Acta Astronautica 216 (2024) 504–517

Contents lists available at ScienceDirect

Acta Astronautica

journal homepage: www.elsevier.com/locate/actaastro

Research paper

Finite-time state-dependent Riccati equation regulation of anthropomorphic dual-arm space manipulator system in free-flying conditions

Alessandro Scalvini [*], Alejandro Suarez, Saeed Rafee Nekoo, Anibal Ollero

The GRVC Robotics Laboratory, Escuela Técnica Superior de Ingeniería, Universidad de Sevilla, Camino de los Descubrimientos s/n, Seville, 41092, Spain

ARTICLE INFO

Keywords:
Dual-arm space manipulator
SDRE
State-dependent coefficient online computation
Optimal control
Space robotics
Nonlinear control

ABSTRACT

This paper introduces a novel approach for regulating the pose of a free-flying dual-arm anthropomorphic space manipulator system (SMS) using a finite-time state-dependent Riccati equation (SDRE) controller. The proposed system finds applications in on-orbit satellite inspection, servicing, space structure assembly, and debris manipulation. The dual-arm SMS presented in this work consists of two 7 degrees of freedom (DoF) robotic arms mounted on a free-flying spacecraft, resulting in a complex 20-DoF system. Due to the high number of DoFs, advanced controller design and efficient computations are necessary. The finite-time SDRE controller relies on the state-dependent coefficient (SDC) parameterization matrices, which are nonlinear apparent linearizations of the dynamics. Conventionally, the computation of SDC matrices is offline and relies on the a priori derivation of the analytical equations governing the dynamics of the system. However, this strategy becomes computationally impractical for high DoF plants. To overcome this issue and deliver a more viable solution, a numerical method to construct and update the SDC matrices at each time step is presented. This approach relies on a screw-theory-based recursive Newton–Euler algorithm designed to reconstruct the manipulator inertia and Coriolis matrices. These quantities are the building blocks of the SDC parameters used in the synthesis of the SDRE controller. Simulation results demonstrate the performances of the finite-time SDRE controller augmented with the online update of the state-dependent coefficients.

1. Introduction

On-orbit operations present multiple risks and difficulties due to the inherent challenges of working in a harsh space environment. However, a sustainable program of space exploration and exploitation relies on the capability of performing complex tasks such as servicing [1,2], assembling [3], debris removal [4], and repairing directly in orbit [5]. Nowadays, a large number of the most complex operations are conducted by humans during spacewalks. These are hours-long, dangerous, and physically demanding missions in which the astronaut and the environment are separated only by a (relatively) thin space suite. In fact, the absence of gravity, extreme temperature fluctuations, vacuum, radiation, and micro-meteoroids pose significant risks for humans. Because of the mentioned reasons, resorting to space robots to perform Extra Vehicular Activities (EVA) is particularly appealing since they are potentially capable of effectively operating in this hostile environment without endangering human lives.

In order to replicate the dexterous manipulation capabilities of human astronauts, it is considered the application of anthropomorphic dual-arm space manipulator systems (SMS) for the realization of maintenance operations on tumbling or cooperative satellites, as illustrated in Fig. 1.

Motivated by the necessity of helping and gradually replacing humans in the most dangerous operations, the field of space robotics has been evolving since the Shuttle Remote Manipulator System, known as Canadarm I, was first introduced in 1981. Remarkable examples of the advancements in this field include the first on-orbit operation conducted by the ETS-VII [6], the Space Station Remote Manipulator System, also known as Canadarm II, and the Special Purpose Dextrous Manipulator. These last are extensively employed on the International Space Station. An additional step forward consists of employing a Space Manipulator System (SMS) capable of conducting On-Orbit Servicing (OOS) missions with limited (teleoperation) or without (fully autonomous) human supervision [7]. As a matter of fact, SMS can be remotely operated to perform challenging tasks such as replacing faulty components/batteries and assisting the astronauts during spacewalks [8]. Direct telemanipulation, achieved utilizing virtual reality and telepresence techniques (visual and haptic feedback) augments the capability of teleoperation systems by providing a more immersive feeling for the operator [9]. Additionally, advanced rendezvous [10] and docking systems [11] allow the servicing spacecraft to approach

* Corresponding author.
E-mail addresses: ascalvini@us.es (A. Scalvini), asuarezfm@us.es (A. Suarez), saerafee@yahoo.com (S.R. Nekoo), aollero@us.es (A. Ollero).

https://doi.org/10.1016/j.actaastro.2024.01.027
Received 26 September 2023; Received in revised form 26 December 2023; Accepted 18 January 2024
Available online 23 January 2024

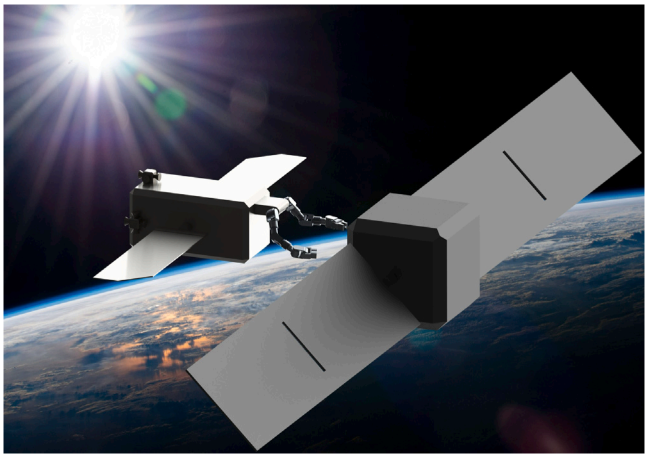


Fig. 1. Render of the Dual arm Space Manipulator System performing a maintenance task. Credits for background image: [16].

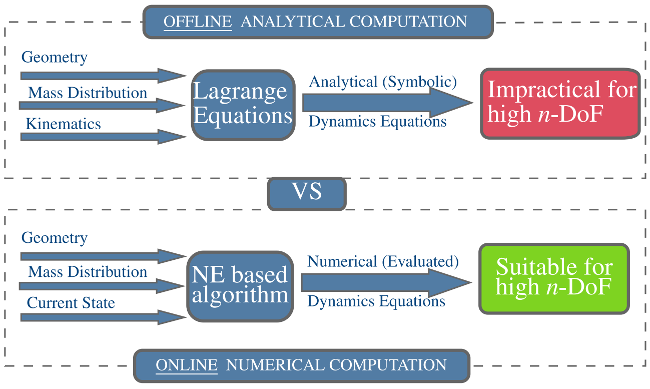


Fig. 2. When the system is characterized by a large number of DoF, the analytical form of the dynamics equations results in considerably long and complex symbolic expressions which can make it impractical to convert them in a useful form (as a function handles). The solution adopted is to implement a Newton–Euler (NE) based algorithm to reconstruct the mass and Coriolis matrices at each time instant. The quantities have been transformed into numerical values, making them considerably more manageable.

and birth to the target satellite, ensuring a secure connection for maintenance operations. With these current solutions, on-orbit servicing is poised to revolutionize space operations by extending the lifespan of satellites [12] and reducing space debris [13], thus improving the sustainability of space missions cost-effectively. The most recent technological advancements in this direction have been extensively reviewed in [1,14,15]. A common trend that can be identified is the necessity to develop a highly agile and dexterous space manipulator system that is capable of performing a wide range of tasks in the different phases of the OOS summarized in [1], namely: (i) rendezvous and closing maneuver, (ii) target identification, (iii) attitude synchronization, (iv) manipulator deployment, (v) capture and (iv) post-capture maneuvers.

In this paper, specific attention will be devoted to the coordinated base-arms motion taking place after the target identification. The base and arms adopt a convenient configuration to grasp the target while ensuring it remains within reach of the end effectors. Thus, in this work, the system is required to change both the attitude and position of the base while deploying the dual-arm manipulator in order to reach an overall pose more suitable for capture. Common strategies used for similar scenarios include the Generalized Jacobian [17] and the Virtual Manipulator [18] to solve the inverse kinematics of the SMS (out of the scope of this paper) in order to obtain a reference for the control. Relatively recent techniques include Optimal control, Robust control [19], and Sliding Mode control [20]. The former tends to be excessively computationally demanding and the latter is likely to produce chattering [1].

What is proposed in this paper is a finite-time SDRE augmented with an online update of the state and input matrices. This differs from the prevalent method reported in the literature. Conventional strategies involve using symbolic computation tools for the offline derivation of the dynamics parameters to a priori assemble the analytical state-dependent coefficients. The SDC parameterization is then numerically evaluated every time the gain matrix of the SDRE is computed. While the standard approach is well-suited for systems with a limited number of Degrees of Freedom (DoF) [21], it becomes exceedingly demanding as the system's DoFs increase ($n \geq 10$). The limitation arises from the length and complexity of the mass and Coriolis matrices entries, which results from the inherently strong nonlinear dynamics exhibited by space robots. This is summarized in Fig. 2. Another distinctive feature of this work, setting it apart from similar studies, lies in the modeling of the base as "free-flying" rather than "free-floating". Previous works considered the model reference adaptive SDRE and output feedback SDRE to address the control of space manipulators in point-to-point and trajectory tracking scenarios [22–24]. However, the authors restricted the spacecraft's motion to free-floating conditions. Furthermore, the model employed in the existing literature on SDRE and space robotics often makes use of strong simplifications to reduce the DoFs. Typically, the case study is limited to planar scenarios as in [23,24]. In this paper, a more complex 20-degree-of-freedom system with a fully actuated free-flying base is considered instead. This additional complexity is reflected in the resulting equations of motion, which become considerably large and difficult to manipulate. As a consequence, the derivation of state-dependent coefficient matrices becomes quite challenging, highlighting once more the contribution of this paper. Consequently, this work demonstrates the potential of this nonlinear method for synthesizing the control of complex nonlinear plants while maintaining asymptotic stability. Even though the SDRE belongs to the Optimal control category, it has the great advantage of leading to a closed-loop solution to the optimal control [25–27]. The solution solves the differential Riccati equation (DRE) which can be reliably evaluated (even online) and thus the control policy can be readily retrieved without actually solving an optimization problem. There are several advantages to using this controller. Firstly, it has local asymptotic stability (global for some cases) based on the Lyapunov theorem and it is optimal in nature [25,28]. Moreover, the numerical solution to the DRE can be reliably computed while the solver might not be able to find a solution to Optimal and Robust control problems, especially when the optimization is non-convex. Another advantage of using this controller is the relative ease of enforcing high-level constraints. This can be done by operating on the weighting matrices of the controller to tune its performances [29]. In conclusion, it is demonstrated the SDRE can be a valuable and versatile tool for controlling complex and highly non-linear systems such as SMSs.

The main contributions of this paper can be summarized as follows:

1. Application of the algorithm for reconstructing the dynamics of free-flying systems, as presented in the Ref. [30], to a robot with multiple connected open chains.
   Implementation of a fully numerical method based on the Newton–Euler algorithm for the online update of the state-dependent coefficient matrices. The usual approach based on the offline analytical derivation of the SDC would fail due to the complexity of the system equations of motion. The considerable amount of highly non-linear terms would greatly impair their manipulation, thus making the assembly of the SDC matrices inconvenient. By instead constructing them online in their numerical form, the problem is avoided entirely.

Table 1
Notation table.

GENERIC SYMBOLS		
Symbol	**Domain**	**Definition**
$\tilde{(\cdot)}$	[–]	Estimated quantity
$\dot{(\cdot)}$	[–]	First time derivative
$\ddot{(\cdot)}$	[–]	Second time derivative
$\hat{(\cdot)}$	[–]	Lie wedge operator
$(\cdot)^{\dagger}$	[–]	Right pseudoinverse
$\{\cdot\}$	[–]	Reference frame
i	$\mathbb{N}$	Index of i-DoF/link
j	$\mathbb{N}$	Index jth sub-chain
n	$\mathbb{N}$	Number of DoF
t	$\mathbb{R}^{+}$	Time
ROBOT MECHANICS		
Symbol	**Domain**	**Definition**
q_i^j	$\mathbb{R}$	ith DoF $\in$ jth sub-chain
$\mathbf{c}(\mathbf{q}(t),\dot{\mathbf{q}}(t))$	$\mathbb{R}^{n}$	Coriolis and centrifugal forces vector
$C(\mathbf{q}(t),\dot{\mathbf{q}}(t))$	$\mathbb{R}^{n\times n}$	Coriolis and centrifugal forces matrix
$M(\mathbf{q}(t))$	$\mathbb{R}^{n\times n}$	Manipulator inertia matrix (also 'mass matrix')
$\mathbf{q}(t)$	$\mathbb{R}^{n}$	Vector of the generalized DoF (joint variables)
${}^{\{A\}}\mathbf{r}_{\{B\}}$	$\mathbb{R}^{3}$	Position of $\{B\}$ with respect to $\{A\}$
${}^{\{A\}}\mathbf{R}_{\{B\}}$	$\mathbb{R}^{3\times 3}$	Orientation of $\{B\}$ with respect to $\{A\}$
${}^{\{A\}}\mathbf{T}_{\{B\}}$	$\mathbb{R}^{4\times 4}$	Pose of $\{B\}$ with respect to $\{A\}$
$\xi_i(t)$	$\mathbb{R}^{6}$	Twist coordinates of ith joint/link
$\tau(t)$	$\mathbb{R}^{n}$	Vector of the generalized forces
CONTROL THEORY		
Symbol	**Domain**	**Definition**
$A(\mathbf{x}(t))$	$\mathbb{R}^{2n\times 2n}$	State dependent coefficient (state-related)
$B(\mathbf{x}(t))$	$\mathbb{R}^{2n\times n}$	State dependent coefficient (input-related)
F	$\mathbb{R}^{2n\times 2n}$	Final state weight
$X(\mathbf{x}(t))$	$\mathbb{R}^{m\times 2n}$	SDRE gain matrix
$Q(\mathbf{x}(t))$	$\mathbb{R}^{2n\times 2n}$	State trajectory weight
$R(\mathbf{x}(t))$	$\mathbb{R}^{n\times n}$	Input trajectory weight
$\mathbf{u}(t)$	$\mathbb{R}^{n}$	Control input vector
$\mathbf{x}(t)$	$\mathbb{R}^{2n}$	State vector: $\mathbf{x}(t) := \left[\mathbf{q}^{\top}(t),\dot{\mathbf{q}}^{\top}(t)\right]^{\top}$

2. Implementation of the finite-time SDRE to a 20 Degree of Freedom (DoF), anthropomorphic dual-arm SMS in free-flying mode for coordinated base-arms motion control.

