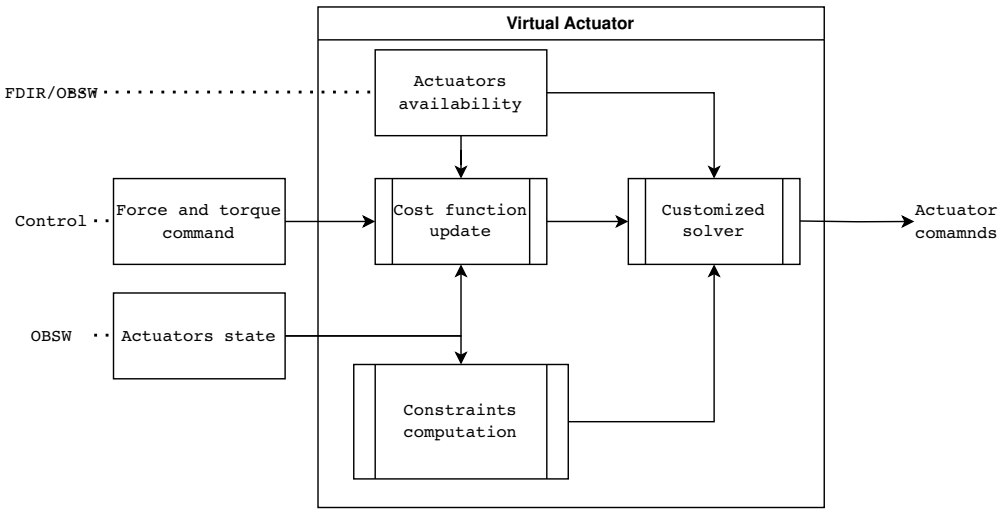


Fig. 2 Flow chart diagram of the autonomous actuator management algorithm

On the solver side, the QP (4) (and hence the special case (9)) can be efficiently solved with an Interior Point Method (IPM) able to manage Quadratically Constrained Quadratic Programs (QCQPs). In this paper, we adopt the generic QCQP solver from the Sener Optimization Toolbox (SOTB) to solve the control allocation described in the previous section. This is a MATLAB-based toolbox for real-time embedded optimization that facilitates the rapid prototyping and testing of QCQPs in space-graded hardware. For further reference, we refer the reader to [11, 12].

Remark 1 *Rather than enforcing control commands as hard constraints, the algorithm employs a relaxed quadratic penalty. This approach allows unrealizable commands to be handled gracefully and supports operation under degraded conditions. High penalties in torque and force errors ensure accurate tracking, whereas penalties in thruster usage help to minimize propellant consumption.*

Remark 2 *To maintain convexity, the minimum impulse bit (MIB) constraint for thrusters is set to a constant lower bound. While zero is used for generality, any value below 1 is acceptable, which is beneficial for high-precision missions using differential thrusting.*

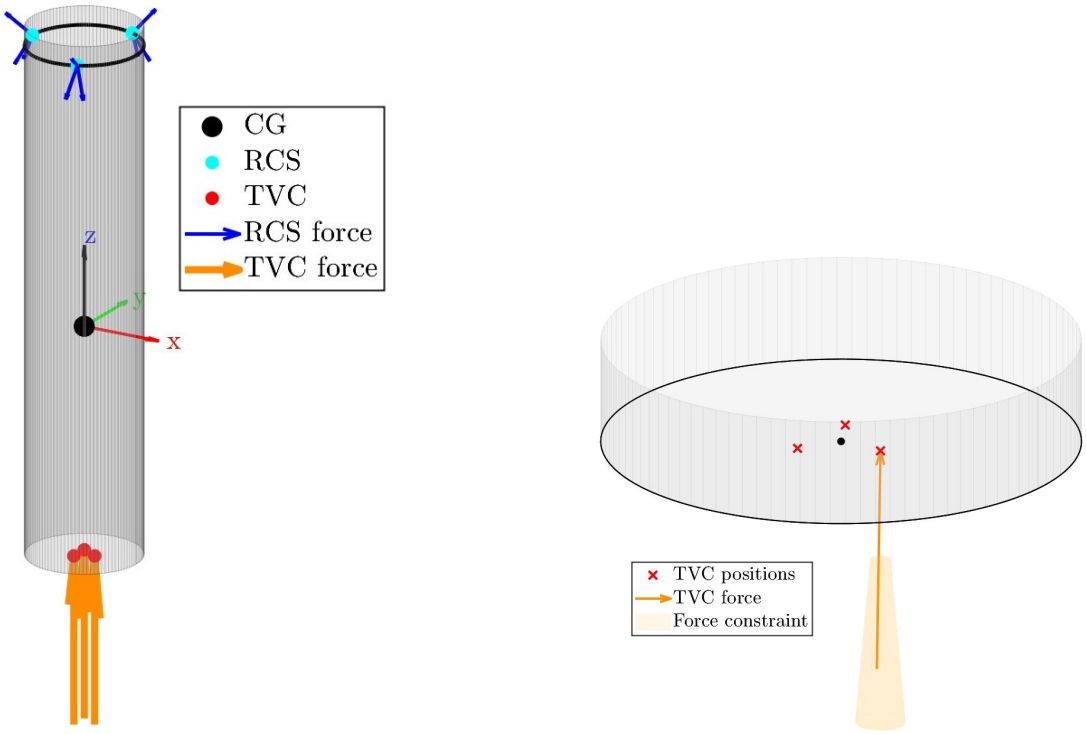
Remark 3 *In our recent work [10], we developed and tested a customized solver for control allocation problems compatible with the structure of* (4)*, but without the quadratic constraint* (4e)*. This solver was tailored for high numerical performance in real-time embedded optimization. In future work, we intend to extend this approach by generalizing the solver to incorporate quadratic constraints as well. On the other hand, TVC with independent gimbal rotations could admit a formulation with polytopic constraints, making the customized solver already compatible with the application.*

