Dynamic Modeling of Branched Robots using Modular Composition

Frederico Fernandes Afonso Silva and Bruno Vilhena Adorno

Abstract—When modeling complex robot systems such as branched robots, whose kinematic structures are a tree, current techniques often require modeling the whole structure from scratch, even when partial models for the branches are available. This paper proposes a systematic modular procedure for the dynamic modeling of branched robots comprising several subsystems, each composed of an arbitrary number of rigid bodies, providing the final dynamic model by reusing previous models of each branch. Unlike previous approaches, the proposed strategy is applicable even if some subsystems are regarded as black boxes, requiring only twists and their time derivatives, and wrenches at the connection points between those subsystems. To help in the model composition, we also propose a weighted directed graph representation where the weights encode the propagation of twists and their time derivatives, and wrenches between the subsystems. A simple linear operation on the graph interconnection matrix provides the dynamics of the whole system. Numerical results using a 24-DoF fixed-base branched robot composed of eight subsystems show that the proposed formalism is as accurate as a state-of-the-art library for robotic dynamic modeling. Additional results using a 30-DoF holonomic branched mobile manipulator composed of three subsystems demonstrate the fidelity of our model to a modern robotics simulator and its capability of dealing with black box subsystems. To further illustrate how the derived model to a modern robotics simulator and its capability of dealing results using a 30-DoF holonomic branched mobile manipulator state-of-the-art library for robotic dynamic modeling. Additional requirements for branched robots, Modular dynamic modeling, Newton-Euler formalism, Topological graph, Model-based control.

Index Terms—Branched robots, Modular dynamic modeling, Newton-Euler formalism, Topological graph, Model-based control.

I. INTRODUCTION

In the robotics literature, the Newton-Euler formalism is usually presented at the level of each rigid body in the mechanical structure, notably by analyzing the effects of twists and wrenches at the nth link/joint/CoM. This provides a systematic dynamic modeling strategy applicable to serial manipulators [1], [2] and open kinematic trees [3], [4], [5]. However, approaches based on this formalism typically give a monolithic solution to the system and do not allow model composition. See, for instance, the formulations presented in [6].

F. F. A. Silva and B. V. Adorno are with the Department of Electrical and Electronic Engineering and the Manchester Centre for Robotics and AI, The University of Manchester, Oxford Road, Engineering Building A, Manchester M13 9PL, United Kingdom (emails: frederico.silva@ieee.org; bruno.adorno@manchester.ac.uk).

This work was partly done when the authors were affiliated with the Federal University of Minas Gerais, Belo Horizonte, MG, Brazil, and supported by the Brazilian agency CAPES, the UK Research and Innovation (UKRI) under the UK government’s Horizon Europe funding guarantee [grant number EPY024508/1], and the Royal Academy of Engineering under the Research Chairs and Senior Research Fellowships programme.

NOMENCLATURE

| Symbol | Description |
|--------|-------------|
| s      | Number of subsystems of the branched robot. |
| S      | Set containing all subsystems of the branched robot. |
| S_i    | Set of all subsystems that succeed the ith subsystem of the branched robot. |
| n_i    | Number of elements in the set S_i. |
| j      | The ith subsystem of the branched robot. |
| S_j    | Subsystem j that is connected to and succeeds the ith subsystem. |
| p_i    | Subsystem p_i that is connected to and precedes the ith subsystem. |
| S_p_i  | Number of joints/links of the ith subsystem. |
| n_s    | Number of joints that precede b_{i,j} in subsystem i. |
| n_b    | Number of rigid bodies of the branched robot. |
| Q       | Sets containing all the joint configurations, velocities, and accelerations of subsystems that are not black boxes. |
| Q_T, Q_A | Sets containing all subsystems of the branched robot. |
| X_p_j   | Vector containing the rigid transformation between the connection point a_i in the preceding subsystem p_i and the CoM of each rigid body within the ith subsystem. |
| X_j     | Vector containing the rigid transformation between the connection point b_{i,j} in the succeeding subsystem j in the ith subsystem and the CoM of each joint of subsystem i. |
| W_s     | Function that maps the stacked vector of total twists and wrenches at the CoM of each link in the ith subsystem to the corresponding vector of wrenches at the n_s joints. |
| Ξ_s     | Stacked vector of twists Ξ_{p,i} ∈ T^{n_s}i and twist time derivatives Ξ_{s,i} ∈ T^{n_s}i, generated by the ith subsystem, at the n_s CoMs of subsystem i ∈ S. |
| Ξ_p_j   | Stacked vector of twists Ξ_{p,i} ∈ T^{n_s}i and twist time derivatives Ξ_{p,j} ∈ T^{n_s}j, expressed in each of the n_s CoMs of subsystem i, resulting from the twist at the connection point a_{i,j}. |
| Ξ_j     | Stacked vector of total twists and twist time derivatives at the connection points b_{i,j} of subsystem j ∈ S. |
| Γ_j     | Stacked vector of wrenches at the n_s joints of subsystem i due to the wrench at the connection point b_{i,j}, with j ∈ S. |
| Γ_j     | Stacked vector of wrenches, Γ_j = [Γ_j^T, 0_{n_s,n_s}^T]^T ∈ W^{n_s}, that is used to propagate the wrench at the connection point b_{i,j} to all joints of subsystem i. |
| Γ_j     | Stacked vector of total wrenches at the n_s joints of subsystem i ∈ S. |
| ζ_j     | External wrench at the end-effector indexed by ζ(ℓ) of leaf subsystem ℓ ∈ S. |
| ζ_j     | The vector of external wrenches at the ℓ ∈ S leaf subsystems of the branched robot. |
| ζ_n     | An n-dimensional vector of zeros in the set W. |
Monolithic formulations for the dynamic model of branched robots are not a novelty. Park et al. [7], [8] presented algorithms based on the Lie algebra associated with the Lie group SE(3) for the dynamic modeling of open-kinematic chains, whereas Featherstone [9], [10] proposed divide-and-conquer algorithms based on spatial algebra, which were later extended by Mukherjee and Anderson [11] to cover flexible bodies. More recent works on the field of branched robots have focused on specific dynamic characterizations such as the analysis of a 5-prismatic–spherical–spherical parallel mechanism [12] and the identification of non-redundant inertial parameters of branched robots [13], contact analysis [14], [15], motion planning [16], [17], and robot design [18].

Nonetheless, there are plenty of motivations for seeking a general formalism for the systematic composition of partial models to obtain the final model of the complete robotic system. One could assemble a robot using existing systems whose dynamic models are already known, such as the limbs of a humanoid robot. In a different scenario, a self-reconfiguring modular robot [19], [20], [21] could possess the dynamic information of its modules that might be reused. From a control perspective, one could be interested in applying distributed control strategies to the subsystems comprising a highly complex dynamic structure composed of thousands of subsystems whose centralized control would be computationally unfeasible. The few works in literature focused on modular composition either require the previous construction of subsystem libraries or demand full knowledge of the dynamic elements of the whole system.

A. Related works

The application of linear graph theory in mechanism analysis is not a novelty either, being used to the dynamic modeling of single rigid bodies [22] and multibody systems composed of open [23] and closed kinematic chains [24], [25], [26], [27], [28], [29]. Most approaches lead to different graphs for the rotational and translational variables of the mechanism [22], [30], [23], [26], [28], whereas others focus more on computational [31] and mechanical [32] aspects than on the dynamic modeling of the system. Despite exploring several aspects of graph theory, the aforementioned strategies do not deal with model composition and are, therefore, monolithic.

To overcome the drawbacks of monolithic approaches, Jain [33] proposed a technique of partitioning and aggregating graphs that considers subsystems in the dynamic modeling of multibody systems. Subgraph elements are used to compose the mass matrix and the vector of nonlinear Coriolis and gyroscopic terms of the aggregated system, and then the stacked vector of generalized forces is calculated. Despite the system being composed of individual modules, full knowledge of the masses, inertia tensors, Coriolis accelerations, and gyroscopic terms of the whole system is required.

McPhee et al. [34] presented a strategy that uses individual subsystem models to derive the dynamic model of mechatronic multibody systems. Their approach uses free vectors and rotational matrices, thus decoupling translational and rotational components, which requires separate graphs. Moreover, that formalism relies on a symbolic implementation, and each subsystem must be symbolically derived before the modeling process for the complete system starts.

Moving away from graph representations, Orsino and Hess-Coelho [35] proposed a strategy for the modular modeling of multibody systems, whose constraints are written as invariants. They use the model of each subsystem to find the constraint equations among the subsystems. Then, the constraint equations are used to obtain the system dynamic equations. Orsino [36] extended that formalism by proposing a hierarchical description of lumped-parameter dynamic systems that lead to a recursive modeling methodology. Albeit both formulations [35], [36] do not impose limitations on how the dynamic equations of each subsystem must be obtained, the final result is not given in terms of the generalized forces or wrenches of the mechanism but rather by a system of differential-algebraic equations of the mechanism’s generalized coordinates, constraint equations, and dynamic equations of motion. Thus, those strategies are not readily applicable to robotics problems where one typically needs to find the joint forces/torques as a function of the robot dynamics, which then are used to design suitable control laws. Moreover, both formulations [35], [36] consider that the complete subsystem models are fully available.

More recently, Kumar et al. [37] have proposed a modular solution to the kinematics and dynamics of series-parallel hybrid robots. Their approach consists of a graph representation in which edges correspond to rigid bodies and nodes represent the joints connecting them. The authors use the Lie algebra se(3) to represent twists and wrenches in the proposed modular recursive Newton-Euler algorithm, achieving a computationally efficient solution. However, the inverse dynamics algorithm does not exploit the system’s graph representation with its resultant algebraic operations to find the dynamic model for the whole system. Furthermore, the current formulation cannot handle black-box subsystems since twists and wrenches are explicitly propagated between all subsequent rigid bodies in other subsystems and must be available for all other modules.