The rest of the paper is organized as follows. Section 2 presents the system modeling and the algorithm to retrieve the dynamic model and Section 3 describes the design of the SDRE controller used in this paper and its synthesis. Section 4 provides implementation details of the simulated system and the controller, whereas Section 5 presents the simulation results. Finally, the conclusions of this work are discussed in Section 6.

2. System modeling

In this section, the adopted notation and main assumptions of the dual arm space manipulator model are presented. After an overview of the system, screw theory is used to formulate the kinematics and the dynamics of the robot since it scales well with the complexity of the model, especially when compared to the Denavit–Hartemberg standard parameterization [31]. Then, the algorithm used to retrieve the manipulator inertia and Coriolis matrices is reported and explained for completeness.

2.1. Notation

Table 1 provides a concise summary of the most commonly used symbols. Due to the formal notation, many of these symbols' definitions will be reiterated for clarity.

2.2. System overview

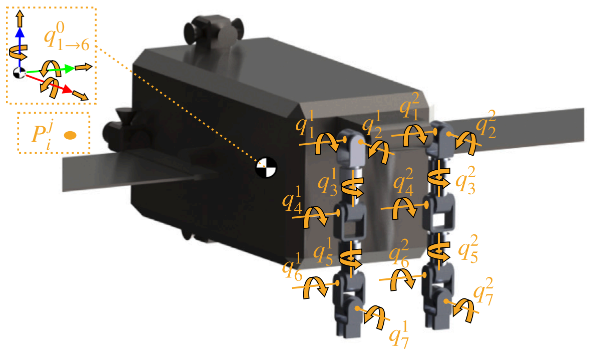
The first critical assumption pertains to the spacecraft operating in a free-flying mode in the microgravity environment of space. Consequently, in accordance to the classification proposed [32], the SMS is capable of employing its actuators for the simultaneous coordinated control of both attitude and position. Consider now the dual arm SMS represented in Fig. 3. The space robot consists of two identical 7-DoF arms with anthropomorphic kinematics and a free-flying 6-DoF base, resulting in a total of 20-DoF system. In particular, the anthropomorphic kinematics of the lightweight dual arm developed in our previous work for aerial manipulation [33] is implemented in this paper as well.

It consists of three joints at the shoulder, one at the elbow and three at the wrist, obtaining the following overall configuration:

1. Shoulder flexion/extension.
2. Shoulder adduction/abduction.
3. Medial/lateral rotation.
4. Elbow flexion/extension.
5. Wrist adduction/abduction.
6. Wrist pronation/supination.
7. Wrist flexion/extension.

All the parts are assumed to be rigid bodies and all the DoFs can be actuated independently. The mass of the joints is assigned to the links and the latter are assumed to have uniform mass distribution (uniform density). The SMS is displayed in two configurations: Fig. 3(a) shows the home configuration in which all the joint variables are set to zero. This configuration is chosen for convenience in the derivation, as outlined in [31,34]. When the SMS adopts this pose, the center of mass (CoM) of the base is coincident with the origin of the inertial frame $\{N\}$ since the variables associated with the base DoFs (q_i^0 for $i = 1,\dots,6$) are considered part of the overall kinematic chain. Thus, they are null at home configuration. Fig. 3(b) shows instead a generic pose of the system. Here, the most important frames can be observed: the inertial frame $\{N\}$, the SMS-base frame $\{0\}$ and the arms-base frames $\{1^1\}$ and $\{1^2\}$. The right superscript 'j' indicates the sub-chain index. The quantities related to the base are indicated with $j = 0$, while those belonging to the right and left arm are indicated with $j = 1$ and $j = 2$, respectively. Consequently, with q_i^j, is indicated the ith DoF of the jth sub-chain, and with P_i^j is indicated the point on the ith joint axis of the jth sub-chain. Recall that the axes and the points lying on them are the building blocks used to derive the kinematics using screw theory [31,34]. Finally, Fig. 3(b) illustrates the position vectors of the origins of specific frames (each attached to a distinct component of the manipulator). These frames are configured with their origins coinciding with the CoM of the respective link and are aligned with the principal axes of the corresponding link. The vector pointing form the origin of frame $\{A\}$ to the origin of frame $\{B\}$ expressed in $\{A\}$ is represented with ${}^{\{A\}}\mathbf{r}_{\{B\}} \in \mathbb{R}^3$. Similarly, ${}^{\{A\}}\mathbf{R}_{\{B\}} \in \mathbb{R}^{3\times 3}$ indicates the rotation matrix expressing the orientation of frame $\{B\}$ with respect to $\{A\}$. Recalling that base DoFs are considered part of the chain, as clarified by Fig. 4, the common chain connecting $\{N\}$ to $\{0\}$ can be interpreted as composed of six links and six joints. The first five links of the chain are virtual[1] while the last is simply the SMS-base itself. From now on, a system with 'hybrid dynamics' and 'tree-like' kinematics chain will be referred to as an Equivalent Manipulator (EM) model. Note that this interpretation is convenient in the development of ground testing systems that employ robotic manipulators for simulating the 6-DoF motion of a space platform [14,35]. In the EM, the 6-DoF joint connecting the base with the inertial frame is imagined as a series of three virtual prismatic (P) joints followed by three virtual revolute (R) joints. Notice that using the 3R joints to describe the orientation is

[1] By 'virtual' it is meant that the component is point-like and mass-less.



(a)

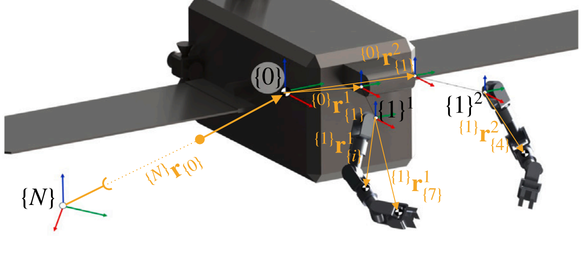

$\{0\}$
${}^{\{0\}}\mathbf{r}^2_{\{1\}}$
${}^{\{0\}}\mathbf{r}^1_{\{1\}}$
$\{1\}^1$
$\{1\}^2$
${}^{\{1\}}\mathbf{r}^1_{\{i\}}$
${}^{\{1\}}\mathbf{r}^2_{\{4\}}$
${}^{\{1\}}\mathbf{r}^1_{\{7\}}$
$\{N\}$
${}^{\{N\}}\mathbf{r}_{\{0\}}$
(b)

Fig. 3. (a) Home Configuration. Notice in this configuration the base frame $\{0\}$ and inertial frame $\{N\}$ coincide since all the joint variables are null. The ith DoF belonging to sub-chain j is indicated with q^j_i. (b) Generic Configuration. With ${}^{\{A\}}\mathbf{r}^j_{\{B\}}$ is intended the distance of the origin of the link-frame $\{B\}$ from $\{A\}$ (expressed in $\{A\}$). Furthermore, both link-frames belong to the jth sub-chain. The arms are detached from the base only for clarity of visualization.

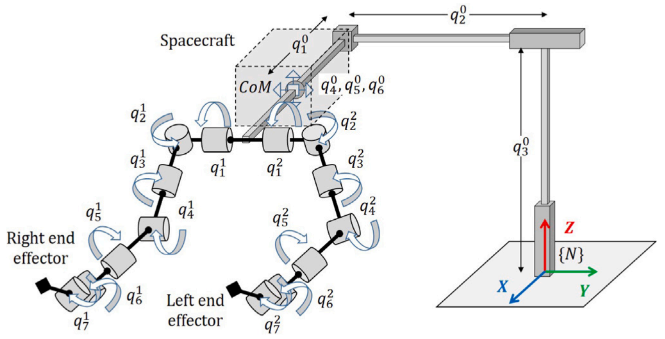


Fig. 4. Kinematic configuration of the anthropomorhpic dual arm space manipulator. The Equivalent Manipulator model employ 6 virtual joints to represent the position (q^0_1, q^0_2, q^0_3) and attitude (q^0_4, q^0_5, q^0_6) of the spacecraft with respect to the Inertial Frame $\{N\}$.

completely equivalent to the $\{X, Y, Z\}$ Euler angles parameterization of the rigid body attitude. Connecting the 3P joint in series with the 3R, as depicted in Fig. 4, the free-flying joint (F) is obtained. The main benefit of this interpretation is that it allows treating the base as six links connected by six joints of a fixed-base open-chain manipulator whose (virtual) base is attached to the inertial frame $\{N\}$. Consequently, the screw theory version of the Newton–Euler algorithm presented in [34] (for fixed-based manipulator) and [30] (for a hybrid dynamics system with single open chain) can be readily extended to our dual-arm free-flying space manipulator (see Section 2.4). Finally, it is important to highlight that all effects resulting from Orbital Mechanics are neglected in this study. Consequently, the reference frame denoted as $\{N\}$ is assumed to be inertial. However, it is important to acknowledge that, given the intended operation in Low Earth Orbit (LEO) or Geostationary Orbit (GEO), non-inertial effects are indeed of significance. In fact, the origin of reference frame $\{N\}$ would actually coincide with the one of Local Vertical Local Horizontal ($\{LVLH\}$) frame of the target satellite, which is intrinsically non-inertial [36]. The mentioned assumption is however grounded in the particular phase of rendezvous under consideration in this work. In fact, during the terminal phase of rendezvous leading to inspection or docking, the spacecraft trajectory is fully controlled, to the point of being completely arbitrary. In some cases, the trajectory of the chaser (the SMS) can even appear linear in the $\{LVLH\}$ of the target, e.g. V-bar approach [36]. This motion is achieved by momentarily counteracting the influences of orbital mechanics through the active propulsion of thrusters[2]. In conclusion, a thorough treatment of the mentioned non-inertial effects is out of the scope of this paper and they are thus neglected by assuming to be able to effectively counteract them as proposed in the literature, see for instance [36–38].

2.3. Kinematics and dynamics

To derive the pose of each relevant mass a screw theory-based formulation of the kinematics is used. Despite the initial effort needed to understand the mathematical formalism, it leads to a quite simple and mechanical procedure [31].

