

Hierarchical Task-Priority Tracking Control in Space Robots

Pietro BRUSCHI^{1,a*} and Davide INVERNIZZI^{1,b}

¹Dipartimento di Scienze e Tecnologie Aerospaziali, Politecnico di Milano, Via La Masa 34,
20156 Milano, Italy

^apietro.bruschi@polimi.it, ^bdavide.invernizzi@polimi.it

Keywords: Hierarchical Control, Space Robotics, GNC

Abstract. This paper extends a previously developed task-priority control algorithm for space robots to higher task dimensions, such that no degrees of freedom remain uncontrolled. The proposed design adopts an inner–outer loop paradigm and leverages a hierarchical formulation of the imposed tasks. This structure ensures modularity, robustness to singularities, and safe task execution in complex space operations.

Introduction

Space robotics has become a vital element of modern space missions, particularly in applications such as In-Orbit Servicing (IOS) and Active Debris Removal (ADR) [1]. These systems typically consist of a manipulator mounted on a satellite base, resulting in kinematic redundancy, where the number of actuated degrees of freedom (DoFs) exceeds those required to control the end-effector's pose. This redundancy provides flexibility to execute tasks in multiple ways—by moving either the base or the manipulator—while optimizing secondary objectives such as dexterity enhancement or joint-limit avoidance.

In many space scenarios, robots must perform multiple tasks simultaneously with different priorities. Redundancy can be exploited using the task-priority control framework, which assigns hierarchical importance to each control goal via nullspace projection [2]. A singularity-robust version of this method was developed in [3] and later generalized to multiple tasks in [4]. Building on this, in [5] we extended the approach to space robotic systems operating on nonlinear configuration manifolds, providing stability guarantees and demonstrating its effectiveness in an orbital capture scenario involving a tumbling target.

The present work advances the methodology introduced in [5] by extending the algorithm to tasks of higher dimension than the system dimension, thus fully exploiting all DoFs. This generalization leverages the methodology presented in [6], which utilizes a projection mechanism that activates a lower-priority task whenever a higher-priority one becomes singular.

The main contribution of this work is the extension of the approach of [6] to the case of space robots. Furthermore, the kinematic task-priority controller is embedded within a hierarchical inner–outer loop architecture that, following [7], ensures almost-global attitude tracking. This modular design enables independent tuning and stability analysis of each layer while maintaining task prioritization, unlike unified structure [6].

Furthermore, the inner loop incorporates a proportional–integral (PI) controller synthesized via H_∞ methods [8], which enhances robustness to spacecraft inertial uncertainties—a key requirement for future space operations.

Notation.

Readers may refer to [9] for a complete description of the spatial notation adopted. We recall here the definition of the pose matrix of a frame j with respect to a frame i , later used in the development of this work:



$${}^iX_j := \begin{bmatrix} {}^iR_j & 0_3 \\ S({}^ip_j){}^iR_j & {}^iR_j \end{bmatrix} := X({}^iR_j, {}^ip_j). \quad (1)$$

The Euclidean norm is denoted by $\|\cdot\|$. We use the symbols $(\cdot)$ and $(\bar{\cdot})$ to indicate errors and desired quantities, respectively. Given a matrix A , $\text{skew}(A) := \frac{A-A^T}{2}$.

The Weighted Moore-Penrose Inverse (WMPI) is compactly written as [6]:

$$A^\dagger = \text{WMPI}(A, W_1, W_2) = R_2^{-1}(R_1 A R_2^{-1})^\dagger R_1, \quad (2)$$

with $W_1 = R_1^T R_1$, and $W_2 = R_2^T R_2$ positive-definite weight matrices.

System Configuration

The scenario considered is the same as that presented in [5] and involves a space robot (a spacecraft equipped with an $N = 7$ -link robotic manipulator) and a second satellite on which servicing operations are to be performed. The latter is referred to as the *target* and the former as the *chaser* throughout this work.

The system configuration is described using multiple Cartesian reference frames to represent the relative motion of the various bodies involved. Figure 1 provides a visual representation of the frames and the transformations along the kinematic chain. Readers may refer to [5] for a more detailed definition and description of the frames.