Hess-Coelho et al. [38] presented a dynamic modular modeling methodology for parallel mechanisms. They use the hierarchical description proposed by Orsino [36] but follow a different approach to derive the model. Jacobian matrices of the subsystem’s angular and linear velocities are used with the Principle of Virtual Power to obtain the Euler-Lagrange model of the robot, and modularity is achieved by using a library of subsystem models. Nonetheless, the process of obtaining such models is highly dependent on geometric analysis of the system (i.e., inverse kinematics) and, if no library containing the subsystems’ models is available, the robot dynamics is found monolithically. Moreover, due to the free-vector representation, translational and rotational components are decoupled. As M Ä E 1 E l l [ 3 9 ] pointed out, decoupled representations based on free vectors do not form a group. Therefore, using free vectors doubles the number of equations in the problem; also, one needs to explicitly consider lever arms and their equivalent for velocities and accelerations. In contrast, in formulations with unified representations, those couplings are implicitly
algebraically found thanks to the machinery of group theory.

Yang et al. [40] proposed a modular approach for the dynamic modeling of cable-driven serial robots. They calculate the energy for each component and then apply an energy-based method to integrate the components into the complete model of the robotic system. Their strategy avoids reformulating the coefficient matrix as the number of modules increases but is not applicable to branched robots containing black-box subsystems.

In conclusion, the existing model composition strategies oftentimes generate different graph representations for the translational and rotational components, increasing the overall complexity, and either require the previous construction of subsystem libraries or demand full knowledge of the dynamic elements of the whole system. Some of them also require a symbolic derivation, as opposed to recursive formulations, which makes it harder to model reconfigurable systems. This paper presents a systematic methodology that overcomes those drawbacks. Table I presents a summary of the differences between our proposed formalism and the works in the literature for the dynamic modeling of branched robots.

B. Statement of contributions

This paper presents the following contributions to the state-of-the-art:

1) A strategy for dynamic model composition, shown in Section II, that is applicable even if some subsystems are regarded as black boxes, requiring only the twists, twist time derivatives, and wrenches at the connection points between different subsystems. Such information can be obtained either from previous calculations or sensor readings.

2) A unified graph representation of the system, shown in Section II-A, that provides the joint wrenches from the calculation of the graph interconnection matrix, in addition to visually depicting the model composition.

3) The proposed formulation imposes no restrictions regarding the algebra used to represent twists and wrenches, as long as some basic properties are respected. Nonetheless, we present an instantiation for dual quaternion algebra in Section III.

4) A model-based wrench-driven end-effector motion controller, shown in Section IV, allowing to control all end-effectors of the branched robot simultaneously.

We perform numerical evaluations of a fixed-base 24-DOF branched robot and a 30-DoF holonomic branched mobile manipulator. We then compare our results with the ones provided by a realistic simulator and Featherstone’s state-of-the-art library for robotic dynamic modeling [3]. To further illustrate how the derived dynamic model can be used in closed-loop control, we also present a simple formulation of a model-based wrench control for branched robots.

This paper is organized as follows: Section II presents the proposed dynamic model composition framework; Section III demonstrates an instantiation of the strategy to dual quaternion algebra; Section IV presents the model-based wrench-driven end-effector motion controller; Section V shows a numerical evaluation of the proposed methodology, and a comparison with a state-of-the-art simulator and state-of-the-art library; Section VI provides the final remarks and points to further research directions; finally, Appendix A briefly reviews the dual quaternion algebra and Appendix B presents the kinematic and dynamic parameters of the robots used in the simulations.

II. ABSTRACT MODEL COMPOSITION

Consider a branched robot composed of $s$ subsystems shown in Fig. 1. Given the interconnection points between pair of subsystems and the twists, twist time derivatives, and wrenches applied at those points, our goal is to obtain a set of equations that describe the whole-body dynamics as a function of the dynamics of each subsystem $i \in \{1, \ldots, s\} \triangleq S$.

Given the subsystem $i \in S$ that immediately precedes and is connected to all $j \in \{j_1, \ldots, j_m\} \triangleq S_i \subset S$, we assume that the first link of subsequent subsystems $j \in S_i$...

1Subsystems can be defined according to what is convenient for each problem. For instance, they could be a mobile base, a manipulator, an off-the-shelf module, or even another branched robot.
can be connected to any link of the preceding subsystem \(i\). For example, in Fig. 1, the first links of subsystem 3 and 7 are connected to the first link of subsystem 1, whereas the first links of subsystems 2 and 5 are connected to the second link of subsystem 1.

Being the complete system an open kinematic tree, each subsystem can only be preceded by one subsystem but can be succeeded by several kinematic chains. Therefore, twists and twist time derivatives generated by the subsystem \(i\) will be propagated to each \(j \in S_i\) and all other subsequent subsystems. On the other hand, the combined wrenches from all \(j \in S_i\) will affect \(i\) and all its preceding subsystems.

Because the forward propagation of twists and twist time derivatives and the backward propagation of wrenches also happen within each subsystem, the idea is similar to the classic Newton-Euler algorithm [41]. Consider that each subsystem \(i \in S\) is composed of \(n_i\) joints/links, is preceded by a subsystem \(p_i \in S\), and is immediately succeeded by all subsystems \(i,j \in S\). Also, consider a function

\[
\mathbf{W}_i : T^{2n_i} \rightarrow W^{n_i}
\]

that maps the stacked vector \(\mathbf{\Xi}_i = \left[ \mathbf{\Xi}_{i,T}^T \mathbf{\Xi}_{i}^T \right]^T \in T^{2n_i}\) of total twists \(\mathbf{\Xi}_{i} \in T^{n_i}\) and twist time derivatives \(\mathbf{\Xi}_{i,T} \in T^{n_i}\) at the center of mass (CoM) of each link in the \(i\)th subsystem to the corresponding vector of total wrenches \(\mathbf{\Gamma}_i \in W^{n_i}\) at the \(n_i\) joints of the \(i\)th subsystem. The wrenches at the joints of each subsystem \(i \in S\) originate from three sources: the twists and their time derivatives at the CoMs of each link in the \(i\)th subsystem; the twist and its time derivative at the connection point \(p_i\) with the preceding subsystem \(p_i\); and the wrenches at the connection points \(b_{i,j}\) with each \(j \in S_i\).

\[
\mathbf{\Gamma}_i = \mathbf{W}_i(\mathbf{\Xi}_i) + \sum_{j \in S_i, j \neq i} \mathbf{\Gamma}_{j,i} \quad \text{with} \quad \mathbf{\Xi}_i = \mathbf{\Xi}_{p_i,i} + \mathbf{\Xi}_{j,i},
\]

where \(\mathbf{\Xi}_{j,i} = \left[ \mathbf{\Xi}_{j,i}^T \mathbf{\Xi}_{j,i}^T \right]^T \in T^{2n_i}\) is the stacked vector of twists and twist time derivatives at the \(n_i\) CoMs of subsystem \(i \in S\); the elements of \(\mathbf{\Xi}_{p,i} \triangleq \left[ \mathbf{\Xi}_{p,i}^T \mathbf{\Xi}_{p,i}^T \right]^T \in T^{2n_i}\) are the twists and twist time derivatives at the connection point \(p_i\).

\(3\)Joints after the connection point are not directly affected by wrenches from subsequent subsystems. Nonetheless, because those wrenches will affect the first \(\eta\) joints of the \(i\)th subsystem, the remaining \(n_i - \eta\) links will be indirectly affected because the twists (and their derivatives) at the \(\eta\) joints arising from this interaction will be propagated to the remaining \(n_i - \eta\) links.
is analogous to \( \Xi \) indicate the wrenches generated at the joints of the subsystem \( p_i \) representing links, and blue crossed circles represent CoMs. The preceding subsystem \( p_i \) represents the root subsystem of the branched robot. Hatched regions indicate the areas influenced by the wrenches at the connection points.

**Figure 2:** Wrenches generated at the joints of the \( i \)th subsystem (pink region). For each subsystem, the large gray circles represent joints, solid black lines represent links, and blue crossed circles represent CoMs. The preceding subsystem \( p_i \) is given in blue, the subsequent subsystems \( j \in S_i \) are colored in green, and red circles on the red dashed lines, numbered from 1 to 3, indicate the connection points. Dotted arrows represent the forward propagation of twists, whereas solid arrows represent the backward propagation of wrenches.

Figure 2: Wrenches generated at the joints of the \( i \)th subsystem (pink region).

For each subsystem, the large gray circles represent joints, solid black lines represent links, and blue crossed circles represent CoMs. The preceding subsystem \( p_i \) represents the root subsystem of the branched robot. Hatched regions indicate the areas influenced by the wrenches at the connection points.

**Example 1.** For the sake of simplicity, let us consider only subsystems 1 and 2 in Fig. 1 and disregard the remaining subsystems. In that case, the vectors \( \Gamma_1 \in W^{n_1} \) and \( \Gamma_2 \in W^{n_2} \) of wrenches at the joints of subsystems 1 and 2, respectively, with \( n_1 = n_2 = 3 \), are given by

\[
\begin{align*}
\Gamma_1 &= W_1 (\Xi_1) + \Gamma_{2,1}, \\
\Gamma_2 &= W_2 (\Xi_2),
\end{align*}
\]

where \( \Xi_1 = \Xi_{1,1} \) and \( \Xi_2 = \Xi_{1,2} + \Xi_{2,2} \).