4 Application to Powered Descent and Landing scenario

4.1 Description of the benchmark

Let us now apply the proposed algorithm to a reusable launcher scenario to demonstrate its performance and capabilities. The test case is inspired by ongoing European Space Agency (ESA) activities targeting future reusable space transportation systems. The selected benchmark consists of a 75-ton (dry mass) rocket stage tasked with re-entering and performing a safe landing within a designated zone. Our analysis focuses on the final segment of the descent - after deorbiting, aerodynamic re-entry, and *nose-up* manoeuvres have been completed.

In this phase, referred to as the *final approach*, the system must execute a vertical propulsive descent, correcting any accumulated deviations from the earlier flight phases, while minimizing propellant consumption and touchdown velocities. The available actuators consist of nine hot-gas thrusters arranged in an RCS configuration and three trottable main engines equipped with a TVC, aligned with the vehicle's primary symmetry axis, as illustrated in Fig. 3a.



(a) Actuator configuration in the launcher vehicle **(b) Actuator configuration (TVC close-up view)**

Fig. 3 Reusable rocket stage sketch

The RCS thrusters are mounted near the top of the vehicle, positioned approximately 25 m above the Center of Gravity (CoG), which is assumed constant for the purposes of this paper. As illustrated in Fig. 3a, they are symmetrically distributed around the vehicle's outer structure outer structure at a radial distance of 5 m, with three thrusters located at each mounting position. Each thruster is capable of delivering a maximum force of $u_i^{max} = 220\,\text{N}$ along the direction defined by the corresponding column of the matrix D^{RCS}, which contains the unit vectors of the thruster directions expressed in the body frame.

$$D^{RCS} = -\begin{bmatrix} 0.4330 & 0.4330 & 0.8660 & 0.0954 & -0.3774 & -0.2820 & -0.3774 & 0.0954 & -0.2820 \\ 0.2500 & -0.2500 & 0.0000 & -0.4908 & -0.3280 & -0.8188 & 0.3280 & 0.4908 & 0.8188 \\ 0.8660 & 0.8660 & -0.5000 & 0.8660 & 0.8660 & -0.5000 & 0.8660 & 0.8660 & -0.5000 \end{bmatrix} \tag{13}$$

The 3 TVC thrusters are mounted near the base of the vehicle, positioned 20.25 m below the vehicle's CoG and offset 1 m from the longitudinal axis. As illustrated in Fig. 3b, they are arranged in an equilateral triangle configuration designed such that a uniform vertical thrust produces no net moment about the CoG. The TVC actuators are subject to the following limitations, using the notation introduced in Section 3.1: $T_{\text{min}}^{TVC} = 240\,\text{kN}$, $T_{\text{max}}^{TVC} = 600\,\text{kN}$ and $\beta = 5$ deg. With these definitions, the dispatching matrix $B = [B_{RCS}, B_{TVC}]$ such that $f = BU$, where $f = [F, \tau]^{\text{T}}$ represents the commanded force and

torque (in N and Nm, respectively), and I_3 denotes the 3×3 identity matrix:

$$B_{RCS} = \begin{bmatrix} & & & & & D_{RCS} & & & \\ 6.25 & -6.25 & 0 & -16.24 & -12.17 & -18.176 & 12.17 & 16.24 & 18.176 \\ -15.16 & -15.16 & -19.15 & -0.66 & 11.16 & 6.06 & 11.16 & -0.66 & 6.06 \\ 1.25 & -1.25 & 0 & 1.41 & -1.08 & 0.33 & 1.08 & -1.41 & -0.33 \end{bmatrix} \tag{14}$$

$$B_{TVC} = \begin{bmatrix} & & & I_3 & I_3 & I_3 & & & \\ 0 & 20.25 & 0 & 0 & 20.25 & -0.87 & 0 & 20.25 & 0.87 \\ -20.25 & 0 & -1 & -20.25 & 0 & 0.5 & -20.25 & 0 & 0.5 \\ 0 & 1 & 0 & 0.87 & -0.5 & 0 & -0.87 & -0.5 & 0 \end{bmatrix} \tag{15}$$