Firstly, the system is placed in a convenient home configuration (see Fig. 3(a)) where each variable associated with a DoF is set to zero. This is a pose that facilitates the identification of the position of strategic points, axes, and orientations of frames with respect to the inertial one $\{N\}$. Namely, the position of the CoM of each link, the direction of joint axes, the coordinates of a point lying in them, and the orientation of each link principal axes (PA) are computed with respect to $\{N\}$. Since the mentioned quantities are all expressed with respect to the inertial frame $\{N\}$ only, a proper choice of home configuration greatly simplifies their identification. Once these elements are available, the homogeneous transformation matrix (HTM) ${}^{\{N\}}\mathbf{T}_{\{K\}}(t) \in \mathbb{R}^{4\times4}$, describing the pose of any frame of interest $\{K\}$ with respect to $\{N\}$ can be systematically derived by applying the product of exponential formula:

$${}^{\{N\}}\mathbf{T}_{\{K\}}(t) = \prod_{i=1}^{k} \left[e^{\hat{\xi}_i(t) q_i(t)} \right] {}^{\{N\}}\mathbf{T}_{\{K\}}(0). \tag{1}$$

A remarkable result presented in [31] reveals that there is no need to compute the Lagrange equations to derive the second-order Equation of Motion (EoM) for any system displaying a Lagrangian with the following structure: where $e^{(\cdot)}$ is the exponential matrix, $\hat{\xi}_i(t) \in \mathbb{R}^{4\times4}$ is the twist associated to the ith joint and $q_i(t)$ is the corresponding joint variable. The operator $\hat{(\cdot)}$ stands for the Lie-algebra ‘wedge’ [31] that maps the twist coordinates, which are packed into a six-entries vector $\xi_i(t) \in \mathbb{R}^6$, to the twist element of the set $se(3)$ itself [31]. The pose of $\{K\}$ relative to $\{N\}$ when the system is at home configuration is represented by the HTM ${}^{\{N\}}\mathbf{T}_{\{K\}}(0) \in \mathbb{R}^{4\times4}$. The index k represents the last joint influencing the forward kinematics of frame $\{K\}$, which is a generic frame whose pose is of interest (usually the frame attached to the end effector). Thus, index k indicates the joint immediately preceding frame $\{K\}$ in the kinematic tree. In order to increase readability, the superscript j is omitted in the above equation. However,

[2] It is important to emphasize that this operational mode can only be sustained for a very limited duration, owing to its substantial fuel consumption and it is reserved for final stages of the rendezvous process, a scenario that precisely aligns with the focus of this paper.

it is important to notice that the product of the exponential formula must be applied for each sub-chain $j = 0, 1$ and 2 independently. A possible optimization could be storing the pose of base sub-chain $j = 0$ (${}^{\{N\}}\mathbf{T}_{\{6\}}$) since it is common to the following DoFs. Indeed the base DoFs influence both arms (thus kinematic chains $j = 1$ and $j = 2$), but the forward kinematics of one arm is independent of the joint variables of the other. The kinematics of the EM is the starting point for deriving the dynamics since, by applying Eq. (1) for each link and end-effector, the time history of the pose of each relevant mass is obtained. This is needed to compute the kinetic energy of the bodies and hence the Lagrangian.

$$L = \frac{1}{2}\dot{\mathbf{q}}(t)^{\top}\mathbf{M}(\mathbf{q}(t))\dot{\mathbf{q}}(t) - V(\mathbf{q}(t)), \tag{2}$$

where $\mathbf{q}(t) \in \mathbb{R}^n$ is the generalized coordinate vector which collects the n joint variables, $\dot{\mathbf{q}}(t)$ its first time derivative, $V(\mathbf{q}(t)) \in \mathbb{R}$ is the potential energy. Since the operation is conducted in a micro-gravity environment, it is possible to immediately set $V(\mathbf{q}(t)) = 0$. The quantity $\mathbf{M}(\mathbf{q}(t)) \in \mathbb{R}^{n\times n}$ is the symmetric positive definite manipulator inertia matrix defined as:

$$\mathbf{M}(\mathbf{q}(t)) = \sum_{i=1}^{n} \mathbf{J}_i^b(\mathbf{q}(t))^{\top} \boldsymbol{\mathcal{M}}_i \, \mathbf{J}_i^b(\mathbf{q}(t)). \tag{3}$$

The terms of Eq. (3) are the generalized inertia matrix $\boldsymbol{\mathcal{M}}_i \in \mathbb{R}^{6\times 6}$ of the ith rigid body of the robot and $\mathbf{J}_i^b(\mathbf{q}(t)) \in \mathbb{R}^{n\times 6}$ is the corresponding body manipulator Jacobian [31]. The role played by this quantity is to 'rotate' the matrix $\boldsymbol{\mathcal{M}}_i$, which is conveniently expressed in the PA frame of the ith rigid body, to the current configuration, thus taking into account the information provided by the forward kinematics. This operation can therefore be considered a generalization of Steiner's theorem [34]. For any mechanical system producing Eq. (2) the second-order dynamics EoMs are readily retrieved:

$$\mathbf{M}(\mathbf{q}(t))\ddot{\mathbf{q}}(t) + \mathbf{C}(\mathbf{q}(t), \dot{\mathbf{q}}(t))\dot{\mathbf{q}}(t) = \boldsymbol{\tau}(t), \tag{4}$$

where $\boldsymbol{\tau}(t)$ is the vector of the generalized forces, $\mathbf{M}(\mathbf{q}(t))$ is known from Eq. (3) while $\mathbf{C}(\mathbf{q}(t), \dot{\mathbf{q}}(t)) \in \mathbb{R}^{n\times n}$ is the Coriolis and centrifugal forces matrix. Interested readers can find a comprehensive treatment of this approach in [31] and in [34]. Although the described Lagrangian formulation is very elegant, it leads to computational problems when used for systems with many DoFs, such as the one presented in this work. In particular, the matrices $\mathbf{M}(\mathbf{q}(t))$ and $\mathbf{C}(\mathbf{q}(t), \dot{\mathbf{q}}(t))$ obtained with the Lagrangian method have a large number of highly complex entries. The complexity is due to the highly non-linear and strongly coupled nature of the system. This leads to products of a very large number of trigonometric functions in almost every term of the mentioned matrices. This hinders both the computational efficiency and the precision of the evaluation of these quantities. Moreover, when the number of DoF becomes sufficiently high ($n \geq 10$), the complexity can make it impractical to perform the symbolic derivation, manipulation, and conversion of these matrix functions into a practical and usable form (such as a function handle). Nonetheless, retrieval of the dynamics is imperative for the implementation of a model-based controller. The solution proposed in this paper is to use an implementation of the Newton–Euler (NE) algorithm efficient enough to be run online.

2.4. Algorithm

To obtain computational efficiency, the screw theory-based approaches presented in [34] and [30] are adapted to the dual-arm free-flying space manipulator examined in this work. The adopted procedure is composed of inner and outer parts. The inner solves the inverse dynamics with a NE-based approach for a given set of $\mathbf{q}(t)$, $\dot{\mathbf{q}}(t)$ and $\ddot{\mathbf{q}}(t)$. The outer part calls the inverse dynamics and sets the value of the mentioned quantities to compute the mass matrix $\mathbf{M}(\mathbf{q}(t))$ and the Coriolis vector $\mathbf{c}(\mathbf{q}(t), \dot{\mathbf{q}}(t)) \in \mathbb{R}^n$. This last is defined as: $\mathbf{c}(\mathbf{q}(t), \dot{\mathbf{q}}(t)) = \mathbf{C}(\mathbf{q}(t), \dot{\mathbf{q}}(t))\dot{\mathbf{q}}(t)$.

Consequently, the inverse dynamics algorithm is run several times with:

- $\mathbf{q}(t) = \mathbf{q}_{\text{meas}}(t)$, $\dot{\mathbf{q}}(t) = \dot{\mathbf{q}}_{\text{meas}}(t)$ and $\ddot{\mathbf{q}}(t) = \mathbf{0}_{n\times 1}$ to compute the Coriolis vector.
- $\mathbf{q}(t) = \mathbf{q}_{\text{meas}}(t)$, $\dot{\mathbf{q}}(t) = \mathbf{0}_{n\times 1}$ and $\ddot{q}_i(t) = 1$ for $i = 1, \ldots, n$ to iteratively compute the columns of the mass matrix.

The subscript 'meas' indicates the output of the physical model. In the case of a hardware-in-the-loop simulation, these quantities are replaced by the actual sensor measurements. Additionally, in both cases, the wrench applied at the end effector is set to zero. In Algorithms 1 and 2 a simplified pseudo-code is reported to show the strategy without the excessive complexity introduced by a formal notation. Notice that the measurement of the acceleration, which is normally quite noisy, is not needed in this algorithm. In fact, the vector $\ddot{\mathbf{q}}(t)$ has either all components equal to zero or only one unitary (the remaining are still zeros).

Algorithm 1 OUTER LOOP- Set quantities

```
 1: Given $\mathbf{q}_{\text{meas}}(t)$ and $\dot{\mathbf{q}}_{\text{meas}}(t)$
 2: procedure SetValues($\mathbf{q}_{\text{meas}}(t)$, $\dot{\mathbf{q}}_{\text{meas}}(t)$)
 3:     if $\mathbf{M}(\mathbf{q}(t))$ then
 4:         $\mathbf{q}(t) \leftarrow \mathbf{q}_{\text{meas}}(t)$,
 5:         $\dot{\mathbf{q}}(t) \leftarrow \mathbf{0}_{n\times 1}$
 6:         $\ddot{\mathbf{q}}(t) \leftarrow \mathbf{0}_{n\times 1}$
 7:         for $i = 1, \ldots, n$ do
 8:             $\ddot{q}_i(t) \leftarrow 1$
 9:             $M(:, i) \leftarrow InvDyn(\mathbf{q}(t), \dot{\mathbf{q}}(t), \ddot{\mathbf{q}}(t))$        ▷ $i$-th column
10:         return $\mathbf{M}(\mathbf{q}(t))$
11:         end for
12:     else if $\mathbf{c}(\mathbf{q}(t), \dot{\mathbf{q}}(t))$ then
13:         $\mathbf{q}(t) \leftarrow \mathbf{q}_{\text{meas}}(t)$,
14:         $\dot{\mathbf{q}}(t) \leftarrow \dot{\mathbf{q}}_{\text{meas}}(t)$
15:         $\ddot{\mathbf{q}}(t) \leftarrow \mathbf{0}_{n\times 1}$
16:         $\mathbf{c}(\mathbf{q}(t), \dot{\mathbf{q}}(t)) \leftarrow InvDyn(\mathbf{q}(t), \dot{\mathbf{q}}(t), \ddot{\mathbf{q}}(t))$
17:         return $\mathbf{c}(\mathbf{q}(t), \dot{\mathbf{q}}(t))$
18:     end if
19: end procedure
```

Algorithm 2 INNER LOOP - Newton Euler Inverse Dynamics

```
 1: Given $\mathbf{q}(t)$, $\dot{\mathbf{q}}(t)$ and $\ddot{\mathbf{q}}(t)$
 2: procedure ForwardIterations($\mathbf{q}(t)$, $\dot{\mathbf{q}}(t)$, $\ddot{\mathbf{q}}(t)$)
 3:     for $i = 1, \ldots, n$ do
 4:         Compute ${}^{\{i\}}\mathbf{T}_{\{i-1\}}(t)$        ▷ Pose link $i$ wrt $i-1$
 5:         Compute $\boldsymbol{\xi}_i(t)$        ▷ Twist link $i$
 6:         Compute $\dot{\boldsymbol{\xi}}_i(t)$        ▷ Twist Rate link $i$
 7:     end for
 8: end procedure
 9: return all ${}^{\{i\}}\mathbf{T}_{\{i-1\}}(t)$, $\boldsymbol{\xi}_i(t)$, $\dot{\boldsymbol{\xi}}_i(t)$
10: procedure BackwardIterations(${}^{\{i\}}\mathbf{T}_{\{i-1\}}(t)$, $\boldsymbol{\xi}_i(t)$, $\dot{\boldsymbol{\xi}}_i(t)$)
11:     for $i = 1, \ldots, n$ do
12:         Compute $\boldsymbol{\mathcal{F}}_i(t)$        ▷ Wrench link $i$
13:         Compute $\tau_i(t)$        ▷ Force/Torque Joint $i$
14:     end for
15: end procedure
16: return $\boldsymbol{\tau}(t)$
```

It is important to notice that the computation of $\mathbf{M}(\mathbf{q}(t))$ is independent of $\mathbf{c}(\mathbf{q}(t), \dot{\mathbf{q}}(t))$ since they are the two separate case of 1. To prove this consider Eq. (4). When the mass matrix is being computed with the underlined procedure, the joint rates must be set to zero. Thus, the NE Inverse Dynamics is called with $\dot{\mathbf{q}}(t) = \mathbf{0}_{n\times 1}$. This leads to the Coriolis term becoming null:

$$\mathbf{c}(\mathbf{q}(t), \dot{\mathbf{q}}(t)) = \mathbf{C}(\mathbf{q}(t), \dot{\mathbf{q}}(t))\dot{\mathbf{q}}(t) = \mathbf{0}_{n\times 1}.$$

Consequently, the left-hand-side of Eq. (4) reduces to the inertial term only:

$$\mathbf{M}(\mathbf{q}(t))\ddot{\mathbf{q}}(t) = \boldsymbol{\tau}(t). \tag{5}$$

Similarly, when the Coriolis vector is derived the joint accelerations vector that is nullified. Thus the dynamics equation simplifies to:

$$\mathbf{c}(\mathbf{q}(t), \dot{\mathbf{q}}(t)) = \boldsymbol{\tau}(t). \tag{6}$$

Furthermore, even the iterations to obtain the columns of the mass matrix are all independent from each other.