Since subsystem 1 is the first in the chain, the wrenches at its joints are generated by the twists and their time derivatives at the CoMs of its own links, and by the reaction wrenches generated by subsystem 2. On the other hand, for subsystem 2, which is the last in the chain if we disregard the other subsystems, the wrenches at the joints are generated by the twists and their time derivatives at the CoMs of its own links, in addition to the the twist and twist derivative that subsystem 1 contributes through the connection point.

**A. Graph representation**

Each subsystem in a branched robot may be represented as a vertex in a graph, in which directed, weighted edges represent the propagation of wrenches, twists, and twist time derivatives. The advantage of such representation is that in addition to visually depicting the model composition, it provides the joint wrenches from the calculation of the graph wrench interconnection matrix. For instance, the weighted graph in Fig. 3 represents the system of Example 1 where dashed edges correspond to the propagation of twists and their time derivatives, and solid edges represent the propagation of wrenches.

The wrench interconnection matrix of the graph presented in Fig. 3 is given by

\[
\mathbf{A} = [A_{ij}] \triangleq \begin{bmatrix}
W_1 (\Xi_1) & \Gamma_{2,1} \\
\mathbf{0}_3 & W_2 (\Xi_2)
\end{bmatrix} \in W^{6 \times 2},
\]

where \( \Xi_1 = \Xi_{1,1} \) and \( \Xi_2 = \Xi_{1,2} + \Xi_{2,2} \). The vectors \( A_{ij} \in W^{n_i} \) of the partitioned matrix \( \mathbf{A} \) indicate the wrench propagation from vertex \( j \) to vertex \( i \), represented as a solid edge. Because there is no wrench propagation from subsystem 1 to 2 in Example 1, \( A_{2,1} = \mathbf{0}_3 \in W^{3} \), which is a vector of zeros in the set \( W \). Therefore, since the corresponding “weight” is zero, there is no solid edge from 1 to 2 in the graph on Fig. 3. The wrench interconnection matrix \( \mathbf{A} \) is analogous to the weighted adjacency matrix of the graph, but the element \( A_{ij} \) provides the joint wrenches imposed by one subsystem onto another instead of a real scalar weight.

More generally, the graph representation of the complete system composed of \( s \) kinematic chains is constructed as follows:

1. Create a vertex for each kinematic chain.
2. Add the edges, according to the following rules:
   a) Each vertex \( i \) has a dashed edge self-loop weighted by its own stacked vector of twists and twist time derivatives \( \Xi_{i,i} \in T^{2n_i} \).
   b) Except for the vertex representing the root subsystem of the branched kinematic chain, each vertex \( i \) has an incoming dashed edge from the vertex \( p_i \) representing its predecessor, “weighted” by the stacked vector of twists and twist time derivatives \( \Xi_{p_i,i} \in T^{2n_i} \), and a solid edge self-loop weighted by its own vector of wrenches given by \( W_i (\Xi_{i,i} + \Xi_{p_i,i}) \in W^{n_i} \).
   c) Except for the vertex representing the root subsystem of the branched kinematic chain, each vertex \( i \)

\[\text{The interconnection matrix is always } \mathbf{A} \in W^{n \times n}, \text{ where } n \text{ is the total number of rigid bodies in the system and } s \text{ is the total number of subsystems of the branched robot.}\]
has an outgoing solid edge that goes to the vertex $p_i$ representing its predecessor, “weighted” by the vector of wrenches $\Gamma_{ij,p_i} \in W^{n_{p_i}}$.

**Example 2.** Consider the 24-DoF branched robot shown in Fig. 1 where $n_1 = n_2 = \cdots = n_8 = 3$. Following the procedure described previously, the weighted graph representing this robot is given in Fig. 2. Moreover, the interconnection matrix $A \in W^{24 \times 8}$ is given by

$$A = \begin{bmatrix}
\mathcal{W}_1 & \Gamma_{2,1} & \Gamma_{3,1} & 0 & 0 & 0 & 0 & 0 \\
0 & \mathcal{W}_2 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & \mathcal{W}_3 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & \mathcal{W}_4 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & \mathcal{W}_5 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & \mathcal{W}_6 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & \mathcal{W}_7 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & \mathcal{W}_8
\end{bmatrix},$$

in which

$$\mathcal{W}_i = \begin{cases}
\mathcal{W}_i(\Xi_{i,i}) & \text{if } i = 1, \\
\mathcal{W}_i(\Xi_{p_i,i}) & \text{if } i \in \{2, \ldots, 8\},
\end{cases}$$

$p_2 = p_3 = p_5 = p_7 = 1, p_4 = 2, p_6 = 5$, and $p_8 = 7$, and $0_3 \in W^3$ is a vector of zeros in $W^3$

The proposition below shows how the adjacency matrix is used to derive the model of the complete assembled system.

**Proposition 3.** Let a branched kinematic system be composed of $n$ rigid bodies divided into a set of $s$ coupled subsystems, each one containing $n_1, n_2, \ldots, n_s$ rigid bodies, respectively. Considering the proposed weighted graph representation with its corresponding adjacency matrix $\mathcal{A}$, the vector of joint wrenches $\Gamma_{\text{total}}$ of the complete system is given by

$$\Gamma_{\text{total}} = [\Gamma_1^T \Gamma_2^T \cdots \Gamma_s^T]^T = A \mathcal{I}_{s} \in W^n, \quad (5)$$

where $\mathcal{I}_{s} \in W^{n_{s}}$ is the vector of the total joint wrenches of the $i$th subsystem, $A \in W^{ns \times s}$ is the interconnection matrix, $\mathcal{I}_{s} \in W^{n_{s}}$ is the vector of identity elements (under the multiplication operation) in the set $W$.

**Proof:** Each block element $A_{ij} \in W^{n_j}$ of the matrix $A$ represents the weight of the edge from vertex $j$ to vertex $i$ in the interconnection graph, and, therefore, the propagation of wrenches from subsystem $j$ to $i$. Hence, each row of $A$ contains all the wrenches acting upon the joints of the $i$th subsystem. Therefore, the vector $\Gamma_{i}$ of the total wrenches of the $i$th subsystem is given by $\Gamma_{i} = \sum_{j=1}^{s} A_{ij} = \sum_{j=1}^{s} (A_{ij} \cdot 1)$, where $1 \in W$ is the identity element in $W$ under the multiplication operation such that $A_{ij} \cdot 1 = A_{ij}$. Thus, $\Gamma_{\text{total}} = A \mathcal{I}_{s}$.

Algorithms 1, 2, and 3 summarize the proposed modular dynamic modeling formalism. The inputs for the dynamic modular composition (DMC) algorithm are the sets $Q, \bar{Q}$, and $\bar{Q}$, respectively, comprised of the joint configurations, velocities, and accelerations of subsystems that are not black boxes. Algorithm 2 presents the forward propagation of the stacked vector of twists and twist time derivatives $\Xi \in T^{2n}$ at the $n = \sum_{i=1}^{s} n_i$ CoMs of the robot, in which a breadth-first search (BFS) algorithm is used to traverse the tree. Since the graph representing the topology of the branched robot is constant, we assume it is internally available. Between lines 3 and 11 the twists $\Xi_{p_i,j}$ are calculated due to the subsystem’s own motion. The implementation of FORWARD_RECURSION in line 9 depends on the algebra used to represent wrenches and twists. For instance, we provide a formulation using dual quaternion algebra in Algorithm 3. In line 10 the poses $X_{p_{ij},j}$ associated with each of the subsystem’s CoMs are calculated as a function of the subsystem’s configuration $q_j \in \bar{Q}$.
Algorithm 1 Dynamic modular composition (DMC) algorithm for a branched robot composed of $s$ subsystems.

1: function DMC($\mathbf{Q}$, $\dot{\mathbf{Q}}$, $\ddot{\mathbf{Q}}$)
2: \hspace{1cm} $\left(\mathbf{X}, \mathbf{\Xi}, \mathbf{\Xi}^0\right) \leftarrow$ DMC_FORWARD_RECURSION($\mathbf{Q}$, $\dot{\mathbf{Q}}$, $\ddot{\mathbf{Q}}$)
3: \hspace{1cm} $\Gamma \leftarrow \mathcal{N}(\mathbf{X}, \mathbf{\Xi}, \mathbf{\Xi}^0)$
4: \hspace{1cm} return $\Gamma$
5: end function

III. MODEL COMPOSITION USING DUAL QUATERNION ALGEBRA

The framework presented in Section II is general, has a high level of abstraction, and thus can be instantiated into different mathematical representations. Here we present one instance based on dual quaternion algebra\(^6\) which is particularly suitable because elements such as unit dual quaternions and pure dual quaternions, when equipped with standard multiplication and addition operations, form Lie groups with associated Lie algebras\(^5\). Furthermore, dual quaternion algebra provides an elegant and efficient representation of homogeneous transformations\(^4\) and screw theory\(^45\). Nonetheless, other representations, such as the spatial algebra\(^5\) or the Lie algebra se\((3)\)\(^4\), might also be used, as long as they capture the high-level operations described in Section II.

Consider the expressions in Section II. If dual quaternions are used, then $\mathbf{M} = \mathbf{T} = \mathcal{H}_p$, where $\mathcal{H}_p$ is the set of pure dual

---

\(^6\)If the configuration of subsystem $p_j$ has not changed, previously calculated and stored values can be used instead of recalculating them.

\(^5\)Basic definitions are shown in Appendix A. For a more comprehensive introduction of dual quaternion algebra, see \[42\].
Algorithm 2 Obtain the twists and their derivatives at the $n$ CoMs of the $s$ subsystems.