4.2 Tracked trajectory (guidance)

The Final Approach is governed by a dedicated Guidance, Navigation, and Control (GNC) mode. In this study, guidance trajectories are applied in open loop: precomputed feedforward force and torque commands are passed directly to the control allocation algorithm to assess its performance. To approximate realistic closed-loop behaviour, an additional term is included to mimic the contribution of a feedback controller (see Section 5 for further details).

Guidance strategies for final approaches in powered descent and landing (PDL) scenarios remains an active field of research and is widely recognized as a key enabler for successful landings. Given the current system state (position, velocity, mass, etc.) and environmental conditions (e.g., wind), the guidance function must compute a trajectory that safely drives the vehicle to the landing site. This is a complex problem that is outside the scope of this paper that will be presented elsewhere. The employed guidance algorithm relies on a Sequential Convex Programming (SCP) strategy inspired by, e.g., [6], and follows a similar formulation to recent works such as [20] and [21]. These approaches represent an evolution of the model predictive control (MPC) schemes employed by SpaceX for their booster landing [22]. The current implementation considers the full 6 DOF vehicle dynamics, modelled according to [23]:

$$\dot{x} = \frac{d}{dt} \begin{bmatrix} r_L \\ v_L \\ m \\ q_{L2B} \\ \omega_B \end{bmatrix} = \begin{bmatrix} v_L \\ \frac{1}{m}\left(T + D\right) + g \\ -\frac{||T||_2}{I_{sp}\, g_0} \\ \frac{1}{2}\, q_{L2B} \odot \omega_B \\ J^{-1}\left(M_T + M_D\right) \end{bmatrix}. \tag{16}$$

Here, r_L and v_L denote the position and velocity vector in the landing point frame (LPF) defined as an inertial frame for the duration of the manoeuvre, with its origin at the landing site, the z-axis pointing upward, and the x-y plane tangent to the local surface. The scalar m represents the vehicle mass, q_{L2B} is the quaternion describing the rotation from the LPF to the body frame, and ω_B the body angular velocity. The forces T and D correspond to thrust and aerodynamic drag, while M_T and M_D represent the associated moments; g is the (constant) gravitational acceleration, and J denotes the inertia matrix of the vehicle. For simplicity, the inertia is assumed to be constant, while the mass varies proportionally to the thrust magnitude according the specific impulse I_{sp}.

The formulated time-optimal Optimal Control Problem (OCP) is subject to the dynamics of (16). The control input is a thrust vector expressed in the body frame and applied at the centre of the TVC layout (indicated by the black dot in Fig. 3b) subject to:

$$N_{TVC} T_{min}^{TVC} \leq T_z \leq \cos(\beta) N_{TVC} T_{max}^{TVC}, \quad T^{\mathsf{T}} Q T \leq 0. \tag{17}$$

In other words, the thrust is modelled as the equivalent net force that would be produced by the TVC layout under uniform activation, while RCS effects are neglected at the guidance level. Note here that the minimum thrust is nonzero, since the final approach spans from the initial post nose-up ignition to touchdown, during which engine shutdown is not permissible due to safety constraints. Consequently, time-optimal planning is essential, as the required horizon length cannot be predicted in advance [20].

In summary, during the PDL scenario, a dedicated GNC system governs the final approach. The guidance module generates a full reference trajectory comprising both states and feedforward thrust/torque commands which must be distributed among the available actuators along with the feedback control inputs. This is precisely where the proposed allocation algorithm plays a central role. Hence, to assess its performance, trajectories are first generated with an SCP guidance algorithm. The resulting force and torque commands are then passed to the control allocation engine, complemented with the simulated feedback inputs, and the resulting input-output errors are evaluated to measure the allocation accuracy.

5 A selected overview of test results

Let us now preset a selection of test results to illustrate the capabilities of the developed actuator manager approach using the PDL scenario as a representative study case. For a fair comparison, we compare the algorithm's performance with a conventional method based on offline generated LUTs which were constructed using the same underlying optimization problem as in (9). These LUTs provide precomputed actuator configurations that generate unitary forces and torques along and about the three body axes; during operation , they are scaled according to commanded output and trimmed if saturation occurs. Following this baseline comparison, we demonstrate additional advantagers of the proposed method in several degraded scenarios involving actuator failures. In such cases, the LUT-based approach is unsuitable without additional tailoring of the use dedicated design fault detection isolation and recovery (FDIR) logic while the proposed algorithm adapts autonomously.