Recall again that for a given measured configuration $\mathbf{q}_{\text{meas}}(t)$, the iterations to compute $\mathbf{M}(\mathbf{q}(\text{t}))$ require to call the inverse dynamics (Algorithm 2) setting $\mathbf{q}(t) = \mathbf{q}_{\text{meas}}(t)$ and $\ddot{\mathbf{q}}(t) = \mathbf{0}_{n\times 1}$ except for one single unitary component $\ddot{q}_i(t) = 1$. Substituting into Eq. (5):

$$\boldsymbol{\tau}(t) = \mathbf{M}(\mathbf{q}_{\text{meas}}(t)) \ \left[0, \dots, 1, 0, \dots, 0\right]^\top = \mathbf{M}(:, i),$$

where the dependence from $\mathbf{q}_{\text{meas}}(t)$ is omitted for clarity. Thus, at each iteration, the generalized torque output of the Inverse Dynamics (Algorithm 2) is actually the ith column of the mass matrix. An almost identical reasoning is applied to compute $\mathbf{c}(\mathbf{q}(t), \dot{\mathbf{q}}(t))$. In conclusion, all these operations can be run in parallel, thus producing a very efficient procedure. In particular, the Inverse Dynamics can be called in parallel $n+1$ times. The first n are due to the mass matrix, while the last is for the Coriolis vector. The final step is obtaining the Coriolis matrix from vector $\mathbf{c}(\mathbf{q}(t), \dot{\mathbf{q}}(t))$ utilizing the right pseudoinverse of the joint rates vector:

$$\mathbf{C}(\mathbf{q}(t), \dot{\mathbf{q}}(t)) = \mathbf{c}(\mathbf{q}(t), \dot{\mathbf{q}}(t))\dot{\mathbf{q}}^\dagger(t). \tag{7}$$

Once these matrices are available, it is possible to build the SDC matrices $\mathbf{A}(\mathbf{x}(t))$ and $\mathbf{B}(\mathbf{x}(t))$ needed for the synthesis of the SDRE, as described in Sections 2.5 and 3, respectively.

2.5. SDC parameterization

At this point, there is still a missing link between the dynamics and control. Namely, the dynamics equation must be expressed in a suitable form for the SDRE. This is done by constructing a state-space-like formulation known as SDC parameterization. With this aim, Eq. (4) is first solved for the second derivative of the joint position:

$$\ddot{\mathbf{q}}(t) = -\mathbf{M}^{-1}(\mathbf{q}(t))\mathbf{C}(\mathbf{q}(t), \dot{\mathbf{q}}(t))\dot{\mathbf{q}}(t) + \mathbf{M}^{-1}(\mathbf{q}(t))\boldsymbol{\tau}(t). \tag{8}$$

By now defining the state vector $\mathbf{x}(t) := \left[\mathbf{q}^\top(t), \dot{\mathbf{q}}^\top(t)\right]^\top \in \mathbb{R}^{2n}$, the previous can be written in a quasi-state-space form:

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} \mathbf{0}_{n\times n} & \mathbf{I}_{n\times n} \\ \mathbf{0}_{n\times n} & -\mathbf{M}^{-1}(\mathbf{q}(t))\mathbf{C}(\mathbf{q}(t), \dot{\mathbf{q}}(t)) \end{bmatrix} \mathbf{x}(t) + \begin{bmatrix} \mathbf{0}_{n\times n} \\ -\mathbf{M}^{-1}(\mathbf{q}(t)) \end{bmatrix} \boldsymbol{\tau}(t). \tag{9}$$

Recalling now the definition of a time-invariant non-linear control-affine system:

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t)) + \mathbf{g}(\mathbf{x}(t), \mathbf{u}(t)), \tag{10}$$

where state vector is denoted by $\mathbf{x}(t) \in \mathbb{R}^{2n}$ and the input one by $\mathbf{u}(t) \in \mathbb{R}^n$. The equilibrium point of the system is zero $\mathbf{f}(\mathbf{0}) = \mathbf{0}$. $\mathbf{f}(\mathbf{x}(t)) : \mathbb{R}^{2n} \to \mathbb{R}^{2n}$ and $\mathbf{g}(\mathbf{x}(t), \mathbf{u}(t)) : \mathbb{R}^{2n} \times \mathbb{R}^n \to \mathbb{R}^{2n}$ are vector-valued smooth piecewise-continuous functions satisfying the local Lipschitz condition. The SDC parameterization of system (10) is:

$$\begin{aligned} \mathbf{f}(\mathbf{x}(t)) &= \mathbf{A}(\mathbf{x}(t))\mathbf{x}(t), \\ \mathbf{g}(\mathbf{x}(t), \mathbf{u}(t)) &= \mathbf{B}(\mathbf{x}(t))\mathbf{u}(t), \end{aligned} \tag{11}$$

where $\mathbf{A}(\mathbf{x}(t)) : \mathbb{R}^{2n} \to \mathbb{R}^{2n\times 2n}$ and $\mathbf{B}(\mathbf{x}(t)) : \mathbb{R}^{2n} \to \mathbb{R}^{2n\times n}$.

By inspecting Eqs. (9) and (11) it is possible to recognize:

$$\begin{aligned} \mathbf{A}(\mathbf{x}(t)) &= \begin{bmatrix} \mathbf{0}_{n\times n} & \mathbf{I}_{n\times n} \\ \mathbf{0}_{n\times n} & -\mathbf{M}^{-1}(\mathbf{q}(t))\mathbf{C}(\mathbf{q}(t), \dot{\mathbf{q}}(t)) \end{bmatrix}, \\ \mathbf{B}(\mathbf{x}(t)) &= \begin{bmatrix} \mathbf{0}_{n\times n} \\ \mathbf{M}^{-1}(\mathbf{q}(t)) \end{bmatrix}. \end{aligned} \tag{12}$$

The parameters $\mathbf{A}(\mathbf{x}(t))$ and $\mathbf{B}(\mathbf{x}(t))$ are recomputed at each time step of the simulation by numerically updating the matrices $\mathbf{M}(\mathbf{q}(t))$ and $\mathbf{C}(\mathbf{q}(t), \dot{\mathbf{q}}(t))$ with the NE-based algorithm described in Section 2.4. The SDC is then used to compute the gain $\mathbf{X}(\mathbf{x}(t))$ as described in the next section.

3. Controller design

Once the SDC parameterization of the dynamics (Eq. (4)) is available, it is possible to formulate the control problem. The objective of the SDRE control design is to minimize the cost function

$$\begin{aligned} J = &\frac{1}{2}\mathbf{x}^\top(T_\text{f})\mathbf{F}\ \mathbf{x}(T_\text{f}) + \\ &\frac{1}{2}\int_0^{T_\text{f}} \{\mathbf{x}^\top(t)\mathbf{Q}(\mathbf{x}(t))\mathbf{x}(t) + \mathbf{u}^\top(t)\mathbf{R}(\mathbf{x}(t))\mathbf{u}(t)\}\, \mathrm{d}t, \end{aligned} \tag{13}$$

to obtain a trade-off between the error of the system and the input effort of the control law. In the cost function integral (13), $\mathbf{Q}(\mathbf{x}(t)) : \mathbb{R}^{2n} \to \mathbb{R}^{2n\times 2n}$ represents the weighting matrix of states and $\mathbf{R}(\mathbf{x}(t)) : \mathbb{R}^{2n} \to \mathbb{R}^{n\times n}$ denotes the weighting matrix for the inputs, symmetric positive semi-definite and definite, respectively. $\mathbf{F} \in \mathbb{R}^{2n\times 2n}$ sets the penalty of states at final time $T_\text{f}\,[\text{s}]$. Two conditions must be met to find the solution to the SDRE: controllability and observability conditions.

Condition 1. The SDC set of $\{\mathbf{A}(\mathbf{x}(t)), \mathbf{B}(\mathbf{x}(t))\}$ is a controllable pair of matrices of system (10) for all $\mathbf{x}(t) \in \mathbb{R}^{2n}$ in $t \in \mathbb{R}^+$ [23].

Condition 2. The SDC set of $\{\mathbf{A}(\mathbf{x}(t)), \mathbf{Q}^{1/2}(\mathbf{x}(t))\}$ is an observable pair of matrices of system (10) for all $\mathbf{x}(t) \in \mathbb{R}^{2n}$ in $t \in \mathbb{R}^+$ in which $\mathbf{Q}^{1/2}(\mathbf{x}(t))$ is the Cholesky decomposition of $\mathbf{Q}(\mathbf{x}(t))$ [23].

Constructing the Hamiltonian function and applying the stationary condition on that results in the control law:

$$\mathbf{u}(t) = -\mathbf{X}(\mathbf{x}(t))\mathbf{x}(t), \tag{14}$$

The control gain is $\mathbf{X}(\mathbf{x}(t)) = \mathbf{R}^{-1}(\mathbf{x}(t))\mathbf{B}^\top(\mathbf{x}(t))\mathbf{K}(\mathbf{x}(t))$ in which $\mathbf{K}(\mathbf{x}(t)) : \mathbb{R}^{2n} \to \mathbb{R}^{2n\times 2n}$ is symmetric positive-definite solution to the state-dependent differential Riccati equation (SDDRE):

$$\begin{aligned} &\mathbf{A}^\top(\mathbf{x}(t))\mathbf{K}(\mathbf{x}(t)) + \mathbf{K}(\mathbf{x}(t))\mathbf{A}(\mathbf{x}(t)) \\ &-\mathbf{K}(\mathbf{x}(t))\mathbf{B}(\mathbf{x}(t))\mathbf{R}^{-1}(\mathbf{x}(t))\mathbf{B}^\top(\mathbf{x}(t))\mathbf{K}(\mathbf{x}(t)) + \mathbf{Q}(\mathbf{x}(t)) = -\dot{\mathbf{K}}(\mathbf{x}(t)), \end{aligned} \tag{15}$$

with final boundary condition $\mathbf{K}(\mathbf{x}(T_\text{f})) = \mathbf{F}$. The derivation of the SDRE controller is based on the regulation of the system to zero equilibrium point. However, it is possible to regulate any workspace point by defining the deviation from a constant reference state:

$$\delta\mathbf{x}(t) = \mathbf{x}_{\text{ref}} - \mathbf{x}(t), \tag{16}$$

where $\mathbf{x}_{\text{ref}}$ is the constant reference state. Consequently, it is possible to redefine the control input vector for the regulation of a state different from zero:

$$\mathbf{u}(t) = \mathbf{X}(t)\delta\mathbf{x}(t). \tag{17}$$

Notice that the minus sign originally present in Eq. (14) is already included in the definition of $\delta\mathbf{x}(t)$. A block-diagram representation of the computation of the $\mathbf{u}(t)$ is included in Fig. 5. The solution to (15) in this work is chosen the "Lyapunov-based method" with a corresponding negative definite solution to Riccati equation [39]. By considering $t \to \infty$ [39], the solution to an infinite-horizon version of (15) is:

$$\begin{aligned} &\mathbf{A}^\top(\mathbf{x}(t))\mathbf{K}_{\text{ss}}^-(\mathbf{x}(t)) + \mathbf{K}_{\text{ss}}^-(\mathbf{x}(t))\mathbf{A}(\mathbf{x}(t)) \\ &-\mathbf{K}_{\text{ss}}^-(\mathbf{x}(t))\mathbf{B}(\mathbf{x}(t))\mathbf{R}^{-1}(\mathbf{x}(t))\mathbf{B}^\top(\mathbf{x}(t))\mathbf{K}_{\text{ss}}^-(\mathbf{x}(t) + \mathbf{Q}(\mathbf{x}(t)) = \mathbf{0}, \end{aligned} \tag{18}$$

in which $\mathbf{K}_{\text{ss}}^-(\mathbf{x}(t))$ is the steady-state symmetric negative-definite solution to (18). Subtracting (15) from (18) and using new variable:

$$\kappa^{-1}(\mathbf{x}(t)) = \mathbf{K}(\mathbf{x}(t)) - \mathbf{K}_{\text{ss}}^-(\mathbf{x}(t)),$$

result in state-dependent differential Lyapunov equation:

$$\dot{\kappa}(\mathbf{x}(t)) = \mathbf{A}_{\text{cl}}(\mathbf{x}(t))\kappa(\mathbf{x}(t)) + \kappa(\mathbf{x}(t))\mathbf{A}_{\text{cl}}^\top(\mathbf{x}(t)) - \mathbf{B}(\mathbf{x}(t))\mathbf{R}^{-1}(\mathbf{x}(t))\mathbf{B}^\top(\mathbf{x}(t)), \tag{19}$$

where

$$\mathbf{A}_{\text{cl}}(\mathbf{x}(t)) = \mathbf{A}(\mathbf{x}(t)) - \mathbf{B}(\mathbf{x}(t))\mathbf{R}^{-1}(\mathbf{x}(t))\mathbf{B}^\top(\mathbf{x}(t))\mathbf{K}_{\text{ss}}^-(\mathbf{x}(t)),$$

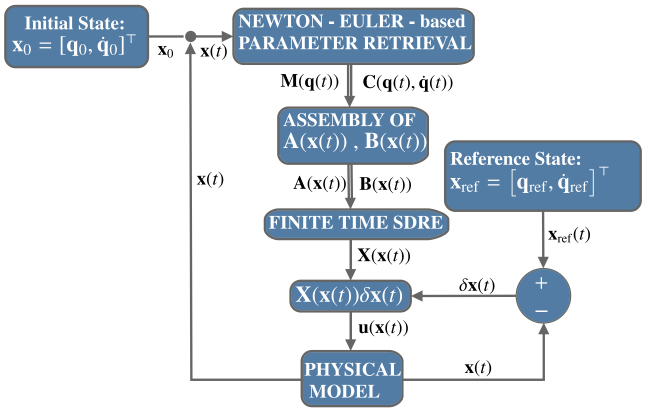

Initial State: $\mathbf{x}_0 = [\mathbf{q}_0, \dot{\mathbf{q}}_0]^\top$
$\mathbf{x}_0$
$\mathbf{x}(t)$
NEWTON - EULER - based PARAMETER RETRIEVAL
$\mathbf{M}(\mathbf{q}(t))$
$\mathbf{C}(\mathbf{q}(t), \dot{\mathbf{q}}(t))$
ASSEMBLY OF $\mathbf{A}(\mathbf{x}(t))$, $\mathbf{B}(\mathbf{x}(t))$
$\mathbf{x}(t)$
$\mathbf{A}(\mathbf{x}(t))$
$\mathbf{B}(\mathbf{x}(t))$
Reference State: $\mathbf{x}_{\text{ref}} = [\mathbf{q}_{\text{ref}}, \dot{\mathbf{q}}_{\text{ref}}]^\top$
FINITE TIME SDRE
$\mathbf{x}_{\text{ref}}(t)$
$\mathbf{X}(\mathbf{x}(t))$
$\mathbf{X}(\mathbf{x}(t))\delta\mathbf{x}(t)$
$\delta\mathbf{x}(t)$
+
−
$\mathbf{u}(\mathbf{x}(t))$
PHYSICAL MODEL
$\mathbf{x}(t)$

Fig. 5. Simplified block diagram of simulation model.

with final boundary condition $\kappa(\mathbf{x}(T_\text{f})) = \{\mathbf{F} - \mathbf{K}_{\text{ss}}^{-}(\mathbf{x}(t))\}^{-1}$.