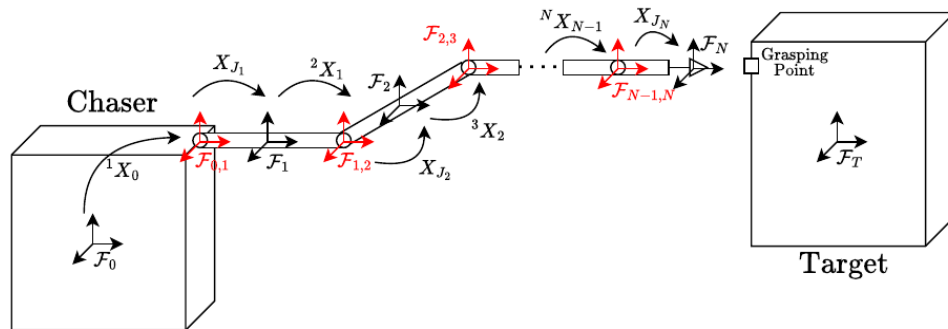


Figure 1: Geometric model of the space robot.

The attitude and position of the space robot base with respect to the target frame $\mathcal{F}_T$ are identified by the rotation matrix $R_0 \in \text{SO}(3)$ and the position vector $p_0 \in \mathbb{R}^3$, respectively. Equivalently, using the spatial notation, the pose of the base frame is denoted $X_0 := X(R_0, p_0) \in \text{SR}(6)$. The manipulator coordinates are described using joint angle variables, namely, as a set of angles collected in a vector $q = (q_1, q_2, \dots, q_N) \in \mathbb{T}^N := \mathbb{S}^1 \times \dots \times \mathbb{S}^1$, where $\mathbb{S}^1$ is identified with $\mathbb{R} \bmod(2\pi)$. These angles specify the relative rotation of the i -th joint frame, $\mathcal{F}_i$, with respect to the link frame $\mathcal{F}_{i-1,i}$. Given $n = 6 + N$ the number of total DoFs of the system, the configuration of the space robot is then uniquely and globally specified by the tuple (X_0, q) , which is an element of the n -dimensional nonlinear manifold $\mathcal{M} := \text{SR}(6) \times \mathbb{T}^N$.

The following system of nonlinear differential equations describes the kinematics of the multi-body system under consideration:

$$\begin{cases} \dot{X}_0 = X_0 v_0^\times \\ \dot{q} = \Omega \end{cases} \quad (3)$$

where $v_0 := ({}^0\omega_{T0}, {}^0v_{T0}) \in \mathbb{R}^6$ is the base twist expressed in $\mathcal{F}_0$, while $\Omega \in \mathbb{R}^N$ represents the vector collecting the angular rates of the joints. For ease of notation, let us define $v := (v_0, \Omega) \in \mathbb{R}^n$ as the system velocity.

The space robot dynamics is then given by the compact expression

$$H(q)\dot{v} + C(v, q)v = u, \quad (4)$$

where $u := (\tau_0, f_0, \tau_q) \in \mathbb{R}^n$ collects the base torque, the base force, and the joint motor torque, respectively, while $H \in \mathbb{R}^{n \times n}$ is the generalized mass matrix and $C \in \mathbb{R}^{n \times n}$ is the Coriolis/centrifugal matrix (See, e.g., [9], for a detailed derivation of these terms).

It is worth reporting the kinematics of the end-effector as well:

$$\dot{X}_N = X_N v_N^\times, \quad (5)$$

with $v_N := ({}^N\omega_{TN}, {}^Nv_{TN}) \in \mathbb{R}^6$ being the twist of the end-effector.

Multi-task priority control law design

Given the system (1)-(2), and a predefined number of tasks r , we remove the implicit assumption of [5] that the total task dimension $m = \sum_{k=1}^r m_k$, namely the overall number of DoFs requested by the tasks, is at most equal to the number of DoFs of the system n . Instead, we focus in the opposite case: we assume that the total task dimension is not less than the DoFs of the robot, i.e., $m \geq n$, as done in [6].