For the generated PDL trajectory (see Fig. 4), the selected representative initial conditions are:

- Distance from landing point $r_{L_0} = [200, 100, 500]^\mathsf{T}$ m.
- Initial velocity $v_{L_0} = [-10, 0, -80]^\mathsf{T}$ m/s.
- Initial mass $m_0 = 90$ tons, i.e., 15 tons of propellant.
- Initial attitude aligned with LPF, i.e., $q_0 = [0, 0, 0, 1]^\mathsf{T}$, without any angular velocity, i.e., $\omega_0 = [0, 0, 0]^\mathsf{T}$ rad/s.

With the extra parameters required for the simulations summarized in Tab. 2.

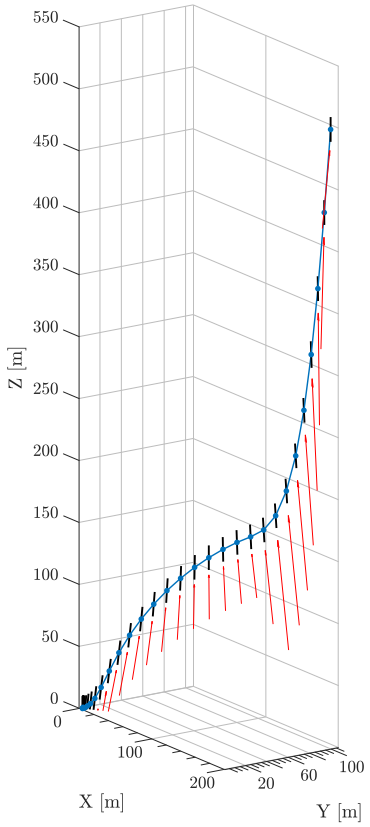


Fig. 4 PDL 3D trajectory in LPF frame, from SCP algorithm

Table 2 Main parameters of the PDL scenario and allocation algorithm

Parameter	Value
J_{xx}, J_{yy}, J_{zz}	$[17095, 17095, 1125]$tons$\cdot$ m^2
J_{xy}, J_{xz}, J_{yz}	$[0, 0, 0]$kg$\cdot$ m^2
u_i^{max}	220 N
T_{max}^{TVC}	600 kN
T_{min}^{TVC}	240 kN
β	5 deg
B_{RCS}	(14)
B_{TVC}	(15)
r	$10^1 \mathbf{1}_{n_{RCS}}$
R	blkdiag$(10^{-6} I_{n_{RCS}}, 10^6 I_{n_{TVC}})$
L_F	diag$([10^{10}, 10^{10}, 10^{10}])$
L_T	diag$([10^{11}, 10^{11}, 10^9])$
Solver tolerance	10^{-6}

5.1 Comparison with conventional allocation in nominal trajectory

We start by considering a nominal scenario in which all thrusters are available and operating nominally. The traditional approach was implemented as follows: For control allocation to the TVC thrusters, we adopt the approach [5], using a pseudoinverse-based method. Here, the commanded thrust magnitude is first evenly distributed across the three actuators, after which a thrust-dependent pseudoinverse is applied to compute the required gimbal deflections to satisfy the desired the force and torque. Saturation on thrust levels and gimbal angles is enforced afterwards. Unlike, [5], no dedicated null-space management strategy is employed in this implementation. Therefore we have:

$$||v_j||_2 = \max\left\{\min\left(F/N_{TVC}, T_{\max}^{TVC}\right), T_{\min}^{TVC}\right\}, \quad j \in \{1, \cdots, N_{TVC}\}, \tag{18a}$$

$$\delta = A^{+}\left(f - \left[0, 0, \sum_{j}^{N_{TVC}} ||v_j||_2, 0, 0, 0\right]^{\mathsf{T}}\right). \tag{18b}$$

Here $\delta = [\delta_{1,1}, \delta_{1,2} \cdots \delta_{N_{TVC},1}, \delta_{N_{TVC},2}]^{\mathsf{T}}$ is set of all TVC gimbal angles, while A^{+} denotes the pseudoinverse mapping from angles to actuator commands given the TVC forces ($||v_j||_2$) [5]. These angles are then saturated such that $\delta_{j,1}^2 + \delta_{j,2}^2 \leq \beta^2$ by maintaining the ratio $\delta_{j,1}/\delta_{j,2}$, for $i \in \{1, \cdots, N_{TVC}\}$, so that the envelope sketched in Fig. 3b is considered.