1: function DMC_FORWARD_RECURSION($\hat{Q}, \dot{\hat{Q}}, \hat{Q}$)
2: $\Xi_1 \leftarrow (\hat{0}_n, \hat{0}_n), \ldots, \Xi_s \leftarrow (\hat{0}_n, \hat{0}_n)$
3: $\mathbf{X}_{p_1,1} \leftarrow (\hat{0}_n, \hat{0}_n), \ldots, \mathbf{X}_{p_s,s} \leftarrow (\hat{0}_n, \hat{0}_n)$
4: queue $\leftarrow 1$ // Initialize the queue
5: Mark subsystem (1) as visited
6: while queue is not empty do
7:   $i \leftarrow$ pop first element from queue
8:   if subsystem ($i$) is not black box then
9:     $\Xi_{i,j} \leftarrow$ FORWARD_RECURSION($\dot{q}_i, \dot{q}_j$)
10:    $\mathbf{X}_{i,j} \leftarrow \mathbf{X}_{i,j}$
11:   end if
12:   for $j \leftarrow j_1$ to $j_{m_i}$ do
13:     if $j$ is unvisited then
14:       Push $j$ to the end of the queue
15:     end if
16:     if subsystem ($j$) is not black box then
17:       Calculate $\mathbf{X}_{p_j,j}$ using $a_i$ and $q_i \in \mathcal{Q}$
18:     else
19:       Calculate $\mathbf{X}_{p,j}$ using $\Xi_i$
20:       end if
21:       $\mathbf{X}_{i,j} \leftarrow \Xi_{i,j}$
22:     end if
23:     Mark subsystem ($j$) as visited
24:   end for
25: end while
26: (\Xi, \Xi) $\leftarrow$ (\Xi_1, \ldots, \Xi_s)
27: \text{return} (\mathbf{X}, \Xi)
28: end function

Algorithm 3 Obtain the total joint wrenches of the $s$ subsystems.

1: function $\mathcal{X}(\mathbf{X}, \Xi, \Xi)$
2: $\Gamma_1 \leftarrow \hat{0}_n, \ldots, \Gamma_s \leftarrow \hat{0}_n$
3: for $i \leftarrow s$ to 1 do
4:   if subsystem ($i$) is not black box then
5:     $\Gamma_i \leftarrow \Gamma_i + \mathcal{W}(\Xi_i)$, with $\Xi_i \subseteq \Xi \cup \Xi$
6:   end if
7:   if subsystem ($p_i$) is not black box then
8:     Calculate $\mathbf{X}_{p_i,i}$ using $\mathcal{X}_{p_i,i}$ and $\Xi_{p_i,i}$
9:   end if
10: end for
11: $\Gamma_{\text{total}} \leftarrow (\Gamma_1, \ldots, \Gamma_s)$
12: \text{return} $\Gamma_{\text{total}}$
13: end function

quaternions; that is, dual quaternions with real part equal to zero (see Appendix A). Also, $\mathbf{1}_n = 1_n \in \mathbb{R}^n$ and $\mathbf{0}_n = 0_n \in \mathbb{R}^n$ because $0, 1 \in \mathbb{R} \subseteq \mathcal{H}$, and $0 \mathcal{X} = \mathcal{X}0 = 0$ and $1 \mathcal{X} = \mathcal{X}1 = \mathcal{X}$ for any $\mathcal{X} \in \mathcal{H}$. Therefore, the high-level expressions used to propagate twists and wrenches remain unchanged, but the low-level expressions used to express those twists and wrenches in different frames explicitly employ the operations of dual quaternion algebra.

More specifically, the twist $\xi_{a_i} \in \mathcal{H}_p$ at the connection point $a_i$ between the preceding subsystem $p_i$ and current subsystem $i$, propagated to each of the CoMs of subsystem $i$, is given by the vector of twists $\Xi_{p_i,i} \in \mathcal{H}^n_p$ such that

$$\Xi_{p_i,i} = \text{Ad}_{a_i}(\mathbf{X}_{p_i,i}) \xi_{a_i},$$

(6)

where the (vector) adjoint operator $\text{Ad}_{a_i}$ is given by (10) and $\mathbf{X}_{p_i,i} = [\mathbf{X}_{p_i,i}^0, \ldots, \mathbf{X}_{p_i,i}^c_i] \in \mathcal{S}^m$ is the vector of the relative poses (i.e., unit dual quaternions) between the connection point $a_i$ and the $c_1, \ldots, c_m$ CoMs of the $i$th subsystem. The vector of dual quaternions $\Xi_{p_i,i} \in \mathcal{H}^n_p$ is given by the time derivative of (6), where each element is given by [46]

$$\frac{d}{dt} \left( \text{Ad}(\mathbf{X}_{p_i,i}^0) \xi_{a_i} \right) = \text{Ad}(\mathbf{X}_{p_i,i}^0) \xi_{a_i} + \xi_{k_i} \mathcal{X} \left( \text{Ad}(\mathbf{X}_{p_i,i}^0) \xi_{a_i} \right),$$

(7)

for $k \in \{1, \ldots, n_i\}$.

Analogously, the wrench $\mathcal{S}_{b_{i,j}} \in \mathcal{H}_p$ at the connection point $b_{i,j}$ between the current subsystem $i$ and the succeeding subsystem $j$, propagated to each of the CoMs of subsystem $i$, is given by the vector of wrenches $\Xi_{b_{i,j}} \in \mathcal{H}^n_p$ such that

$$\Gamma_{b_{i,j}} = \text{Ad}_{b_{i,j}}(\mathbf{X}_{j,i}) \xi_{b_{i,j}},$$

(8)

in which $\mathbf{X}_{j,i} = [\mathbf{X}_{j,i}^0, 0_{n_i-j}^T] \in \mathcal{H}^n_i$, with $\mathbf{X}_{j,i} = [\mathbf{x}_{b_{i,j}}, \ldots, \mathbf{x}_{b_{i,j}}] \in \mathcal{S}^n$ being the vector of relative poses (i.e., unit dual quaternions) between the connection point $b_{i,j}$ and each of the $\eta_i \leq n_i$ joints of subsystem $i$ that precede $b_{i,j}$. Furthermore, the wrench $\mathcal{S}_{b_{i,j}}$ has the form

$$\mathcal{S}_{b_{i,j}} = \mathcal{S}_{b_{i,j}}^0 + \mathcal{X} \mathcal{X}_{b_{i,j}},$$

where $\mathcal{S}_{b_{i,j}}^0 = f_{z_i} + f_y \hat{k} + f_z \hat{j}$ is the force at the connection point $b_{i,j}$ given by Newton’s second law and $\mathcal{X}_{b_{i,j}} = \mathcal{X}_{b_{i,j}}$.

It is important to notice that $\xi_{a_i}$ in (6) is the twist at connection point $a_i$ with respect to frame $\mathcal{F}_0$, expressed in frame $\mathcal{F}_{a_i}$. Analogously, $\mathcal{S}_{b_{i,j}}$ is the wrench at connection point $b_{i,j}$ with respect to frame $\mathcal{F}_0$, expressed in frame $\mathcal{F}_{b_{i,j}}$. We use three indices here because the twist/wrench between two frames can be seen from a third frame. For instance, $\mathcal{X}_{a,b}^c$ is the twist of frame $\mathcal{F}_b$ with respect to frame $\mathcal{F}_a$, expressed in frame $\mathcal{F}_c$. 
\[ \tau_y \dot{\hat{k}} + \tau_z \dot{\hat{k}} \] is the torque about \( b_{ij} \) due to the change of its angular momentum, given by the Euler’s rotation equation.

Moreover, the function \( \mathcal{W}_i \) is given by (1), in which the sets \( \mathcal{T} \) and \( \mathcal{W} \) are replaced by the set \( \mathcal{H}_p \), and the low-level dynamic equations of serial kinematic chains using dual quaternions are demonstrated in [46]. Algorithms 4–5 and 7 summarize the dual quaternion Newton-Euler formalism.

It is important to highlight that since these algorithms present the low-level dynamics of a subsystem, the indexes used in them correspond to bodies within each subsystem.

**Algorithm 4** Dual Quaternion Newton-Euler Algorithm for a serial mechanism [46]. Vector \( \Gamma \in \mathcal{H}_p \) contains the wrenches at each joint of the \( k \)-DOF serial mechanism, whereas \( \Xi \in \mathcal{H}_p \) is the stacked vector of twists and twists derivatives at the CoM of each link, and \( q, \dot{q}, \ddot{q} \) are the joint configurations, joint velocities, and joint accelerations, whose dimensions depend on the number of parameters used to represent each joint (e.g., each prismatic or revolute joint adds one dimension, each cylindrical joint adds two, each planar, spherical, and helical joints add three, and each 6-DOF joint adds six dimensions).

```
1: function NEWTON_EULER(q, \dot{q}, \ddot{q})
2: \( \Xi \leftarrow \text{FORWARD_RECURSION}(q, \dot{q}, \ddot{q}) \)
3: \( \Gamma \leftarrow \text{BACKWARD_RECURSION}(\Xi) \)
4: return \( \Gamma \)
5: end function
```

**Algorithm 5** Forward recursion to obtain the twists and their derivatives for the CoM of all robot links [46]. Each joint \( q_i = (q_{i1}, \ldots, q_{ik}) \) and its higher time derivatives are represented by a different number of parameters depending on their type.

```
1: function FORWARD_RECURSION(q, \dot{q}, \ddot{q})
2: \( \xi_{0,0}^{ci} \leftarrow 0 \) and \( \xi_{0,0}^{ci} \leftarrow 0 \)
3: for \( i \leftarrow 1 \) to \( k \) do
4: \( \left( \xi_{i-1,1}^{ci}, \xi_{i-1,1}^{ci} \right) \leftarrow \text{JOINT_TWIST}(\xi_i, \dot{\xi}_i) \)
5: \( \Gamma \leftarrow \text{Calculation of the 3rd CoM twist} \)
6: \( \xi_{0,0}^{ci} \leftarrow \text{Ad}(\xi_{i-1}^{ci}) \xi_{i-1,1}^{ci} + \text{Ad}(\xi_{i-1}^{ci}) \xi_{i-1,1}^{ci} + \text{Ad}(\xi_{i-1}^{ci}) \xi_{i-1,1}^{ci} \)
7: \( \xi_{0,0}^{ci} \leftarrow \text{Calculation of the 3rd CoM twist derivative} \)
8: \( \xi_{0,0}^{ci} \leftarrow \text{Ad}(\xi_{i-1}^{ci}) \xi_{i-1,1}^{ci} + \text{Ad}(\xi_{i-1}^{ci}) \xi_{i-1,1}^{ci} + \text{Ad}(\xi_{i-1}^{ci}) \xi_{i-1,1}^{ci} \)
9: end for
10: return \( \Xi \leftarrow \left( \xi_{i-1,1}^{ci}, \ldots, \xi_{i-1,1}^{ci} \right) \)
11: \( \Xi \leftarrow \left[ \Xi \ T \ \Xi \right] \)
12: return \( \Xi \)
13: end function
```

The following example illustrates the application of the proposed modular composition strategy to derive the dynamic model of the subsystems 1 and 2 shown in Fig. 1.