The closed-form solution to (19) is found by

$$\kappa(\mathbf{x}(t)) = \mathbf{E}(\mathbf{x}(t)) + \exp\{\mathbf{A}_{\text{cl}}(\mathbf{x}(t))(t - T_\text{f})\}[\mathbf{P}(\kappa(T_\text{f})) - \mathbf{E}(\mathbf{x}(t))]\exp\{\mathbf{A}_{\text{cl}}^\top(\mathbf{x}(t))(t - T_\text{f})\},$$

where $\mathbf{E}(\mathbf{x}(t))$ is the solution to algebraic Lyapunov equation:

$$\mathbf{A}_{\text{cl}}(\mathbf{x}(t))\mathbf{E}(\mathbf{x}(t)) + \mathbf{E}(\mathbf{x}(t))\mathbf{A}_{\text{cl}}^\top(\mathbf{x}(t)) = \mathbf{B}(\mathbf{x}(t))\mathbf{R}^{-1}(\mathbf{x}(t))\mathbf{B}^\top(\mathbf{x}(t)).$$

Finally, the solution to (15) is defined by

$$\mathbf{K}(\mathbf{x}(t)) = \mathbf{K}_{\text{ss}}^{-}(\mathbf{x}(t)) + \kappa^{-1}(\mathbf{x}(t)).$$

The application of this controller is described in Section 4.

4. Implementation

In this section, the SDC parameterization of the dynamics is derived and analyzed in terms of observability and controllability. Moreover, the SIMULINK-Simscape implementation used in this work is described using the simplified block diagram scheme represented in Fig. 5.

4.1. Observability and controllability

To guarantee the existence of a solution to the state-dependent differential Riccati equation (Eq. (15)), it is necessary and sufficient to demonstrate that the system is observable and controllable, as stated by Conditions 1 and 2 presented in Section 3. These conditions are satisfied if and only if the Kalman observability and controllability matrices are full rank.

For the sake of completeness, recall that the observability $\mathcal{O}(\mathbf{x}(t)) : \mathbb{R}^{2n} \rightarrow \mathbb{R}^{2n^2 \times 2n}$ and controllability $\mathcal{C}(\mathbf{x}(t)) : \mathbb{R}^{2n} \rightarrow \mathbb{R}^{2n \times 2n^2}$ matrices are defined as:

$$\mathcal{O}(\mathbf{x}(t)) = \begin{bmatrix} \mathbf{Q}^{1/2} \\ \mathbf{Q}^{1/2}\mathbf{A} \\ \mathbf{Q}^{1/2}\mathbf{A}^2 \\ \vdots \\ \mathbf{Q}^{1/2}\mathbf{A}^{n-1} \end{bmatrix}, \qquad (20)$$
$$\mathcal{C}(\mathbf{x}(t)) = \begin{bmatrix} \mathbf{B} & \mathbf{AB} & \mathbf{A}^2\mathbf{B} & \dots & \mathbf{A}^{n-1}\mathbf{B} \end{bmatrix}.$$

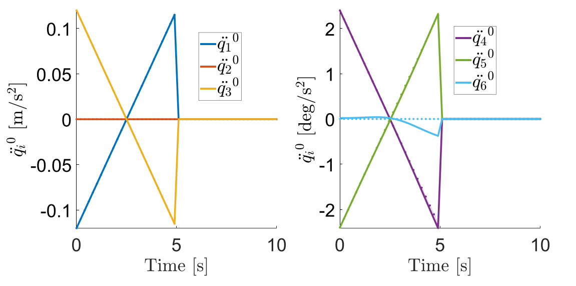
In the observability matrix, $\mathbf{Q} = \mathbf{Q}(\mathbf{x}(t))$ is used in place of the output matrix since a full-state feedback regulation case is considered [23]. If these matrices are always full rank it implies the linear independence of their columns, thus guaranteeing that they will never be singular. These conditions can be checked throughout the real-world operation by making sure that they are not singular at time $t = 0$ and that their ranks never change. Even though these requirements could be computed online during the mission, they would not be of actual help if a singular (unobservable/uncontrollable) configuration is reached while the SMS is operating. This would prevent the computation of the gain matrix $\mathbf{X}(\mathbf{x}(t))$, thus leaving the system uncontrolled and potentially causing a failure of the mission. The mere knowledge of having lost one of these conditions is likely not enough to avoid instability. Consequently, it is of paramount importance to have some guarantees of not being affected by this scenario. In this work, it is decided instead to exploit the SIMULINK-Simscape model for validation through simulation. Indeed, the rank of the matrices $\mathcal{O}(\mathbf{x}(t))$ and $\mathcal{C}(\mathbf{x}(t))$ is checked at each time step of the simulation when the SDDRE is solved. A hint on the reason why observability and controllability should be constant properties of the system can be found in the block-structure of the SDC matrices $\mathbf{A}(\mathbf{x}(t))$ and $\mathbf{B}(\mathbf{x}(t))$, defined in Eq. (12). By recalling that the generalized inertia matrix $\mathbf{M}(\mathbf{q}(t))$ is symmetric and positive definite, thus its diagonal component will always be greater than zero. By looking at the lower block $\mathbf{B}(\mathbf{x}(t))$ can be immediately implied that control authority over the system ($\ddot{\mathbf{q}}(t)$ more specifically) will never be lost. The only remaining issue to be checked is the possibility of $-\mathbf{M}^{-1}(\mathbf{q}(t))\mathbf{C}(\mathbf{q}(t), \dot{\mathbf{q}}(t))$ to be singular. Given the observations made about $\mathbf{M}(\mathbf{q}(t))$, if a problem is to exist it must lie in the matrix $\mathbf{C}(\mathbf{q}(t), \dot{\mathbf{q}}(t))$. During the time frame of the simulation, this is never encountered. Even though the discussion above is limited to controllability, the same observation could be extended to observability as well. In conclusion, even though the intuition here described is not as strong as an ‘if and only if’ condition, it can be considered acceptable when united with successful simulation results.

4.2. SIMULINK implementation

The SMS dynamics and control are implemented and simulated within SIMULINK environment. In particular, the Simscape built-in application is leveraged to model and simulate multi-body and multi-domain physical systems. In this work, Simscape is used to easily assemble and simulate the physical model of the space robot, depicted in Fig. 3. A simplified block diagram describing the overall signal flow as implemented is shown in Fig. 5. The simulation starts at the initial state $\mathbf{x}_0$, which coincides with the Home Configuration of Fig. 3(a). Using the algorithm explained in Section 2.4, the generalized inertia and Coriolis matrices of the SMS are retrieved for the initial conditions and the SDC are assembled as described in Section 2.5. These matrices are then used for the synthesis of the SDRE gain matrix outlined in Section 3. The output of the Finite Time SDRE block is thus the matrix $\mathbf{X}(\mathbf{x})$ described in Eq. (14). More specifically, the MATLAB built-in commands `care` and `lyap` are used to numerically solve the Riccati and Lyapunov equations, respectively. Notice that this is the only computationally inefficient step of the proposed controller, not the online computation of $\mathbf{M}(\mathbf{q}(t))$ and $\mathbf{C}(\mathbf{q}(t), \dot{\mathbf{q}}(t))$. The reason for this bottleneck is that the mentioned functions are not ‘naturally’ implemented in SIMULINK but run on MATLAB engine instead, thus reducing substantially the performances. However, this can be readily fixed by writing the same algorithm in a more efficient language when implemented in real hardware. After the gain matrix $\mathbf{X}(\mathbf{x}(t))$ is available, it is right-multiplied by the current state error $\delta(\mathbf{x}(t))$ and the resulting control input $\mathbf{u}(\mathbf{x}(t))$ is given to the physical model. SIMULINK/Simscape assembles and integrates the equations of motions of the actuated system and thus the new current state $\mathbf{x}(t)$ is extracted and fed back to the NE-based algorithm for parameter retrieval. The cycle is then repeated until the final time instant.

5. Simulation results

In this section, the results of the simulations are reported and analyzed. Firstly the numerical stability of the algorithm presented in Section 2.4 is evaluated by introducing an open loop case as a benchmark. Then, the system is simulated in a representative case while controlled using the finite time SDRE presented in Section 3 to evaluate


(a) (**L**) Base ($j = 0$) P-Joints Accelerations. (**R**) Base ($j = 0$) R-Joints Accelerations.

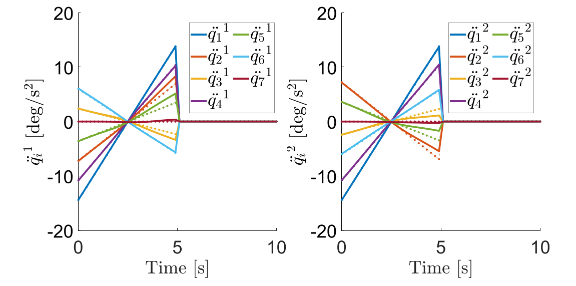
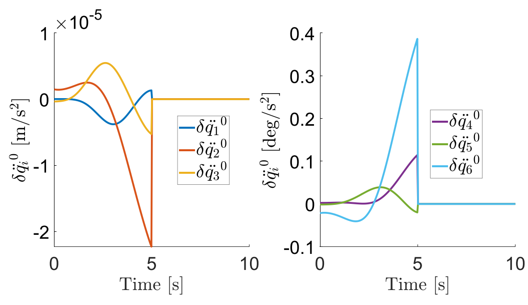

(b) (**L**) ARM - R ($j = 1$) Joints Accelerations. (**R**) ARM - L ($j = 2$) Joints Accelerations.

Fig. 6. Open loop joints accelerations. The solid lines represent the time evolution of $\ddot{q}_i^j$, while the dotted lines stand for the desired accelerations. The reference acceleration is given for the first $5(s)$ of the simulation which is kept running until $10(s)$ to show that no significant errors occur afterwards.


(a) (**L**) Base ($j = 0$) P-Joints Accelerations. (**R**) Base ($j = 0$) R-Joints Accelerations.

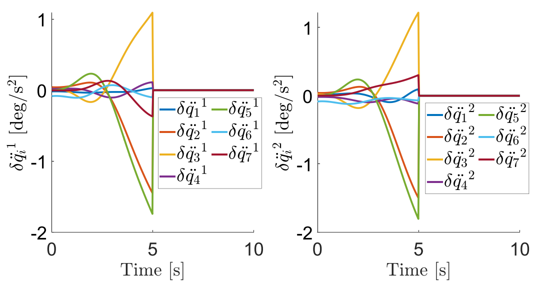

(b) (**L**) ARM - R ($j = 1$) Joints Accelerations. (**R**) ARM - L ($j = 2$) Joints Accelerations.

Fig. 7. Open loop joints accelerations error. The solid lines represent the time evolution of the error on the joint acceleration $\delta\ddot{q}_i^j$. The reference acceleration is given for the first $5(s)$ of the simulation which is kept running until $10(s)$ to show that no unexpected errors occur afterwards.

its performance. Finally, the proposed method is compared against the LQR controller to demonstrate its potentiality and it is tested (without re-tuning) in a scenario with increased dynamic coupling to prove its adaptability.