If torque errors (τ_r) arise due to saturation or pseudoinverse linearization in the previous TVC allocation step, they are redistributed accross the RCS thrusters using a dedicated pseudoinverse-based

solution. Residual force components are intentionally neglected to prioritize torque correction, where the RCS layout is more effective. Since the pseudoinverse may yield negative thrust commands, we apply a heuristic correction in the null-space of the dispatching matrix; a uniform positive bias is added to all thrusters so that the most negative command is shifted to zero. This bias is selected as the closest null-space vector to an equal distribution among all thrusters. However, this heuristic does not prevent potential saturation at the maximum force limit. Saturation is applied only after allocation; any resulting errors are left uncorrected, as mitigating them would require additional, specialized handling strategies. The required operations are:

$$B^{\tau}_{RCS} = [0_3, I_3] B_{RCS}; \tag{19a}$$

$$B^{+} = (B^{\tau}_{RCS})^{\top} (B^{\tau}_{RCS} (B^{\tau}_{RCS})^{\top})^{-1}; \tag{19b}$$

$$D = \text{null}(B^{\tau}_{RCS}); \tag{19c}$$

$$d = D \left(D^{\top} \left(D D^{\top} \right)^{-1} \mathbf{1}_{n_{RCS}} \right); \tag{19d}$$

$$u = \min \left\{ \left(B^{+} \tau_r + \min \left(B^{+} \tau_r / d < 0 \right) \cdot d \right), u^{\max} \right\}; \tag{19e}$$

with $\mathbf{1}_{n_{RCS}}$ a column vector of ones with n_{RCS} elements.

Remark 4 *These limitations and heuristic workarounds in the conventional allocation approach already highlight a key drawback compared to the proposed method. In contrast, our algorithm jointly minimizes errors across both RCS and TVC actuators in a unified optimization framework, inherently handling actuator constraints without relying on linearizations or ad hoc logic.*

Remark 5 *While more sophisticated pseudoinverse-based techniques, such as those in considered in [5], could improve the performance of conventional allocation methods, the goal of this paper is different. Our emphasis is on demonstrating that the proposed algorithm can be readily adapted to different mission scenarios without the need for custom-designed logic or algorithmic development, only parameter tuning is required.*

For the comparison, the reference trajectory generated in Section 4 is used as the commanded input. To emulate a feedback controller, we superimpose additional force and torque terms modeled as low-frequency white noise, scaled relative to the guidance feedforward signals. This creates more demanding command profiles and drives the allocation algorithms closer to their operational limits. Fig. 5 presents the resulting input–output errors for each control channel, defined as the difference between the commanded values and the achieved actuator outputs.

As can be seen, the optimization-based algorithm demonstrates improved performance in most of the cases, delivering actuation commands without errors whenever possible. The conventional approach achieves the same level of accuracy in case actuator saturation is not required to realize the commanded forces/torques or when the command is out of the actuation envelope. However, note that for the latter the optimization-based allocation behaves better, as it minimizes the overall error according to (9). This can also be observed for some cases in which the optimization-based algorithm errors is slightly above the conventional one, yet it is smaller in other components for the same instant.

In summary, the optimization-based allocation algorithm enhances the performance if compared with a conventional pseudo-inverse based solution for demanding scenarios in which the full actuation envelope is to be exploited, i.e., saturation of some actuators is unavoidable. The proposed algorithm is able to account for actuator constraints to provide the commands with minimum error (zero if possible), enhancing system agility and increasing GNC design robustness.