As remarked in the introduction, the controller developed in this work relies on an inner–outer loop paradigm, where the outer loop commands to the inner one a virtual velocity input.

The commanded virtual velocity is determined based on the following process: a set of tasks is first established, after which the procedure outlined in [6] is applied. For each task, the dynamically consistent Jacobian matrix is computed starting from the actual Jacobian. According to the following routine:

$$\begin{aligned} \hat{J}_k &= J_k P_{k-1} \\ P_k &= P_{k-1} (I_N - \hat{J}_k^\dagger \hat{J}_k) \end{aligned} \quad (6)$$

where $k = 1, \dots, r$ and $\hat{J}_k^\dagger = \text{WMPI}(\hat{J}_k, W_{1k}, W_2)$. The weight $W_{1k} = I_{m_k} \in \mathbb{R}^{m_k \times m_k}$ is chosen as an identity matrix of appropriate dimension, while $W_2 = H(q)$. Moreover, $P_0 = I_n$.

Using this definition, the Jacobian matrix of the first task remains unchanged. The recursive use of nullspace projectors will ensure that tasks are decoupled, since each lower-priority task will be applied only in directions that do not interfere with higher-priority tasks.

The pseudoinverses of the dynamically consistent Jacobian matrices may be compactly collected in a stack as well:

$$\hat{J}^\dagger = [\hat{J}_1^\dagger \quad \dots \quad \hat{J}_r^\dagger]. \quad (7)$$

The control law of the outer loop can now be explicitly stated:

$$v^c = \hat{J}^\dagger \zeta. \quad (8)$$

The stack $\zeta = (\zeta_1 \dots \zeta_r) \in \mathbb{R}^m$ collects the commanded tasks' velocities, which can be designed using different stabilizers, according to specific needs.

Then, the inner-loop controller computes the control input u to track the generalized velocity $\mathbf{v}^c \in \mathbb{R}^n$, provided by the outer loop to achieve the task objectives, according to the following law:

$$u = K_p \tilde{\mathbf{v}} + K_I x_I + H(q) \dot{\mathbf{v}}^c + C(q, \dot{q}) \mathbf{v}^c, \quad (9)$$

where $\tilde{\mathbf{v}} = \mathbf{v} - \mathbf{v}^c \in \mathbb{R}^n$ is the control error, and $x_I \in \mathbb{R}^n$ is the integral state, defined such that $\dot{x}_I = \tilde{\mathbf{v}}$, which provides integral action on the twist error. The matrices $K_p \in \mathbb{R}^{n \times n}$ and $K_I \in \mathbb{R}^{n \times n}$ are gain matrices tuned using robust control techniques [8].

Numerical Results

In this section, we present the same scenario considered in [5], where a space robot must perform an orbital capture on a tumbling target. The control objectives are translated into two tasks: tracking the pose (position and attitude) of the end-effector and tracking the pose of the base spacecraft. In this work, we have added a third task, representing a safe joint configuration $\bar{q}$ [6]. Such a quantity is a set of joint angles that the arm has to track whenever the upper tasks become singular, *i.e.*, the corresponding Jacobian loses rank.

The three tasks considered can be specified as functions of the system state:

$$\begin{aligned} \xi &= (X_N, X_0, q) \\ \zeta &= (v_N^c, v_0^c, \Omega^c) \end{aligned} \quad (10)$$

where $\xi \in \mathbb{R}^m$ is the task specification vector.

Table 1: Selected Tasks.

Task #	Description	Jacobian	# of DoFs
1	Tracking EE's pose: X_N	$J_1 = [J_b \quad J_m]$	$m_1 = 6$
2	Tracking base pose: X_0	$J_2 = [I_6 \quad 0_{6 \times 7}]$	$m_2 = 6$
3	Safe joint config.: q	$J_3 = [0_{7 \times 6} \quad I_7]$	$m_3 = 7$

The three tasks, along with their Jacobians, are reported in Table 1.