**Algorithm 6** Function to obtain the twists of some of the most commonly used joints in robotics [46].

```
1: function JOINT_TWIST(q_i, \dot{q}_i, \ddot{q}_i)
2: if revolute joint then
3: \( \xi_{i-1,1}^{ci} \leftarrow \omega l_i^{i-1} \) and \( \xi_{i-1,1}^{ci} \leftarrow \omega l_i^{i-1} \)
4: else if prismatic joint then
5: \( \xi_{i-1,1}^{ci} \leftarrow \varepsilon v_i l_i^{i-1} \) and \( \xi_{i-1,1}^{ci} \leftarrow \varepsilon v_i l_i^{i-1} \)
6: else if spherical joint then
7: \( \xi_{i-1,1}^{ci} \leftarrow \omega l_i^{i-1} \) and \( \xi_{i-1,1}^{ci} \leftarrow \omega l_i^{i-1} \)
8: end if
9: end function
```

**Algorithm 7** Backward recursion to obtain the wrenches at the robot joints [46].

```
1: function BACKWARD_RECURSION(\Xi)
2: \( \xi_{0,k+1}^l \leftarrow \text{external_wrench} \)
3: for \( i \leftarrow k \) to 1 do
4: \( \xi_{0,ci}^{ci} \leftarrow \Xi^{ci} \) and \( \xi_{0,ci}^{ci} \leftarrow \Xi^{ci} \)
5: \( f_{0,ci}^{ci} \leftarrow m_i \left( \text{Ad}(\xi_{i-1}^{ci}) + \text{Ad}(\xi_{i-1}^{ci}) \right) \times \left( \text{Ad}^T, \text{Ad}^T \right) \)
6: \( \Gamma_{0,ci}^{ci} \leftarrow \text{Calculation of the \( \Gamma \)} \)
7: \( \Gamma_{0,ci}^{ci} \leftarrow \text{Calculation of the \( \Gamma \)} \)
8: \( \text{Let} \ \Gamma^{ci} \leftarrow \Gamma_{0,ci}^{ci} \)
9: \( \Gamma^{ci} \leftarrow \text{Calculation of the \( \Gamma \)} \)
10: end for
11: \( \text{return} \ \Gamma \)
12: end function
```

**Example 4.** Consider Example 1 illustrated in Fig. 3. If dual quaternions are used, then \( \Gamma_1, \Gamma_2 \in \mathcal{W}^3 = \mathcal{H}_p \). Also,

\[
\Gamma_{2,1}^{ci} \leftarrow \text{Ad}^3 \left( \text{X}_2^{ci}, \text{X}_1^{ci}, \text{X}_0^{ci} \right) \xi_{0,bi,2}^{ci,2}.
\]

where \( \xi_{0,bi,2}^{ci,2} \) is the wrench propagated from subsystem 2 to
subsystem 1 at the connection point \( b_{1,2} \), in which

\[
\mathbf{X}_{2,1} = \begin{bmatrix} \mathbf{X}_{2,1}^T \end{bmatrix}^T = \begin{bmatrix} \mathbf{X}_{b_{1,2}}^0 & \mathbf{X}_{b_{1,2}}^1 \end{bmatrix}^0 \in \mathcal{H}_p^3.
\]

The first two elements of \( \mathbf{X}_{2,1} \) are different from zero because point \( b_{1,2} \) is connected at the second link of subsystem 1. Therefore, \( \mathcal{X}_{b_{1,2}}^{0,1,2} \) does not directly affect its last link. Moreover,

\[
\mathcal{E}_{1,2} = \Lambda d_3 (\mathbf{X}_{1,2}) \mathcal{X}_{0,0,2}^{a_2},
\]

in which \( \mathcal{X}_{0,0,2}^{a_2} \) is the twist propagated from subsystem 1 to subsystem 2 at the connection point \( a_2 \), and

\[
\mathbf{X}_{1,2} = \begin{bmatrix} \mathbf{X}^{e_1}_{0,2} \mathbf{X}^{e_2}_{0,2} \mathbf{X}^{e_3}_{0,2} \end{bmatrix} \in \mathcal{H}_p^3,
\]

where the symbol “\( \tilde{\cdot} \)” indicates the frames \( F_{c_i} \) located at the CoM of each link in subsystem 2 (as opposed to the frames \( F_{c_i} \) located at the CoM of each link in subsystem 1). Lastly,

\[
\mathcal{E}_{2,2} = \begin{bmatrix} \mathcal{E}^{e_1}_{0,2} \mathcal{E}^{e_2}_{0,2} \mathcal{E}^{e_3}_{0,2} \end{bmatrix} \in \mathcal{H}_p^3
\]

is the vector of twists at the CoMs of subsystem 2 that are caused exclusively by the motion of the joints of subsystem 2.

A. Computational complexity

As presented in [46], Algorithm 1 of the dual quaternion Newton-Euler algorithm (dqNE), has a linear cost in the number of DoF of a serial kinematic chain with arbitrary joints. Therefore, each subsystem in the modular composition is calculated with complexity \( O(n_i) \), where \( n_i \) is the number of DoF of the ith subsystem.

Algorithm 2, which calculates the twists and their derivatives at the \( n = \sum_{i=1}^{s} n_i \) CoMs of the \( s \) subsystems, uses a breadth-first search algorithm to traverse all the \( s \) subsystems in the tree. In the worst case when a node is visited, Algorithm 5 is executed once to calculate the forward recursion of dqNE in Line 7 with complexity \( O(n_i) \) and Line 10 is calculated with complexity \( O(2n_i) \) because it correspond to a sum of vectors \( \mathcal{E}_{1,2}, \mathcal{E}_{2,2} \in T^{2n_i} \). Moreover, Line 7 is executed \( s \) times. As such, we have

\[
O(s) + O \left( \sum_{i=1}^{s} n_i \right) + O \left( \sum_{i=1}^{s} 2n_i \right)
\]

\[
= O(s) + O(n) + 2O(n) = O(n),
\]

because \( n \geq s \), as each subsystem has at least one DoF \(^{10}\). Additionally, Line 10 is executed \( s - 1 \) times \(^{11}\) with complexity \( O(n_i) \), as we calculate the \( n_i \) elements of \( \mathbf{X}_{2,1} \). Furthermore, Line 22 is also executed \( s - 1 \) times with complexity \( O(n_i) \) \(^{12}\) while the remaining operations between lines 12 to 27 have constant complexity \( O(1) \) (e.g., dequeuing, reading from a sensor, etc.). Thus, we have

\[
O \left( \sum_{j=2}^{s} n_j \right) + O \left( \sum_{j=2}^{s} n_j \right) + O ((s - 1) \cdot 1)
\]

\[
= O(n - n_1) + O(n - n_1) + O(s - 1) = O(n).
\]

Consequently, the complexity of Algorithm 2 is \( O(n) + O(n) = O(n) \), assuming no black-box subsystems and a centralized computation.

As for Algorithm 3 it traverses matrix \( A \) from right to left. In the worst case, in Line 3 it calls Algorithm 4 \( s \) times with complexity \( O(n_i) \) plus it performs \( s \) additions of elements in \( W^{n_i} \) with complexity \( O(n_i) \). Thus,

\[
O \left( \sum_{i=1}^{s} n_i \right) + O \left( \sum_{i=1}^{s} n_i \right) = O(n) + O(n) = O(n).
\]

Additionally, between lines 7 and 16 there are \( s - 1 \) operations with complexity \( O(n_i) \) in lines 8, 14, and 15 and complexity \( O(1) \) in the remaining lines \(^{13}\). Consequently,

\[
3O \left( \sum_{i=2}^{s} n_i \right) + O(s - 1) = 3O(n - n_1) + O(s - 1) = O(n).
\]

Therefore, the total complexity of Algorithm 3 is \( O(n) + O(n) = O(n) \).

Consequently, the total complexity of Algorithm 1 is \( O(n) + O(n) = O(n) \). On the other hand, the complexity of Newton-Euler-based monolithic approaches is also \( O(n) \) \([6], p. 51\). This means that the proposed Algorithm 1 allows for modular composition and the inclusion of black box subsystems without incurring a higher complexity even in the worst case. Furthermore, it has the potential to be faster than a monolithic solution with the aid of parallelism. For that, there are plenty of works in the literature regarding parallel BFS algorithms \([48], [49], [50]\), which could be applied to Algorithm 2.

Additionally, Algorithm 4 could compute the elements on the columns of matrix \( A \) in parallel for each branch, although information must be synchronized when adding wrenches at points connecting different branches. However, parallelization is out of scope and will be explored in future work.