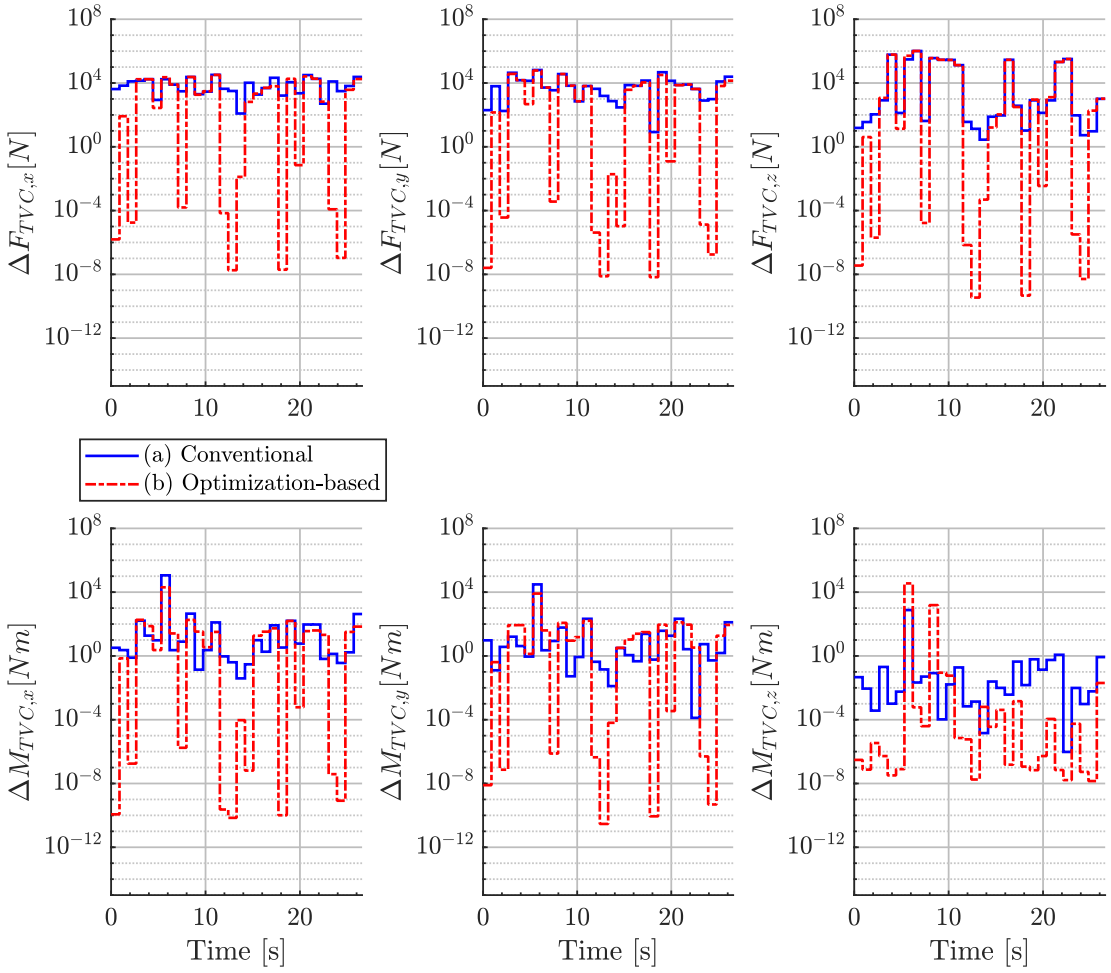


Fig. 5 Control allocation errors comparison, stress case

5.2 Degraded scenario

In the previous subsection, we have validated the proposed algorithm for nominal operations and demonstrated enhanced performance if compared to conventional approaches whenever saturation of actuators is required to achieve the command. In this section, we show how the optimization-based actuator manager can also operate in degraded scenarios. For the demonstration, we compare the allocation errors of the guidance feedforward commands (realizable without errors)[2] in a nominal scenario[3], i.e., with all actuators available, and:

- **Test 1:** *Deflection loss.* TVC thruster 1 loses gimbal capability, i.e., direction of application force $v_1^x = v_1^y = 0$.
- **Test 2:** *Thrust envelope loss.* The maximum thrust of TVC thruster 1 is reduced by 20 %, and the minimum thrust of TVC thruster 2 is increased by 20 %. The result is a reduced actuation envelope.
- **Test 3:** *RCS loss.* RCS thrusters 1, 4, 5 and 9 become unavailable.

[2]Commands are generated in the optimization scheme introduced in Sec. 4, thus accounting for system actuation limitations. Margins are included to account for the simplification introduced in (17). Note that this simplification is introduced for computational burden reduction, yet the complete envelope consideration under convex constraints is possible.

[3]Note that a comparison with conventional approaches is not possible as they would require dedicated FDIR solutions or null-space management algorithms, which is out of the scope of the present study.

Remark 6 *The tested degraded scenarios can be handled by the proposed algorithm without requiring modification. They can be considered by simply updating the parameters, mainly setting to zero the columns in B for which actuation is unavailable and/or replacing the upper and lower bounds in* (9).