The end-effector Jacobian J_1 is defined by

$$\begin{aligned} J_b &:= {}^N X_0 \\ J_m &:= [\cdots {}^N X_i X_{J_i}(q_i) s_r \cdots] \end{aligned} \quad (11)$$

with $s_r := [0 \ 0 \ 1 \ 0 \ 0 \ 0]^T \in \mathbb{R}^6$ being the motion constraint of the revolute joint.

Once the tasks have been specified, the problem may be solved by selecting the following commands:

$$\begin{aligned} v_N^c &= \gamma^X(\tilde{X}_N) + \tilde{X}_N^{-1} \tilde{v}_N \\ v_0^c &= \gamma^0(\tilde{X}_0) + \tilde{X}_0^{-1} \tilde{v}_0 \\ \Omega^c &= -K_q \tilde{q} \end{aligned}$$

The outer loop stabilizer is selected as in [5]. For $\kappa = 0, N$:

$$\gamma^\kappa = (\tilde{e}_\kappa, -\tilde{R}_\kappa K_\kappa^p \tilde{p}_\kappa),$$

where $\tilde{e}_\kappa = -S^{-1}(\text{skew}(K_\kappa^R \tilde{R}_\kappa))$ is the attitude error, and $K_\kappa^p \in \mathbb{R}^{3 \times 3} > 0$, $K_\kappa^R \in \mathbb{R}^{3 \times 3}$ and $K_q \in \mathbb{R}^{N \times N} > 0$ are symmetric gain matrices.

In the scenario considered, a reference trajectory is commanded to the end-effector to follow the grasping point on the target and perform an orbital capture. The base spacecraft is commanded to follow such a grasping point. The third task is commanded as a safe pose different from the one naturally obtained with the first two tasks active.

Two representative simulations are reported next.

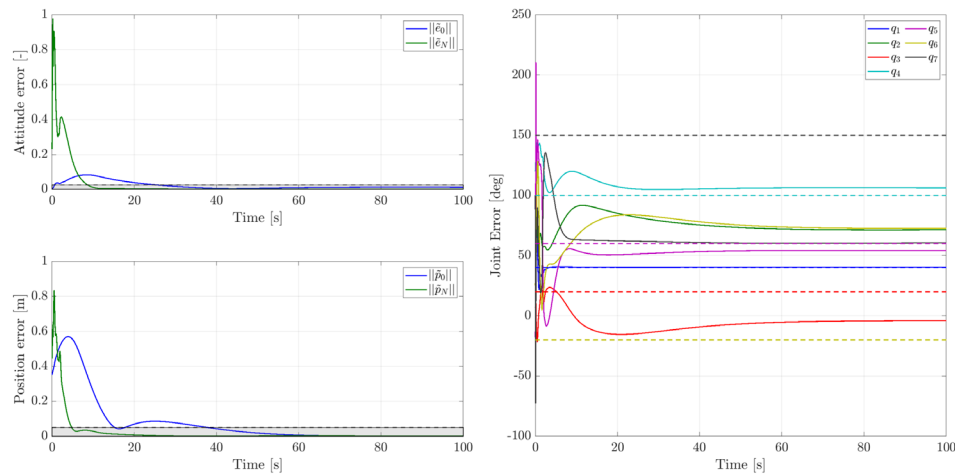


Figure 2: Pose errors of base and end-effector, and joint angles – non-singular ICs.

In the first simulation, the first two tasks are performed, as in both cases the position error norm and the attitude error norm converge below the threshold error (gray area in (Figure 2-left)) while the joint angles do not reach the safe configuration (dashed lines), except for the first joint q_1 , that represents the only DoF not used by previous tasks (Figure 2-right).

In the second simulation (Figure 3), the initial conditions (ICs) of the manipulator are modified to make it start the maneuver in a singular configuration. In this way, the third task comes into action and the joints that result in critical positions (q_3 and q_5) reach the safe configuration. Then, the capture is performed nominally, after successfully exiting the singularity. The gray area in the joints plot marks the instant where the capture actually begins.

Conclusions

This paper has extended the hierarchical task-priority algorithm proposed in our previous work [5] to scenarios where the task dimension exceeds the DoFs of the system, while preserving modularity and robustness to kinematic singularities. Simulation results have proven the effectiveness of the proposed control scheme. By leveraging the safe pose at the end of the tasks stack, the system manages to escape from the singularity condition and subsequently performs the capture.