5.1. Testing the algorithm: Open loop feedback linearization

To evaluate the numerical stability of the algorithm employed to retrieve $\mathbf{M}(\mathbf{q}(t))$ and $\mathbf{C}(\mathbf{q}(t), \dot{\mathbf{q}}(t))$, it is decided to implement an open loop version of the feedback linearization (FL) controller. Consider the following control law:

$$\boldsymbol{\tau}_{\text{FL}}(\ddot{\mathbf{q}}_{\text{des}}(t), t) = \widetilde{\mathbf{M}}(\mathbf{q}(t))\ddot{\mathbf{q}}_{\text{des}}(t) + \widetilde{\mathbf{C}}(\mathbf{q}(t), \dot{\mathbf{q}}(t))\dot{\mathbf{q}}(t), \tag{21}$$

where $\ddot{\mathbf{q}}_{\text{des}}(t)$ collects the desired time histories of the joints' accelerations and the symbol $\widetilde{(\cdot)}$ indicates that the quantity is an estimation of the real one. Substituting (21) in Eq. (4) results in:

$$\begin{aligned}\mathbf{M}(\mathbf{q}(t))\ddot{\mathbf{q}}(t) + \mathbf{C}(\mathbf{q}(t), \dot{\mathbf{q}}(t))\dot{\mathbf{q}}(t) &= \\ \widetilde{\mathbf{M}}(\mathbf{q}(t))\ddot{\mathbf{q}}_{\text{des}}(t) &+ \widetilde{\mathbf{C}}(\mathbf{q}(t), \dot{\mathbf{q}}(t))\dot{\mathbf{q}}(t).\end{aligned} \tag{22}$$

If the dynamics is correctly retrieved by the algorithm a full cancellation of the Coriolis and inertial terms should be achieved, thus obtaining the simple double integrator equation:

$$\ddot{\mathbf{q}}(t) = \ddot{\mathbf{q}}_{\text{des}}(t). \tag{23}$$

This is a simple and reliable performance metric: the system can be simulated applying at each time instant the control input $\boldsymbol{\tau}_{\text{FL}}(\ddot{\mathbf{q}}_{\text{des}}(t), t)$ for a desired acceleration vector and check afterward the extent to which the one obtained from the physical model matches the reference. An implicit assumption of this strategy is assuming the Simscape physical model as the ground truth that the algorithm under analysis aims to reconstruct. This allows us to test if this method can successfully estimate the dynamic properties from a partially known model and the measurements of $\mathbf{q}(t)$ and $\dot{\mathbf{q}}(t)$. In this work, it is decided to construct $\ddot{\mathbf{q}}_{\text{des}}(t)$ such that the corresponding joint rates profiles are parabolic in time, which implies that the joints' accelerations will be linear in t. Thus, the ith desired joint rate of the jth sub-chain takes the following simple polynomial form:

$$\dot{q}^j_{i,\text{ des}}(t) = a_0 + a_1 t + a_2 t^2. \tag{24}$$

Among all the possible desired trajectories, the one beginning and ending at rest is selected.

Thus, by setting null rate at initial ($t = 0$) and final time ($t = T_{\text{f}}$), the following conditions apply:

$$\begin{aligned}\dot{q}^j_{i,\text{ des}}(0) = 0 &\implies a_0 = 0, \\ \dot{q}^j_{i,\text{ des}}(T_{\text{f}}) = 0 &\implies a_2 = -a_1 T_{\text{f}}.\end{aligned} \tag{25}$$

Integrating the Eq. (24) it possible to impose the final amount of displacement of the ith joint of jth sub-chain θ_i^j, thus finding the last free parameter a_1:

$$q^j_{i,\text{ des}}(T_{\text{f}}) = \theta_i^j \implies a_1 = -6\frac{\theta_i^j}{T_{\text{f}}^3}. \tag{26}$$

Consequently, substituting the coefficients in Eq. (24) and taking the time derivative, the (linear) acceleration profile of the ith joint variable is obtained:

$$\ddot{q}^j_{\text{des},i}(t) = -6\frac{\theta_i^j}{T_{\text{f}}^3}(1 - 2t). \tag{27}$$

In Fig. 6 the results of this test are displayed. The reference is tracked quite accurately since the curves almost overlap and only a small deviation is produced. The results are confirmed by analyzing the evolution of the error of the acceleration displayed in Fig. 7. The first two plots show that the base linear and angular errors remain negligible throughout the simulation. The acceleration error experienced by the arms joints is greater but remains low and almost constant until about $3(s)$. Afterwards, some components start growing linearly. Under the normal assumption of never operating in an open loop for several seconds, this outcome ensures that the controller's feedback will prevent the propagation of errors over time.

Table 2
Final pose joint positions. The parameter θ_i^j indicates final amount of displacement of the ith joint of jth sub-chain.

Final pose joint values					
Base		Arm - R		Arm - L	
Joint	Value	Joint	Value	Joint	Value
θ_1^0	1 [m]	θ_1^1	−60 [deg]	θ_1^2	−60 [deg]
θ_2^0	0 [m]	θ_2^1	−25 [deg]	θ_2^2	25 [deg]
θ_3^0	1 [m]	θ_3^1	10 [deg]	θ_3^2	−10 [deg]
θ_4^0	40 [deg]	θ_4^1	−45 [deg]	θ_4^2	−45 [deg]
θ_5^0	0 [deg]	θ_5^1	−15 [deg]	θ_5^2	15 [deg]
θ_6^0	−25 [deg]	θ_6^1	25 [deg]	θ_6^2	−25 [deg]
NA	NA	θ_7^1	0 [deg]	θ_7^2	0 [deg]

Table 3
Weight matrices of the finite time SDRE cost functional.

Weighting	Symbol	Value
State	$\mathbf{Q}$	$\mathrm{diag}\{\mathbf{10}_{6\times1},\ \mathbf{8}_{14\times1},\ \mathbf{1}_{20\times1}\}$
Final State	$\mathbf{F}$	$20\mathbf{Q}$
Control Effort	$\mathbf{R}$	$\mathrm{diag}\{\mathbf{1}_{6\times1}\times10^{-3},\ \mathbf{10}_{14\times1}\}$

5.2. Regulation case

Once the reliability of the algorithm presented in Section 2.4 is proven, the loop is closed to validate the capability of the finite time SDRE described in Section 3 to regulate the SMS pose. The space robot is required to move from the home configuration discussed in Section 2 and depicted in Fig. 3(a) to a final pose whose values are reported in Table 2.

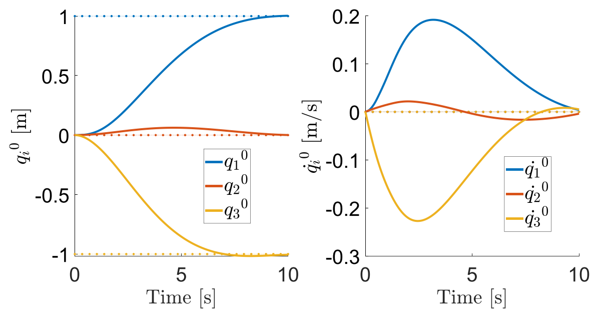
Since a regulation scenario is under analysis, the desired final rates are simply equal to zero:

$$\dot{q}_i^j(T_\mathrm{f}) = \dot{\theta}_i^j = 0\ \forall\ i, j. \tag{28}$$

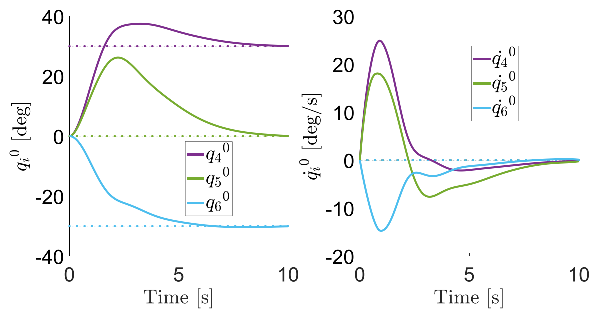
In this paper, the simulation time interval is set to $t = [0, 10](\mathrm{s})$ and is carried out using the weighting matrices (Eq. (13)) reported in Table 3.

The tuning process of the Finite Time SDRE controller involves adjusting these three key coefficient matrices: $\mathbf{Q}$, $\mathbf{F}$, and $\mathbf{R}$. Iterative refinement is often necessary to enhance the controller's performance [40].

Usually, these matrices are selected as diagonal, in order to be able to gain some a priori intuition of their influence on the SDRE. In this case, the $\mathbf{Q}$ matrix is linked to the state and demands an increase in its elements to minimize errors in regulation or tracking. Conversely, reducing the elements of the $\mathbf{R}$ matrix enhances the precision of the controller. The first half of the diagonal elements of $\mathbf{Q}$ pertain to the joints' positions, while the second half relates to their rates. Thus, an analogy can be drawn with the influence of proportional and derivative gains in a PD controller. Finally, the $\mathbf{F}$ matrix, weighting on the final state, is typically chosen to be proportional to $\mathbf{Q}$, with the coefficient expressing the relative importance of $\mathbf{F}$ compared to $\mathbf{Q}$ in shaping the controller's behavior. Careful adjustment of these matrices is crucial for achieving adequate system performance. In this work, the standard approach described before is used to realize the controller, but newer strategies are under development. For instance, in [41] the implementation of (SDRE) control algorithms through the use of artificial neural networks is presented with the aim of reducing the computational cost of this controller. Also in [42] it is proposed a combination of fuzzy logic and neural networks is leveraged to approximate the SDRE control. The evolution of the DoFs belonging to the base ($j = 0$) during regulation is shown in Fig. 8. In the left plots, it is possible to notice that each joint variable approaches and meets the reference at $t = T_\mathrm{f}$. The lack of overshooting and the negligible steady state error indicate that the controller is capable of producing precise and well-damped closed-loop dynamics. In the plots on the right, the trajectories of the joint rates can instead be appreciated. As it is possible to see, they start and end at zero following a bell-shaped trajectory. This shows that the SMS base has reached a stable final pose.

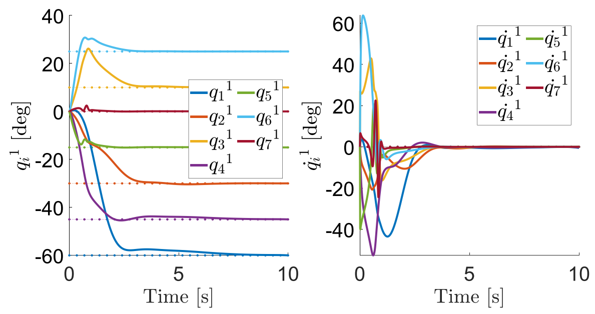


(a) (**L**) Base ($j = 0$) P-Joints Positions. (**R**) Base ($j = 0$) P-Joints Rates.

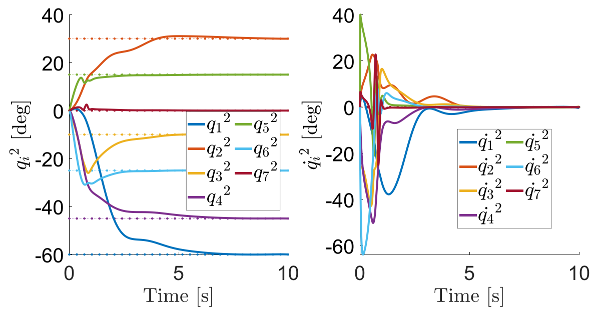


(b) (**L**) Base ($j = 0$) R-Joints Positions. (**R**) Base ($j = 0$) R-Joints Rates.

Fig. 8. Closed loop base joints positions and rates. The solid lines represent the time evolution of q_i^0 and $\dot{q}_i^0$, while the dotted lines stand for the reference values.



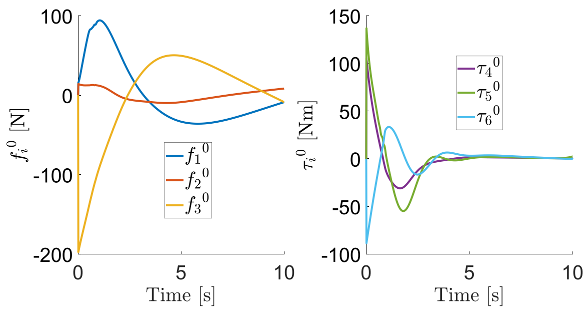
(a) (**L**) ARM - R ($j = 1$) Joints Positions. (**R**) ARM - R ($j = 1$) Joints Rates.



(b) (**L**) ARM - L ($j = 2$) Joints Positions. (**R**) ARM - L ($j = 2$) Joints Rates.

Fig. 9. Closed loop arms joints positions and rates. The solid lines represent the time evolution of q_i^j and $\dot{q}_i^j$, while the dotted lines stand for the reference values.

Similar observations can be made when the arm joints are studied. In Fig. 9 the closed loop evolution in time of the arms' joint angles and


(a) (**L**) Base ($j = 0$) P-Joints Forces. (**R**) Base ($j = 0$) R-Joints Torques.

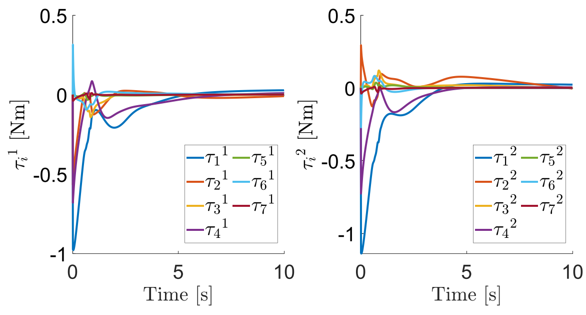

(b) (**L**) ARM - R ($j = 1$) Joints Torques. (**R**) ARM - L ($j = 2$) Joints Torques.