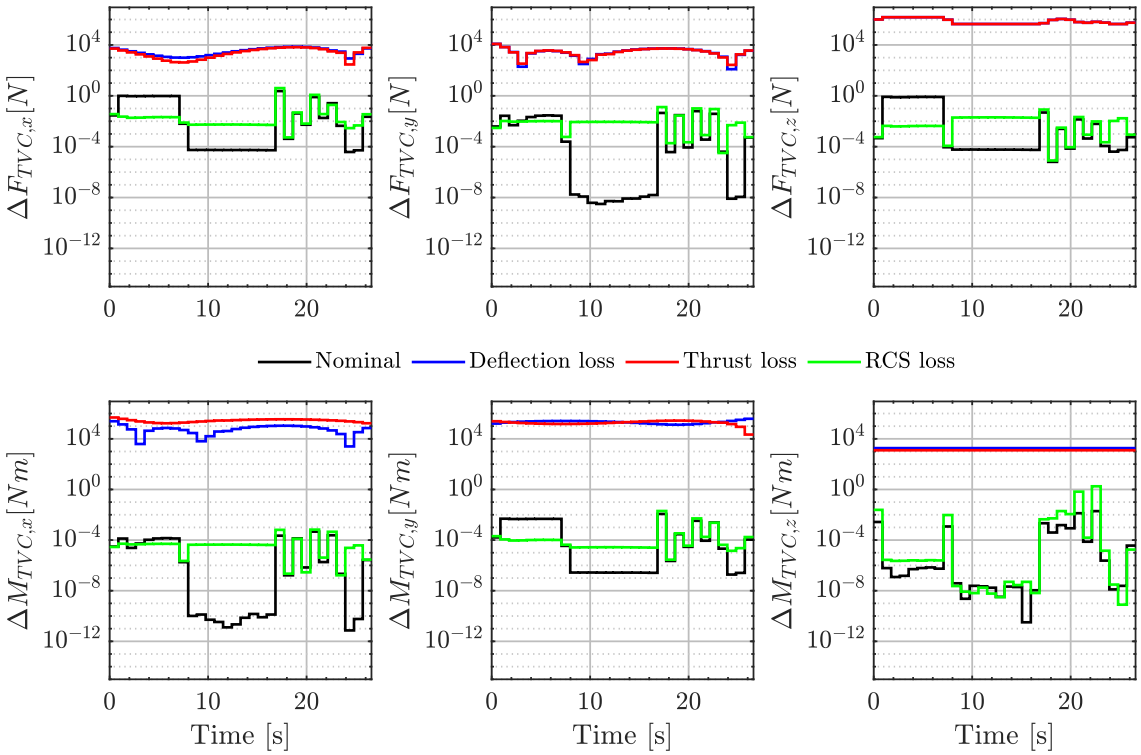


Fig. 6 Control allocation errors with only feedforward guidance, comparison of degraded scenarios

The different test results are shown in Fig. 6. First, since experiments commands are realizable with only TVC (ensured at guidance level), it can be observed that the proposed autonomous allocation algorithm generates negligible errors in the nominal scenario even when some RCS are unavailable. This result confirms the discussion of Section 3, i.e., the proposed optimization-based control allocation algorithm distributes commands without errors if those are feasible, even if the constraint of (1) is relaxed into a cost penalty. Furthermore, it shows the benefit of considering the TVC as an actuation vector instead of throttle plus gimbal angles, as no linearizations are required. Note that when RCS are available, those are used (hybridization) to further reduce the error and the overall consumption according to the cost penalties R and r. On the other hand, in Test 1 and 2 the input commands become unrealizable. We show that for those cases the optimization-based algorithm is still able to find a solution. Moreover, the computed actuation corresponds to the best actuation solution (according to tuning, i.e., L) to generate minimum allocation error.

Overall, this section shows that the proposed optimization-based actuator manager can operate in a variety of degraded scenarios with just parameter updating (i.e., without requiring FDIR specific algorithms or tasks). All while minimizing command errors, achieving zero errors for realizable commands with the available configuration.

6 Final remarks and conclusions

In this paper, we have formulated a generic and versatile convex optimization-based algorithm for the systematic distribution of control commands to a set of available actuators while minimizing objectives

such as fuel consumption and considering complex operational constraints. The proposed approach aids the realization of increasingly complex missions, reduces the operational complexity, and embers failure management in a smart way.

The algorithm's effectiveness was demonstrated on a representative launcher-booster powered descent and landing problem and compared to a conventional approach. The results indicate that the algorithm consistently produces feasible solutions and maintains functionality even under severely degraded conditions. Moreover, the results demonstrate improved performance if compared to a traditional approach that exploits a pseudoinverse-based solution.

In summary, we have demonstrated that the actuator management solution is a scalable and reliable approach for next-generation space missions. In fact, the presented algorithm is currently being implemented for several missions, including an orbit transfer vehicle and a rendezvous and docking mission. Future developments will focus on the incorporation of online methods for calibrating the system matrices dynamically with the aim to enhance the algorithm's adaptability and resilience.

Acknowledgments

Part of this work was performed in the scope of a European Space Agency (ESA) project called "virtual AOCS sensors and actuators for an all-in-one-mode AOCS", contract no. 4000140236/22/NL/MGu. The authors are thankful for the funding thereof.

Declaration of Use of Artificial Intelligence

Artificial intelligence was use to prepare the manuscript in order to improve the readability of the text in terms of language and grammar.

References

[1] Tor A. Johansen and Thor I. Fossen. Control allocation—a survey. *Automatica*, 49(5):1087–1103, 2013. ISSN: 0005-1098. doi: 10.1016/j.automatica.2013.01.035.

[2] Franck Martel. Optimal 6 axis command of a space vehicle with a precomputed thruster selection catalogue table, 2004.

[3] Afonso Botelho, Paulo Rosa, and João Lemos. Explicit spacecraft thruster control allocation with minimum impulse bit. *IEEE Transactions on Control Systems Technology*, PP:1–12, 12 2024. doi: 10.1109/TCST.2024.3511266.