IV. WRENCH-DRIVEN END-EFFECTOR MOTION CONTROL

Advanced general modeling techniques such as the ones in Sections II and III are valuable on their own. Nonetheless, robot dynamic models are more useful if amenable to control design. Therefore, this section illustrates how control laws can be easily designed when using our proposed formalism.

The dual quaternion Newton-Euler formalism for branched robots can be seen as the function \( \mathcal{N} : \mathcal{S}_p \times H_p^m \times H_p^m \rightarrow H_p^m \) given by

\[
\mathcal{G} = \mathcal{N} (\mathbf{X}, \mathcal{E}, \mathcal{E}, \mathcal{E}) \in H_p^m,
\]

\(^{10}\)We consider the upper bound for the Big O function \([47], p. 47\).

\(^{11}\)In the worse scenario, lines 12 to 27 are calculated for all subsystems, except for root subsystem \( n_1 \).

\(^{12}\)The twist \( \mathcal{X}^{a_{ij}}_{0,0,2} \) is propagated \( 2n_j \) times to generate the vector \( \mathcal{E}_{p_{ij},j} \) \( \in T^{2n_j} \). Moreover, \( O(2n_j) = O(n_j) \).

\(^{13}\)Only the root subsystem has no precedent subsystem. Thus, in the worse case, lines \( 7 \) to 16 are executed for all subsystems but subsystem 1.
where $\mathbf{X}$ is the vector of wrenches at the $n$-DoF branched robot joints, $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]^T \in \mathbb{S}^n$ is the vector containing the poses associated with each CoM, and $\mathbb{H}_n$ are the stacked vector of twists and twist time derivatives at the CoMs, respectively.

The vector $\mathbf{N}$ of the branched robot’s joint wrenches can be decomposed into three components, $\mathbf{N} = \mathbf{N}_M + \mathbf{N}_C + \mathbf{N}_g$, where $\mathbf{N}_M = \mathbf{N}(\mathbf{X}, 0_n, \mathbf{M})$ is the vector of joint wrenches due to inertial components, $\mathbf{N}_C = \mathbf{N}(\mathbf{X}, \mathbf{M}, 0_n)$ is the vector of joint wrenches due to Coriolis and Centrifugal effects, $\mathbf{N}_g = \mathbf{N}(\mathbf{X}, 0_n, 0_n)$ is the vector of joint wrenches due to gravitational effects, and $0_n \in \mathbb{R}^n \subset \mathbb{H}^n$. The function $\mathbf{N}$ in (9) is a natural extension from the one used for serial robots [46].

Similarly to Newton-Euler algorithms for serial kinematic chains [46], given the desired wrenches $\mathbf{\zeta}_{\ell}^{L(l)} \in \mathcal{H}_p$, at the end-effectors of all $\ell$ leaves, where $\mathcal{L}(\ell)$ returns the index of the end-effector frame of the leaf subsystem $l \in \{1, \ldots, \ell\}$, the stacked vector of external wrenches $\mathbf{Z}_e = [\mathbf{z}_{e_1}, \ldots, \mathbf{z}_{e_\ell}] \in \mathcal{H}_p$, can be easily propagated during the backward recursion by letting $\mathbf{z}_{e_{k+1}}^{L(l)} \leftarrow \mathbf{z}_{e_k}^{L(l)}$ in line 2 of Algorithm 7 for the backward recursion in the $l$th leaf subsystem. For that, we first define the function

$$\mathbf{N}(\mathbf{X}, \mathbf{M}) : \mathbb{S}^n \times \mathcal{H}_p \times \mathcal{H}_p \times \mathcal{H}_p \to \mathcal{H}_p,$$

such that $\mathbf{N} = \mathbf{N}(\mathbf{X}, \mathbf{M}, 0_n) \in \mathcal{H}_p$. Similarly to (9), $\mathbf{N}(\mathbf{X}, \mathbf{M}) = \mathbf{N}_M + \mathbf{N}_C + \mathbf{N}_g + \mathbf{N}_Z$, where $\mathbf{N}_M = \mathbf{N}(\mathbf{X}, 0_n, \mathbf{M})$, $\mathbf{N}_C = \mathbf{N}(\mathbf{X}, \mathbf{M}, 0_n)$, $\mathbf{N}_g = \mathbf{N}(\mathbf{X}, 0_n, 0_n)$, and $\mathbf{N}_Z = \mathbf{N}(\mathbf{X}, 0_n, 0_n, \mathbf{Z}_e) - \mathbf{N}_g$.

Given the vector of desired poses $\mathbf{x}_d = [\mathbf{x}_{d_1}, \ldots, \mathbf{x}_{d_\ell}]^T \in \mathbb{S}^l$ at the $\ell$ leaves’ end-effectors, the goal is to design attractive vector fields for $\mathbf{Z}_e \in \mathcal{H}_p$ to ensure that the vector $\mathbf{X} \in \mathbb{S}^l$ of end-effector poses converges to $\mathbf{x}_d$. We design $\mathbf{Z}_e$ by using a straightforward extension of the controller presented in [51]. First, given the $l$th end-effector pose $\mathbf{x}_d$, the desired pose $\mathbf{z}_{d_l}$, and the pose error $\hat{\mathbf{d}}_l$, the $l$th end-effector twist feedback linearizing control input is given by

$$U_l = -k_p \log \hat{\mathbf{d}}_l - k_v \xi_{d_l}^{L(l)} + \Ad(\hat{\mathbf{d}}_l^T) \hat{\xi}_{d_l}^{L(l)} + \left(\Ad(\hat{\mathbf{d}}_l^T) \xi_{d_l}^{L(l)} \right) \times \hat{\xi}_{d_l}^{L(l)},$$

where $\xi_{d_l}^{L(l)} \in \mathcal{H}_p$ is the twist of the $l$th leaf end-effector satisfying $\hat{\mathbf{d}}_l = \frac{1}{\ell} \hat{\mathbf{d}}_l \xi_{d_l}^{L(l)}$, with $k_p, k_v \in (0, \infty)$ being the controller gains, and $\xi_{d_l}^{L(l)}$ and $\xi_{d_l}^{L(l)}$ are the desired $l$th leaf end-effector twist and twist derivatives expressed in the end-effector frame of the $l$th leaf [51]. As demonstrated in [51], $\hat{\mathbf{d}}_l$ converges asymptotically to $1$ when the end-effector error dynamics is given by $\hat{\mathbf{d}}_l = \frac{1}{\ell} \hat{\mathbf{d}}_l U_l$ with $U_l$ defined as in (11), which implies that $\mathbf{z}_e \to \mathbf{z}_{d_l}$ when $l \to \infty$.

Therefore, we define $\mathbf{Z}_e \triangleq \left[\text{swap}(U_1), \ldots, \text{swap}(U_\ell)\right]^T$ to control the $l$ leaves’ end-effectors, where the swap operator defined in Appendix A is used to ensure that the rotational and linear components of $\mathbf{U}_l$ match the rotational and linear components of $\mathbf{\zeta}_{d_l}^{L(l)}$ in $\mathbf{Z}_e$. Using (11) in (10) with $\mathbf{M} = \mathbf{M} = 0_n$, the joint wrench inputs $\mathbf{N}_u$ are given by

$$\mathbf{N}_u \triangleq \mathbf{N} = \mathbf{N}(\mathbf{X}, 0_n, 0_n, \mathbf{Z}_e) = \mathbf{N}_g + \mathbf{N}_Z,$$

where $\mathbf{N}_g$ is the gravity compensation term, and $\mathbf{N}_Z$ is the vector of $n$ joint wrenches induced by the $\ell$ end-effector wrenches $\mathbf{Z}_e$ that enforce the end-effector twist feedback linearizing control inputs $\mathbf{U}_l$ for all $l \in \{1, \ldots, \ell\}$. Since the inertial and Coriolis/centrifugal terms in $\mathbf{N}_{g+l}$ are eliminated, it is equivalent to a feedback-linearizing controller with gravity compensation.

V. NUMERICAL EVALUATION AND SIMULATION RESULTS

To evaluate the accuracy and correctness of the model composition methodology proposed in Sections II and III we performed numerical evaluations using two robots; namely, the fixed-base 24-DoF branched manipulator (BM) shown in Fig. 1 and the 30-DoF holonomic mobile branched manipulator (MBM) shown in Fig. 2. We also include qualitative results to evaluate the wrench-driven end-effector motion control in Section IV.

We implemented the simulations on the robot simulator CoppeliaSim Edu V4.4.0 [52] with the MuJoCo [53] physics engine. The implementation was done in MATLAB 2023b, and the computational library DQ Robotics [54] was used for dual quaternion algebra on a computer running Ubuntu 20.04 LTS 64 bits equipped with an Intel i7-6500u with 8GB RAM.

Figure 5: A 30-DoF holonomic mobile branched manipulator (MBM) composed of three subsystems represented by the colored rigid bodies. The second subsystem (blue) is considered as a black box subsystem.
A. Simulation setup

The BM has eight subsystems, each composed of three DoFs, some containing prismatic or revolute joints. Therefore, the configuration vector is defined as

$$
q_{BM} = [q_1^T, \ldots, q_8^T]^T \in \mathbb{R}^{24},
$$

with $q_1, \ldots, q_8 \in \mathbb{R}^3$.