Fig. 10. Closed loop joints forces and torques. All control inputs are computed using the finite time SDRE.

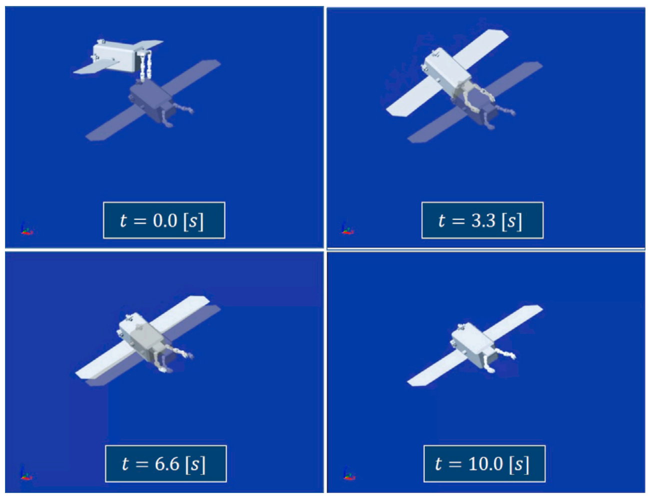
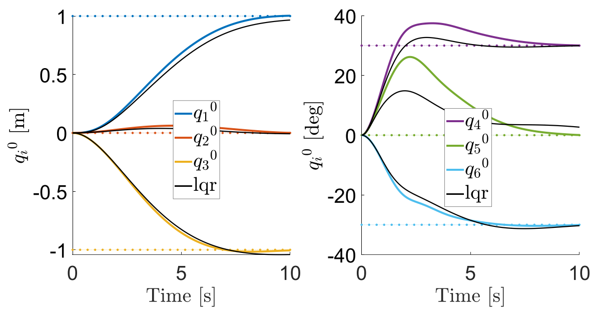


Fig. 11. Snapshots of the simulated operation at four different time instants.

rates are displayed. The reference position is reached very quickly and it is maintained in steady state condition for the rest of the simulation. This is confirmed through an analysis of the arm joint rates, which upon reaching zero (the reference point), remain at rest. Once again, the capabilities of finite-time SDRE in generating precise and well-damped closed-loop dynamics are validated.

Finally, the closed-loop control forces and torques are shown in Fig. 10. Notice that the response generated by the SDRE is quite typical for an optimal controller. Indeed, it tends to generate a relatively large control (and acceleration) at the beginning of the operation (when the error is larger). As long as the actuators do not reach saturation, this should not be perceived as a limitation, but rather as an added reason to realize a realistic simulation before conducting any hardware-in-the-loop testing. In addition, an ideal actuation is assumed, thus neglecting the actuator dynamics. Nevertheless, to maintain generality, the goal is to design a controller that requires reasonable forces and torques. Given the results displayed in Fig. 10, it appears clear that the magnitude of the control action is well within the capability of the actuators available on the market for a medium-sized spacecraft, such as the SMS presented in this work (see Appendix B).


(a) (**L**) Base ($j = 0$) P-Joints Positions. (**R**) Base ($j = 0$) R-Joints Positions.

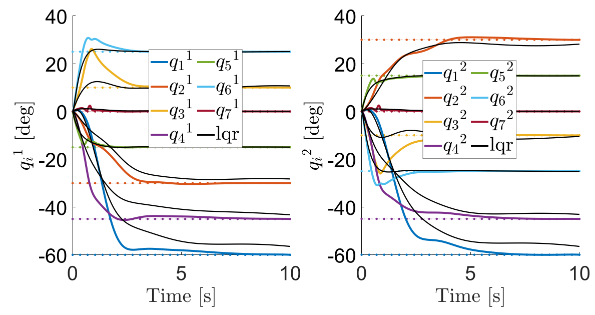
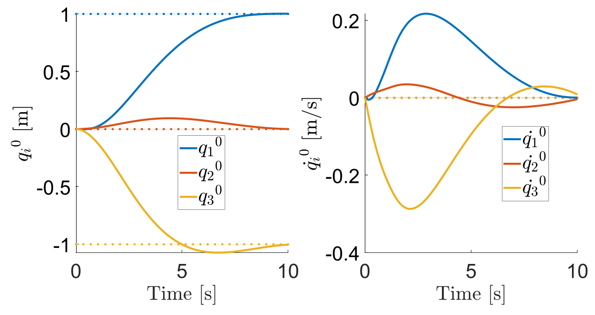

(b) (**L**) ARM - R ($j = 1$) Joints Positions.(**L**) ARM - L ($j = 2$) Joints Positions.

Fig. 12. Comparison between SDRE controller and LQR. The colored solid lines represent the time evolution of q_i^0 when the SDRE is employed and while the black lines display the response generated by the LQR. The dotted lines stand for the reference values. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)


(a) (**L**) Base ($j = 0$) P-Joints Positions. (**R**) Base ($j = 0$) P-Joints Rates.

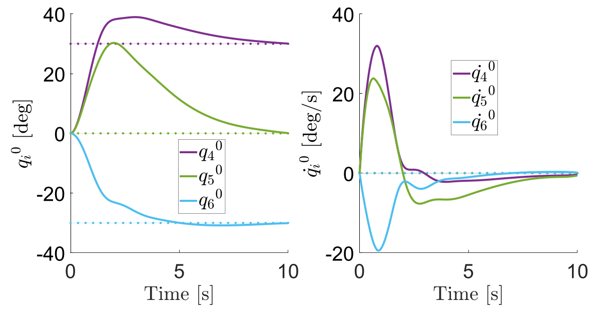
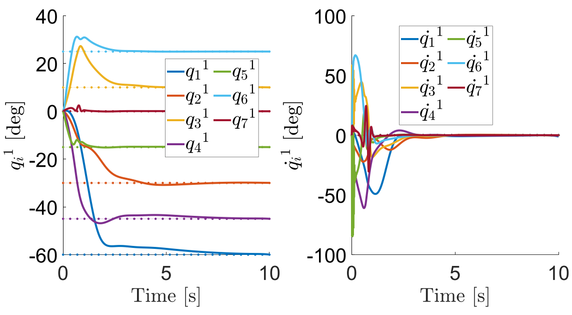
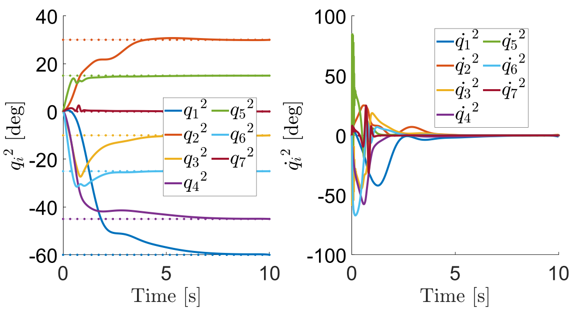

(b) (**L**) Base ($j = 0$) R-Joints Positions. (**R**) Base ($j = 0$) R-Joints Rates.

Fig. 13. Depleted-fuel closed loop base joints positions and rates. The solid lines represent the time evolution of q_i^0 and $\dot{q}_i^0$, while the dotted lines stand for the reference values.



(a) (**L**) ARM - R ($j = 1$) Joints Positions. (**R**) ARM - R ($j = 1$) Joints Rates.



(b) (**L**) ARM - L ($j = 2$) Joints Positions. (**R**) ARM - L ($j = 2$) Joints Rates.

Fig. 14. Depleted-fuel closed loop arms joints positions and rates. The solid lines represent the time evolution of q_i^j and $\dot{q}_i^j$, while the dotted lines stand for the reference values.

The reader can develop a comprehensive understanding of the overall motion, encompassing both the base and arms, by referring to Fig. 11.

5.3. Performance comparison

The State-Dependent Riccati Equation controller is tested against a simpler Linear Quadratic Regulator (LQR) to demonstrate the benefits of using the method proposed in this work.

In order to obtain the LQR, the state-dependent coefficients $\mathbf{A}(\mathbf{x}(t))$ and $\mathbf{B}(\mathbf{x}(t))$ are evaluated at the initial state $t = 0$ to obtain the state and input matrices. The authors acknowledge that this is not the most rigorous approach for the LQR synthesis, since it would require the linearization of the system about the trim point. However, when the system is characterized by many strongly coupled equations of motion, it becomes impractical to manipulate them, as clarified by Fig. 2. Consequently, it is decided to use the evaluation at time $t = 0$ of the SDC parameters as the state and input matrices of the LQR. Moreover, the LQR is synthesized using the same **Q** and **R** matrices reported in Table 3 to produce a fair test. As it is possible to notice in Fig. 12, the SDRE outperforms the LQR since it is able to regulate the desired pose with higher accuracy. The results provided in Tables B.4 and B.6 evidence that the position errors at the end of the simulation are higher for the LQR compared to the SDRE in almost all the variables. The key advantage of the SDRE lies in its adaptive nature, which allows it to dynamically adjust its control parameters based on the system's state. On the contrary, the LQR relies on fixed gains and assumes linearity, which leads to sensibly greater final time errors. In summary, the State-Dependent Riccati Equation controller outperforms the LQR by offering a more flexible approach to control in nonlinear systems, leading to significantly reduced positioning errors and improved overall performance.

5.4. Further analysis

During the mission the fuel reserves available on board the SMS will progressively be depleted. This, in turn, will change the inertial properties of the system. More specifically, the base of the space robot will experience a relevant loss of mass which could easily amount to a third of the total mass of the base. The natural consequence of a less massive base is an increase in the dynamic coupling. In fact, any reaction force/torque exerted by the arms on the base will now have a more pronounced effect on its position and attitude. The main results of simulating the system with a 30% lighter base are reported in Figs. 13 and 14. The performance of the Finite Time SDRE is slightly degraded by the increased dynamic coupling. In particular, it is possible to see that closed loop system exhibits slower dynamics. The results are further confirmed in Table B.5, which recounts the positioning errors of each joint. The results are less precise than those obtained for the nominal case (see Table B.4) but remain acceptable nonetheless. The necessity to move at slower rates is likely not an issue in space applications, where safety constraints impose strict upper-bound speeds to minimize the damage of possible unscheduled impacts [36].

Furthermore, is important to stress that the same weighting matrices reported in Table 3 are used in this simulation as well. This demonstrates a certain degree of robustness of the proposed method even though practical applications would likely opt for additional tuning.

6. Conclusion

This paper presented a numerical method for online computation of the state-dependent coefficient (SDC) matrices in the context of the finite-time state-dependent Riccati equation (SDRE) controller. The conventional offline approach for generating the SDC matrices becomes impractical for high DoF complex systems where deriving the analytical dynamics equations is not a viable option. To overcome this issue, the proposed numerical method updates the SDC matrices at each time step, providing a novel solution compared to the offline approach. The study focused on a fully coupled dynamic model of a dual-arm space robot, which involves two robotic arms mounted on a six-degree-of-freedom free-flying spacecraft. By modeling the dual-arm space robot as an equivalent manipulator model, the generalized inertia and Coriolis matrices are obtained numerically leveraging an NE-based algorithm. This approach allows for online computation of these matrices, despite the high number of degrees of freedom (DoFs), which is crucial for the synthesis of the SDRE controller. To demonstrate the effectiveness of the finite-time SDRE controller augmented with the online numerical derivation of the SDC, various illustrative tasks are analyzed. In particular, the system is tasked to reach a stable configuration within a time interval of $10(s)$. The performances of the finite-time SDRE during this operation are compared to those of a simpler LQR and the SDRE clearly outperforms the latter in terms of better positioning errors. Finally, the SDRE undergoes a robustness test in which it has to regulate a system characterized by a stronger dynamic coupling (due to a 30% lighter base) without re-tuning the weighting matrices. The simulation results validate the efficacy of the proposed method in handling space manipulation tasks. Furthermore, the SDRE controller emerges as a promising solution for real-world applications.

Declaration of competing interest

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

Acknowledgments

The authors would like to thank the reviewers, the associate editor, and the editor-in-chief for their very helpful comments and the time that they spent reading the manuscript and providing such precise feedback. Their comments and suggestions significantly improved the quality of this manuscript.

This work was supported by the project "Smart Robotics for On-Orbit Servicing Applications" (AROSA, CPP-2021-008629), funded by the Spanish Ministerio de Ciencia e Innovacion.

Appendix A. Tables of final positioning errors

The final positioning error for the Finite Time SDRE both in nominal and depleted-fuel scenarios are reported in Tables B.4 and B.5, respectively. Finally, Table B.6 display the same data for the LQR.

Appendix B. Inertial and geometric parameters

SMS parameters are reported in Table B.7.

Table B.4
Finite-Time SDRE positioning errors at final time $t = 10$(s). The parameter δq_i^j indicates the positioning error the ith joint of jth sub-chain at the end of the simulation.