[4] Qiang Shen, Danwei Wang, Senqiang Zhu, and EK Poh. Robust control allocation for spacecraft attitude tracking under actuator faults. *IEEE Transactions on Control Systems Technology*, 25, 2016. doi: 10.1109/TCST.2016.2574763.

[5] Diego Navarro-Tapia, Pedro Simplicio, and Andrés Marcos. Fault-tolerant dynamic allocation strategies for launcher systems. *Aerospace*, 12:393, 04 2025. doi: 10.3390/aerospace12050393.

[6] Danylo Malyuta, Taylor P. Reynolds, Michael Szmuk, Thomas Lew, Riccardo Bonalli, Marco Pavone, and Behçet Açıkmeşe. Convex optimization for trajectory generation: A tutorial on generating dynamically feasible trajectories reliably and efficiently. *IEEE Control Systems Magazine*, 42(5):40–113, 2022. doi: 10.1109/MCS.2022.3187542.

[7] Marc Bodson. Evaluation of optimization methods for control allocation. *Journal of Guidance, Control, and Dynamics*, 25(4):703–711, 2002. doi: 10.2514/2.4937.

[8] Pablo Servidia and Ricardo Sánchez-Peña. Spacecraft thruster control allocation problems. *IEEE Transactions on Automatic Control*, 50:245 – 249, 2005. doi: 10.1109/TAC.2004.841923.

[9] Yuanchao Yang and Zichen Gao. A new method for control allocation of aircraft flight control system. *IEEE Transactions on Automatic Control*, 65(4):1413–1428, 2020. doi: 10.1109/TAC.2019.2918122.

[10] Jesús Ramírez, Joost Veenman, Ilario Cantiello, and Valentin Preda. A generic actuator management solution for space applications based on convex optimization. *Aerospace*, 12(9), 2025. ISSN: 2226-4310. doi: 10.3390/aerospace12090850.

[11] Jesús Ramírez and Lukas Hewing. Sequential convex programming for optimal line of sight steering in agile missions, 2022. doi: 10.48550/arXiv.2206.06061.

[12] Jesús Ramírez, Jorge Cardín, Hector Gutierrez, Francesco Cacciatore, and Valentin Preda. Embedded optimization for space rider reentry module parafoil GNC. *International Astronautical Congress (IAC)*, 2023.

[13] Landis Markley and John Crassidis. *Fundamentals of Spacecraft Attitude Determination and Control*. Springer, 2014. ISBN: ISBN: 978-1-4939-0802-8. doi: 10.1007/978-1-4939-0802-8.

[14] Salvador Llorente-Martinez et al. Performance achievement and verification of unprecedented stability aocs for euclid. In *ESA GNC Conference*, pages 1–16, 2023. doi: 10.5270/esa-gnc-icatt-2023-226.

[15] Joost Veenman, Daniel Serrano-Lombillo, and Raúl Sánchez-Maestro. Robust control design methodology of the 6dof flight-formation of proba-3. In *ESA GNC Conference*, pages 1–11, 2016.

[16] Raúl Sánchez-Maestro, Joost Veenman, Jorge Cardin, Jonathan Grzymisch, and Valentin Preda. Gnc functional architecture design and implementation of the lisa drag-free control system. In *ESA GNC Conference*, pages 1–15, 2023. doi: 10.5270/esa-gnc-icatt-2023-119.

[17] Robin Verschueren, Gianluca Frison, Dimitris Kouzoupis, Jonathan Frey, Niels van Duijkeren, Andrea Zanelli, Branimir Novoselnik, Thivaharan Albin, Rien Quirynen, and Moritz Diehl. acados – a modular open-source framework for fast embedded optimal control. *Mathematical Programming Computation*, 2021.

[18] Michael Grant and Stephen Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. `https://cvxr.com/cvx`, Mar. 2014.

[19] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[20] Alice De Oliveira and Michèle Lavagna. Coupling of Advanced Guidance and Robust Control for the Descent and Precise Landing of Reusable Launchers. *Aerospace*, 11(11):914, Nov. 2024. ISSN: 2226-4310. doi: 10.3390/aerospace11110914.

[21] Jacopo Guadagnini, Pietro Ghignoni, Fabio Spada, Gabriele De Zaiacomo, and Afonso Botelho. End-to-End GNC Solution for Reusable Launch Vehicles. *Aerospace*, 12(4):339, Apr. 2025. ISSN: 2226-4310. doi: 10.3390/aerospace12040339.

[22] Lars Blackmore. Autonomous precision landing of space rockets, 2016.

[23] Marco Sagliano, David Seelbinder, Stephan Theil, and Ping Lu. Six-degree-of-freedom rocket landing optimization via augmented convex–concave decomposition. *Journal of Guidance, Control, and Dynamics*, 47:1–16, 2023. doi: 10.2514/1.G007570.