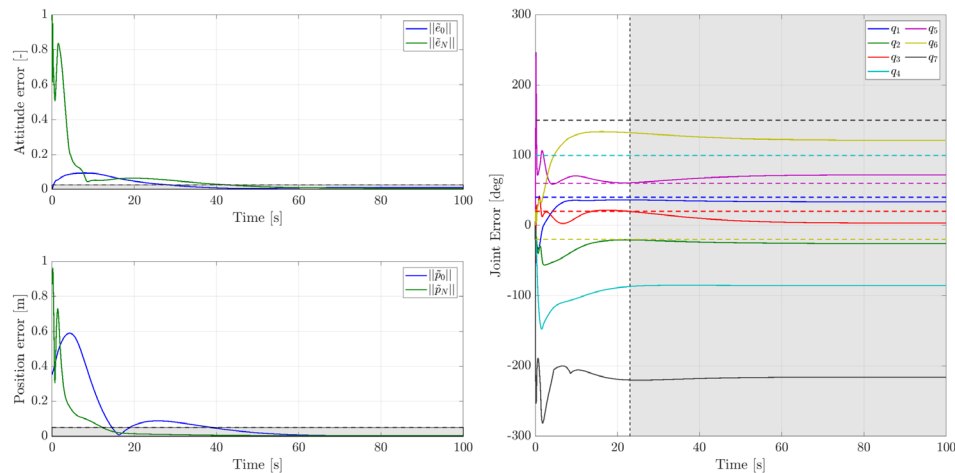


Figure 3: Pose errors of base and end-effector, and joint angles – singular ICs.

References

- [1] P. Tsiotras, M. King-Smith, and L. Ticozzi, Spacecraft-mounted robotics, Annual Review of Control, Robotics, and Autonomous Systems, vol. 6, 2023. <https://doi.org/10.1146/annurev-control-062122-082114>
- [2] Y. Nakamura, H. Hanafusa, and T. Yoshikawa, Task-Priority Based Redundancy Control of Robot Manipulators. The International Journal of Robotics Research, vol. 6, no. 2, pp 3-15, 1987. <https://doi.org/10.1177/027836498700600201>
- [3] S. Chiaverini, Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators, IEEE Transactions on Robotics and Automation, vol. 13, no. 3, pp. 398-410, 1997. <https://doi.org/10.1109/70.585902>
- [4] G. Antonelli, Stability Analysis for Prioritized Closed-Loop Inverse Kinematic Algorithms for Redundant Robotic Systems, IEEE Transactions on Robotics, vol. 25, no. 5, pp. 985-994, Oct. 2009. <https://doi.org/10.1109/TRO.2009.2017135>
- [5] P. Bruschi and D. Invernizzi, Singularity-free task-priority design for trajectory tracking in space robots, 2025 American Control Conference (ACC), Denver, USA, 2025, pp. 3825-3830. <https://doi.org/10.23919/ACC63710.2025.11107894>
- [6] G. Garofalo and C. Ott, Hierarchical tracking control with arbitrary task dimensions: Application to trajectory tracking on submanifolds, IEEE Robot. Autom. Lett., vol. 5, no. 4, pp. 6153-6160, 2020. <https://doi.org/10.1109/LRA.2020.3010449>
- [7] D. Invernizzi, M. Lovera and L. Zaccarian, Hierarchical dynamic control for robust attitude tracking, IFAC-PapersOnLine, vol. 53, no. 2, pp. 6171-6176, 2020. <https://doi.org/10.1016/j.ifacol.2020.12.1701>
- [8] Apkarian, P., Gahinet, P., and Buhr, C., Multi-model, multi-objective tuning of fixed-structure controllers, 2014 European Control Conference (ECC), IEEE, Strasbourg, France, 2014, pp. 856-861. <https://doi.org/10.1109/ECC.2014.6862200>
- [9] R. Featherstone, Rigid body dynamics algorithms. New York : Springer, 2008. <https://doi.org/10.1007/978-1-4899-7560-7>