Although the MBM is composed of three subsystems, the second one is a black box in our simulation. Thus, the generalized coordinates of the MBM were defined as

$$
q_{MBM} \triangleq [\hat{q}_1^T \quad \hat{q}_3^T]^T \in \mathbb{R}^6,
$$

where $\hat{q}_1 \triangleq [x_{\text{base}} \quad y_{\text{base}} \quad \phi_{\text{base}}]^T \in \mathbb{R}^3$ is the configuration vector of subsystem 1 (i.e., the holonomic base), with $x_{\text{base}}$ and $y_{\text{base}}$ being the Cartesian coordinates and $\phi_{\text{base}} \in [0, 2\pi)$ being the angle of rotation of the holonomic base. The vector $\hat{q}_3 \in \mathbb{R}^3$ contains the joint configurations of subsystem 3 in the MBM.

The robots followed arbitrary trajectories in the configuration space, and their configurations ($q_{MBM}$ and $q_{BM}$) and configuration velocities ($\dot{q}_{MBM}$ and $\dot{q}_{BM}$) were read from CoppeliaSim. Since the simulator does not allow the direct reading of accelerations, $\ddot{q}_{MBM}$ and $\ddot{q}_{BM}$ were filtered using a discrete filter and used to obtain the configuration accelerations $\ddot{q}_{MBM}$ and $\ddot{q}_{BM}$ by means of numerical differentiation based on Richardson extrapolation [55] p. 322). Moreover, the generalized torque vectors $\tau_{MBM} \in \mathbb{R}^6$ and $\tau_{BM} \in \mathbb{R}^{24}$ were also read from CoppeliaSim. For the branches, this information was directly obtained from the joints, whereas for the holonomic base we used a force sensor at the connection point with the first link of the second subsystem.

We used Algorithm 1 to obtain the total wrenches, namely $\Gamma_{BM} \in \mathcal{H}_p^{24}$ at the BM’s joints and $\Gamma_{MBM} \in \mathcal{H}_a^{8}$ at the mobile base and joints of MBM’s subsystem 3. Afterward, wrenches $\Gamma_{MBM}$ and $\Gamma_{BM}$ were projected onto the body motion axes 46 to obtain torques at revolute joints and around the vertical axis of the mobile base, and linear forces at prismatic joints of the branched manipulator and linear motion components of the mobile base.

The comparisons between the generalized force waveforms obtained using the dual quaternion Newton-Euler model composition (dqNEMC) ($\tau_{MBM}$ and $\tau_{BM}$) and the ones read from CoppeliaSim were made considering the root mean square error (RMSE) and the coefficient of multiple correlation (CMC) 56 between them. The CMC provides a coefficient ranging between zero and one that indicates how similar two given waveforms are. Identical waveforms have CMC equal to one, whereas completely different waveforms have CMC equal to zero. Furthermore, the CMC formulation 56 focuses on assessing the similarity between waveforms acquired synchronously from different models within movement-cycles when the effect of the model on the waveform similarity is the only variable of interest.

Moreover, we also compared our results with the recursive Newton-Euler algorithm (sv2NE) available in Featherstone’s Spatial v2 package 14 a widely used and well-established library that implements the robot dynamic modeling based on spatial algebra 3. Spatial algebra has been used on real complex robotic platforms, such as humanoids, thanks to its good accuracy and computational performance 57. Therefore, it is a good basis of comparison. However, since the Spatial v2 package does not support either mobile bases or black box subsystems, we considered the sv2NE only for the fixed-base branched robot. For that simulation, the joint torques $\tau_{sv2NE} \in \mathbb{R}^{24}$ were obtained from the sv2NE. Then, we calculated the RMSEs and CMCs between $\tau_{sv2NE}$ and the measured joint torques from CoppeliaSim, as well as the RMSEs and CMCs between the joint torques obtained with sv2NE and the dqNEMC.

B. Model accuracy using the BM

The BM shown in Fig. 1 is composed of eight subsystems. Subsystems 1, 2, 4, 5, 6 and 8 are 3-DoF serial kinematic chains with revolute joints, whereas subsystems 3 and 7 are 3-DoF serial kinematic chains with prismatic joints. Appendix B (see Table IV) presents the kinematic and dynamic information of those subsystems.

The joint robots received sinusoidal position inputs given by $u(t) = 0.01 \sin(2\pi t)$ rad, where $1_{24} \in \mathbb{R}^{24}$ is a vector of ones. The reference $u(t)$ was tracked by CoppeliaSim’s internal joint controllers, and the dqNEMC and the sv2NE then receive the measured values of $q$ and $\dot{q}$. Before being numerically differentiated to obtain $\ddot{q}$, the joint velocities $\dot{q}$ were filtered with a second-order discrete low-pass Butterworth filter with normalized cutoff frequency of 100 Hz to filter out measurement noises introduced by CoppeliaSim.

Table II presents the RMSE and the CMC between the joint torque waveforms obtained using dqNEMC and sv2NE and the values obtained from CoppeliaSim, the baseline. Both dqNEMC and sv2NE presented low mean RMSEs with small standard deviations. Moreover, dqNEMC and sv2NE also had mean and minimum CMC close to one, with small standard deviation, and high maximum CMC, thus indicating high similarity between the joint torque waveform obtained from dqNEMC, sv2NE, and the values from CoppeliaSim. Moreover, the CMCs between the dqNEMC and the sv2NE are all equal to one, and their RMSE is of the order of 10^{-14}.

Furthermore, the dqNEMC is numerically equivalent to the sv2NE, which demonstrates the accuracy of our proposed strategy when compared to Featherstone’s spatial recursive Newton-Euler algorithm. However, it is worth highlighting that the dqNEMC is based on a modular dynamic model of the robot, whereas the sv2NE obtains the joint torques through a monolithic solution (i.e., without considering the existence of subsystems).

For qualitative analysis, Fig. C presents the joint torques obtained using dqNEMC and sv2NE, alongside the CoppeliaSim values, for the minimum, maximum, and intermediate CMCs found during simulations. Even for the smallest value of CMC

14Available at: http://royfeatherstone.org/spatial/v2/
The closer the CMC is to one, the more similar the waveforms are. The CMC for the BM is given by:

\[
\text{CMC}_{i,i} = \frac{1}{\sqrt{\text{RMSE}_{i,i}^2 + \text{RMSE}_{i,i}^2}},
\]

where \(\text{CMC}_{i,i}\) is the CMC for joint \(i\), \(\text{RMSE}_{i,i}\) is the root mean square error for joint \(i\), and \(\text{RMSE}_{i,i}\) is the root mean square error for joint \(i\). This formula assesses the similarity between the waveform obtained from CoppeliaSim and the waveform obtained using our model.

(i.e., 0.9888), the joint torques obtained using our model composition formulation match closely the CoppeliaSim values. The small discrepancies arise from discretization effects, small kinematic and dynamic parameters uncertainties, and unmodeled effects in CoppeliaSim, such as friction, measurement noises, and internal controller dynamics.

C. Model accuracy using the MBM and black-box subsystems

The MBM shown in Fig. 5 is composed of three subsystems, with the second being considered as a black box. The first is a holonomic mobile base, subsystem 2 is the 24-DoF branched manipulator shown in Fig. 4, and subsystem 3 consists of a 3-DoF serial mechanism with prismatic joints. Appendix B (see Table IV) presents the kinematic and dynamic information of those subsystems. Subsystems 1 and 3 do not have access to the internal states of subsystem 2. Before being used in those calculations, only require information from subsystem 1 and subsystem 3, respectively.

The robot followed sinusoidal joint/base reference position trajectories given by \(u(t) = 0.01 \sin(2\pi t) \text{ rad}\), which were tracked by CoppeliaSim’s internal joint/base controllers. Notice that the joints of the black-box subsystem 2 are actuated only to simulate possible internal dynamics. However, subsystems 1 and 3 do not have access to this information or the internal states of subsystem 2. Before being used in the model, information obtained from CoppeliaSim (\(\dot{q}_{p,3}\) and \(q_{p,3}\)) was filtered with a second-order discrete low-pass Butterworth filter with normalized cutoff frequency of 30 Hz.

As explained in Section V-A, the dqNEMC then receives the values of \(q\), \(\dot{q}\), and \(\ddot{q}\). The comparison is made considering the generalized forces \(\tau\) read from the joints and the force sensor.

\[\begin{bmatrix}
\dot{\Theta}_1 \\
\dot{\Theta}_2 \\
\Theta_3
\end{bmatrix} = A' H^4_{p \times 3} \begin{bmatrix}
\dot{W}_1 \\
\dot{W}_2 \\
W_3
\end{bmatrix} + \begin{bmatrix}
\Gamma_{2,1} \\
\Gamma_{1,3} \\
0
\end{bmatrix},
\]

in which \(\Theta_1 \in \mathbb{R}^3 \subset H^4_{p}\) is a vector of zeros. Furthermore, \(\mathcal{W}_i = \mathcal{W}_i(\Xi_{i,i})\) when \(i = 1\), and \(\mathcal{W}_i = \mathcal{W}_i(\Xi_{p,i} + \Xi_{i,i})\) when \(i = 3\), with \(p_2 = 1\) and \(p_3 = 2\). Notice that, although subsystem 2 is a black box, the wrench \(\mathcal{W}_{p,2} = \Xi_{b,1,2}\) at the connection point with subsystem 1 is available through direct measurements from a six-axis force sensor.

The twist \(\xi_{p,3}\) and twist derivative \(\dot{\xi}_{p,3}\) at the connection point between subsystems 2 and 3, which are necessary to calculate \(\Xi_{p,3}\), were obtained directly from CoppeliaSim for simplicity. Nonetheless, this information could be either measured through appropriate sensors or communicated by subsystem 2 after its internal calculation. Therefore, \(\Gamma_{2,1}\) is calculated using \(\Xi_{b,1,2}\) and \(\Xi_{p,3}\) is calculated using \(\xi_{p,3}\) and \(\dot{\xi}_{p,3}\), where \(X_{2,1}\) and \(X_{p,3}\) used in those calculations only require information from subsystem 1 and subsystem 3, respectively.