Finite Time SDRE positioning error at final time $t = 10$(s)					
Base		Arm - R		Arm - L	
Joint	Value	Joint	Value	Joint	Value
δq_1^0	−0.003 [m]	δq_1^1	−0.012 [deg]	δq_1^2	−0.011 [deg]
δq_2^0	0.001 [m]	δq_2^1	−0.029 [deg]	δq_2^2	0.024 [deg]
δq_3^0	0.005 [m]	δq_3^1	−0.009 [deg]	δq_3^2	0.011 [deg]
δq_4^0	−0.036 [deg]	δq_4^1	−0.002 [deg]	δq_4^2	0.001 [deg]
δq_5^0	−0.046 [deg]	δq_5^1	−0.001 [deg]	δq_5^2	−0.001 [deg]
δq_6^0	0.093 [deg]	δq_6^1	−0.002 [deg]	δq_6^2	0.003 [deg]
NA	NA	δq_7^1	−0.002 [deg]	δq_7^2	0.001 [deg]

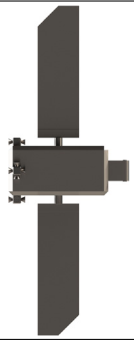
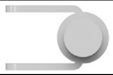
Table B.5
Finite-Time SDRE (Depleted-fuel) positioning errors at final time $t = 10$(s). The parameter δq_i^j indicates the positioning error the ith joint of jth sub-chain at the end of the simulation.

Finite Time SDRE (Depleted Fuel) positioning error at final time $t = 10$(s)					
Base		Arm - R		Arm - L	
Joint	Value	Joint	Value	Joint	Value
δq_1^0	−0.002 [m]	δq_1^1	−0.023 [deg]	δq_1^2	−0.019 [deg]
δq_2^0	−0.004 [m]	δq_2^1	−0.044 [deg]	δq_2^2	0.021 [deg]
δq_3^0	0.004 [m]	δq_3^1	−0.015 [deg]	δq_3^2	0.012 [deg]
δq_4^0	−0.063 [deg]	δq_4^1	−0.006 [deg]	δq_4^2	0.004 [deg]
δq_5^0	−0.085 [deg]	δq_5^1	−0.001 [deg]	δq_5^2	−0.001 [deg]
δq_6^0	0.071 [deg]	δq_6^1	−0.002 [deg]	δq_6^2	0.003 [deg]
NA	NA	δq_7^1	−0.005 [deg]	δq_7^2	0.001 [deg]

Table B.6
LQR positioning errors at final time $t = 10$(s). The parameter δq_i^j indicates the positioning error the ith joint of jth sub-chain at the end of the simulation.

LQR positioning error at final time $t = 10$(s)					
Base		Arm - R		Arm - L	
Joint	Value	Joint	Value	Joint	Value
δq_1^0	0.034 [m]	δq_1^1	−3.510 [deg]	δq_1^2	−3.480 [deg]
δq_2^0	0.007 [m]	δq_2^1	−1.758 [deg]	δq_2^2	1.822 [deg]
δq_3^0	0.0405 [m]	δq_3^1	−0.694 [deg]	δq_3^2	0.456 [deg]
δq_4^0	0.037 [deg]	δq_4^1	−1.765 [deg]	δq_4^2	−1.908 [deg]
δq_5^0	−2.683 [deg]	δq_5^1	−0.083 [deg]	δq_5^2	0.096 [deg]
δq_6^0	0.295 [deg]	δq_6^1	−0.074 [deg]	δq_6^2	0.088 [deg]
NA	NA	δq_7^1	−0.075 [deg]	δq_7^2	−0.075 [deg]

Table B.7
Inertial and geometric parameters of SMS. The characteristic length of each component is found using $L^{*} = \frac{Volme}{Area}$.

Drawing	Link index	Mass [kg]	{PA}-Inertia [Kg m^2]	Characteristic length [m]
	0 – 6	890	$\mathrm{diag}\{[98, 130, 179]\}$	0.12
	1^1 and 1^2	1.04	$10^{-3} \times \mathrm{diag}\{[0.85, 0.82, 0.92]\}$	0.01
	2^1 and 2^2	1.43	$10^{-3} \times \mathrm{diag}\{[3.60, 4.25, 1.61]\}$	0.01
	3^1 and 3^2	1.03	$10^{-3} \times \mathrm{diag}\{[1.01, 0.95, 0.83]\}$	0.01
	4^1 and 4^2	1.48	$10^{-3} \times \mathrm{diag}\{[5.13, 4.43, 1.72]\}$	0.01
	5^1 and 5^2	0.95	$10^{-3} \times \mathrm{diag}\{[0.77, 0.84, 0.81]\}$	0.01
	6^1 and 6^2	1.32	$10^{-3} \times \mathrm{diag}\{[2.16, 2.70, 1.61]\}$	0.01
	7^1 and 7^2	0.71	$10^{-3} \times \mathrm{diag}\{[3.40, 2.20, 2.02]\}$	0.01

References

[1] B.M. Moghaddam, R. Chhabra, On the guidance, navigation and control of in-orbit space robotic missions: A survey and prospective vision, Acta Astronaut. 184 (2021) 70–100.
[2] R.R. Santos, D.A. Rade, I.M. da Fonseca, A machine learning strategy for optimal path planning of space robotic manipulator in on-orbit servicing, Acta Astronaut. 191 (2022) 41–54.
[3] Z. Cheng, X. Hou, X. Zhang, L. Zhou, J. Guo, C. Song, In-orbit assembly mission for the space solar power station, Acta Astronaut. 129 (2016) 299–308.
[4] J.L. Forshaw, G.S. Aglietti, S. Fellowes, T. Salmon, I. Retat, A. Hall, T. Chabot, A. Pisseloup, D. Tye, C. Bernal, et al., The active space debris removal mission RemoveDebris. Part 1: From concept to launch, Acta Astronaut. 168 (2020) 293–309.
[5] B. Ma, Z. Jiang, Y. Liu, Z. Xie, Advances in space robots for on-orbit servicing: A comprehensive review, Adv. Intell. Syst. (2023) 2200397.
[6] K. Yoshida, Engineering test satellite VII flight experiments for space robot dynamics and control: theories on laboratory test beds ten years ago, now in orbit, Int. J. Robot. Res. 22 (5) (2003) 321–335.
[7] E. Papadopoulos, F. Aghili, O. Ma, R. Lampariello, Robotic manipulation and capture in space: A survey, Front. Robotics AI (2021) 228.
[8] G. Hirzinger, K. Landzettel, B. Brunner, M. Fischer, C. Preusche, D. Reintsema, A. Albu-Schäffer, G. Schreiber, B.-M. Steinmetz, Dlr's robotics technologies for on-orbit servicing, Adv. Robot. 18 (2) (2004) 139–174.
[9] D. Reintsema, K. Landzettel, G. Hirzinger, Dlr's advanced telerobotic concepts and experiments for on-orbit servicing, Adv. Telerobotics (2007) 323–345.
[10] Y. Luo, J. Zhang, G. Tang, Survey of orbital dynamics and control of space rendezvous, Chin. J. Aeronaut. 27 (1) (2014) 1–11.
[11] J.R. Wertz, R. Bell, Autonomous rendezvous and docking technologies: status and prospects, Space Syst. Technol. Oper. 5088 (2003) 20–30.
[12] C. Kaiser, F. Sjöberg, J.M. Delcura, B. Eilertsen, SMART-OLEV—An orbital life extension vehicle for servicing commercial spacecrafts in GEO, Acta Astronaut. 63 (1–4) (2008) 400–410.
[13] C.P. Mark, S. Kamath, Review of active space debris removal methods, Space Policy 47 (2019) 194–206.
[14] A. Flores-Abad, O. Ma, K. Pham, S. Ulrich, A review of space robotics technologies for on-orbit servicing, Prog. Aerosp. Sci. 68 (2014) 1–26.
[15] Y. Gao, S. Chien, Review on space robotics: Toward top-level science through space exploration, Science Robotics 2 (7) (2017) eaan5074.
[16] NASA, NASA images, NASA (2015) https://www.nasa.gov/multimedia/imagegallery/index.html.
[17] Y. Umetani, K. Yoshida, et al., Resolved motion rate control of space manipulators with generalized Jacobian matrix, IEEE Trans. Robotics Autom. 5 (3) (1989) 303–314.
[18] Z. Vafa, S. Dubowsky, On the dynamics of manipulators in space using the virtual manipulator approach, in: Proceedings. 1987 IEEE International Conference on Robotics and Automation, Vol. 4, IEEE, 1987, pp. 579–585.
[19] C. Li, Adaptive and robust composite control of coordinated motion of space robot system with prismatic joint, in: Proceedings of the 4th World Congress on Intelligent Control and Automation (Cat. No. 02EX527), Vol. 2, IEEE, 2002, pp. 1255–1259.
[20] Y.-P. Chen, S.-C. Lo, Sliding-mode controller design for spacecraft attitude tracking maneuvers, IEEE Trans. Aerosp. Electron. Syst. 29 (4) (1993) 1328–1333.
[21] N. Nasiri, A. Fakharian, M.B. Menhaj, A novel controller for nonlinear uncertain systems using a combination of SDRE and function approximation technique: Regulation and tracking of flexible-joint manipulators, J. Franklin Inst. B 358 (10) (2021) 5185–5212.
[22] S.R. Nekoo, Model reference adaptive state-dependent Riccati equation control of nonlinear uncertain systems: Regulation and tracking of free-floating space manipulators, Aerosp. Sci. Technol. 84 (2019) 348–360.
[23] S.R. Nekoo, Output-and state-dependent Riccati equation: An output feedback controller design, Aerosp. Sci. Technol. 126 (2022) 107649.
[24] T. Qi, L. Yanfei, W. Zhong, Z. Pengtao, Speed trajectory tracking control for free-floating space robot with uncertain inertial parameters, in: IOP Conference Series: Materials Science and Engineering, Vol. 569, (3) IOP Publishing, 2019, 032028.
[25] T. Çimen, State-dependent Riccati equation (SDRE) control: a survey, IFAC Proc. Vol. 41 (2) (2008) 3761–3775.
[26] J. Yao, S. Rafee Nekoo, M. Xin, Approximate optimal control design for quadrotors: A computationally fast solution, Optim. Control Appl. Methods (2023).
[27] T. Çimen, Systematic and effective design of nonlinear feedback controllers via the state-dependent Riccati equation (SDRE) method, Annu. Rev. Control 34 (1) (2010) 32–51.
[28] T. Cimen, Survey of state-dependent Riccati equation in nonlinear optimal feedback control synthesis, J. Guid. Control Dyn. 35 (4) (2012) 1025–1047.
[29] R. Babazadeh, R. Selmic, Distance-based multiagent formation control with energy constraints using SDRE, IEEE Trans. Aerosp. Electron. Syst. 56 (1) (2019) 41–56.
[30] R. Featherstone, Rigid Body Dynamics Algorithms, Springer, 2014.
[31] R.M. Murray, Z. Li, S.S. Sastry, S.S. Sastry, A Mathematical Introduction to Robotic Manipulation, CRC Press, 1994.
[32] M. Wilde, S. Kwok Choon, A. Grompone, M. Romano, Equations of motion of free-floating spacecraft-manipulator systems: An engineer's tutorial, Front. Robotics AI 5 (2018) 41.
[33] A. Suarez, G. Heredia, A. Ollero, Design of an anthropomorphic, compliant, and lightweight dual arm for aerial manipulation, IEEE Access 6 (2018) 29173–29189.
[34] K.M. Lynch, F.C. Park, Modern Robotics, Cambridge University Press, 2017.
[35] W. Xu, B. Liang, Y. Xu, Survey of modeling, planning, and ground verification of space robotic systems, Acta Astronaut. 68 (11–12) (2011) 1629–1649.
[36] W. Fehse, Automated Rendezvous and Docking of Spacecraft, Vol. 16, Cambridge University Press, 2003.
[37] W. Xu, Y. Liu, Y. Xu, The coordinated motion planning of a dual-arm space robot for target capturing, Robotica 30 (5) (2012) 755–771.
[38] M. Massari, M. Zamaro, Application of SDRE technique to orbital and attitude control of spacecraft formation flying, Acta Astronaut. 94 (1) (2014) 409–420.
[39] M.H. Korayem, S. Nekoo, Finite-time state-dependent Riccati equation for time-varying nonaffine systems: Rigid and flexible joint manipulator control, ISA Trans. 54 (2015) 125–144.
[40] S.R. Nekoo, B. Geranmehr, Nonlinear observer-based optimal control using the state-dependent Riccati equation for a class of non-affine control systems, J. Control Eng. Appl. Inform. 16 (2) (2014) 5–13.
[41] R.F. da Costa, O. Saotome, E. Rafikova, R. Machado, Fast real-time SDRE controllers using neural networks, ISA Trans. 118 (2021) 133–143.
[42] S.-W. Kim, S.-Y. Park, C. Park, Spacecraft attitude control using neuro-fuzzy approximation of the optimal controllers, Adv. Space Res. 57 (1) (2016) 137–152.