The information propagated by the Newton-Euler algorithm relates to joint actuation torques rather than to reaction torques at the joints. Thus, the torque \(\dot{\Theta}_{b,1,2}\) is the opposite of the value read from the sensor (i.e., \(\dot{\Theta}_{b,1,2} = -\dot{\Theta}_{\text{sens}}\)).

Because subsystem 2 is a black box, its internal states cannot be accessed. Nonetheless, the connection points can be regarded as the outputs of black-box subsystems.
at the base.

Table III presents the RMSE and the CMC between the joint torque waveforms obtained through the dqNEMC and the values obtained from CoppeliaSim. As with the previous simulation, the dqNEMC presented low RMSEs and mean and minimum CMC close to one, with small standard deviation, and high maximum CMC, thus indicating high similarity between the joint torque waveform obtained from dqNEMC and the values from CoppeliaSim.

For qualitative analysis, Fig. 8 presents the joint torques obtained using dqNEMC alongside the CoppeliaSim values, for the minimum, maximum, and intermediate CMCs found during simulations. Even for the smallest value of CMC (i.e., 0.9498), the joint torques obtained using our model composition formulation match closely the CoppeliaSim values. The small discrepancies arise from discretization, unmodeled effects in CoppeliaSim, and small uncertainties in the kinematic and dynamic models.

![Figure 8: Torque waveforms for three joints of the MBM. Solid curves correspond to the CoppeliaSim values, whereas dashed curves with circle markers correspond to the values obtained using the dqNEMC for the joint torque waveforms of the first (CMC = 0.9879) and second (CMC = 0.9498) joints of subsystem 3 and the generalized force along the z-axis of subsystem 1 (CMC = 1.0000).](image)

D. Closed-loop control of the MBM

Consider the MBM shown in Fig. 5. However, rather than considering the whole BM as a black box subsystem, we consider all its eight subsystems explicitly as we want to control the robot. As a result, the MBM consists of 10 subsystems in this simulation. Subsystem 1 in Fig 5 is the root node and is attached to subsystem 1 from the BM, which is now labeled subsystem 2. Consequently, all subsystem labels shown in Fig. 1 are increased by one. Finally, subsystem 10 is attached to the first link of the new subsystem 3 (previously labeled subsystem 2 in Fig. 1).

Using (12), we control the end-effector pose of the leaf subsystems of the MBM. Subsystems 4, 5, 9, and 10 received desired end-effector poses within their workspace, whereas desired pose of subsystem 7 was given by its initial end-effector pose.

Fig. 8 shows the norm of the pose error for all leaf subsystems’ end-effectors, each given by \( \| \tilde{y}_l \|_2 \) with \( \tilde{y}_l = \text{vec}_0 (\log \tilde{e}_l) \) for all \( l \in \{4, 5, 7, 9, 10\} \), and the norm of the total error given by \( \| \hat{y} \|_2 \) where

\[
\hat{y} = \begin{bmatrix} y_4^T & y_5^T & y_7^T & y_9^T & y_{10}^T \end{bmatrix}^T. 
\]

The oscillatory behavior of the error response is expected because the desired closed-loop error dynamics for the end-effector poses due to control law (11) is described by a second-order system (51). Nonetheless, the error for the end-effector poses of all subsystems decay and achieve steady-state.

VI. Conclusions

This paper has presented a modular composition strategy for the dynamic modeling of branched robots that provides a high level of abstraction and enables combining the dynamics of simpler mechanisms to obtain the whole-body dynamics. The proposed formalism requires only twists, twist time derivatives, and wrenches at the connection points between different subsystems to find the coupled dynamics of the combined mechanisms. Thus, distinct from other approaches in the literature, our strategy works even when subsystems are black boxes, as long as the required information at the connection points is known, which can be done through sensor readings. This property is particularly appealing in modular robotics, where different independent kinematic structures can be arbitrarily combined, making the preprogramming of dynamic modeling equations for the whole system impractical. Furthermore, we also proposed a graph representation for the complete branched mechanism, where each vertex is an open kinematic chain, and the wrenches at the joints result from the graph interconnection matrix. This representation enables obtaining the dynamics of the whole system through straightforward algebraic operations. Moreover, all these features are achieved, in the worst case, with the same linear complexity in the number of DoFs as well-established monolithic recursive Newton-Euler algorithms. Additionally, we have presented a formulation for wrench-driven end-effector motion control to illustrate the applicability of the model obtained through the recursive equations of the dynamic model decomposition.
Simulation results have shown that our strategy is numerically equivalent to monolithic solutions, such as Featherstone’s Spatial v2 Newton-Euler algorithm, whose dynamic model is built considering the whole open kinematic tree at once and assumes full knowledge of twists, twists derivatives, and wrenches acting on all rigid bodies in the multibody system. Those results have also shown that our formalism can be effectively used to obtain the dynamic model of a branched mobile manipulator containing a large black box subsystem. Indeed, the values of joint torques in non-black systems closely matched the ones given by the simulator, as attested by the logarithmic mapping is defined as \( \log \mathbf{p} = (\phi m + \mathbf{p}) / 2 \).

The set \( H = \{ h \in H : \| h \| = 1 \} \) of pure dual quaternions is used to represent twists and wrenches, which are represented in different coordinate systems using the adjoint operator \( \text{Ad} : \mathcal{S} \times H \rightarrow H \).

The set \( \mathcal{S} = \{ s \in S : s = (s_a + \varepsilon s_b a) \} \) of pure dual quaternions is used to represent twists and wrenches, which are represented in different coordinate systems using the adjoint operator \( \text{Ad} : \mathcal{S} \times H \rightarrow H \).

The set \( \mathcal{S} = \{ s \in S : s = (s_a + \varepsilon s_b a) \} \) of pure dual quaternions is used to represent twists and wrenches, which are represented in different coordinate systems using the adjoint operator \( \text{Ad} : \mathcal{S} \times H \rightarrow H \).

**APPENDIX A**

**DUAL QUATERNION ALGEBRA**

Dual quaternions \([5]\) are elements of the set

\[
\mathcal{H} \triangleq \{ h_P + \varepsilon h_D : h_P, h_D \in H, \varepsilon \neq 0, \varepsilon^2 = 0 \},
\]

where \( H \triangleq \{ h_1 + i h_2 + j h_3 + k h_4 : h_1, h_2, h_3, h_4 \in \mathbb{R} \} \) is the set of quaternions, where \( i, j, k \) are imaginary units with the properties \( i^2 = j^2 = k^2 = i j k = -1 \) \([5]\).

Addition and multiplication of dual quaternions are analogous to their counterparts of real and complex numbers. One must only respect the properties of the dual unit \( \varepsilon \) and imaginary units \( i, j, k \). Given \( h \in H \) with \( h = h_P + \varepsilon h_D \), we define swap \( : H \rightarrow H \) such that \( h_{p} \equiv h_{d} + \varepsilon h_\varepsilon h \).

The subset \( \mathcal{S} = \{ s \in \mathcal{H} : \| s \| = 1 \} \) of unit dual quaternions, where \( \| s \| = \sqrt{h_h^2 + h_h^2} + h_h^2 \), with \( h_a \) being the conjugate of \( h \) \([42]\), is used to represent poses (position and orientation) in the three-dimensional space and form the group \( \text{Spin}(3) \times \mathbb{R}^3 \) under the multiplication operation. Any \( \mathbf{g} \in \mathcal{S} \) can always be written as \( \mathbf{g} = r + \varepsilon (1/2) p r \), where \( p = p_\varepsilon + \varepsilon p_\varepsilon \) represents the position \( \langle x, y, z \rangle \) and \( r = \cos(\phi/2) + \mathbf{n} \sin(\phi/2) \) represents a rotation, in which \( \phi \in [0, 2\pi) \) is the rotation angle around the rotation axis \( \mathbf{n} \in \mathbb{H} \cap \mathbb{S}^2 \), with \( \mathbb{H} \triangleq \{ h \in H : \text{Re}(h) = 0 \} \), where \( \text{Re}(h) + i h_2 + j h_3 + k h_4 \) \( h_1 \), and \( \mathbb{S}^2 = \{ h \in H : \| h \| = 1 \} \) \([5]\). Given \( \mathbf{g} \in \mathcal{S} \), the logarithmic mapping is defined as \( \log \mathbf{g} = (\phi m + \mathbf{p}) / 2 \).

**REFERENCES**

[1] B. Siciliano, L. Sciavicco, L. Villani, and G. Oriolo, *Robotics*. London: Springer London, 2009. [Online]. Available: [http://link.springer.com/10.1007/978-1-84628-642-1](http://link.springer.com/10.1007/978-1-84628-642-1).

[2] M. W. Spong, S. Hutchinson, and M. Vidyasagar, *Robot Modeling and Control*. New York, NY: Wiley, 2006.

[3] R. Featherstone, *Rigid Body Dynamics Algorithms*. New York, NY: Springer, 2008.
Table IV: Kinematic and dynamic information of the robots used in simulation.

| Link | DH Parameters | CoM | Inertia tensor (\(T = (\mathbf{I}_x, \mathbf{I}_y, \mathbf{I}_z) \in \mathbb{R}^3\)) |
|------|---------------|-----|--------------------------------------------------|
| 3-DoF manipulators with revolute/prismatic joints |
| 1 | \(\alpha_1 = 0\), \(d_1 = 0.187\), \(\alpha_2 = -0.187\), \(\theta_1 = 0.80\) | \(\mathbf{I}_x = 0.80i\), \(\mathbf{I}_y = 0.80j\), \(\mathbf{I}_z = 0.80k\) |
| 2 | \(\alpha_1 = 0\), \(d_1 = 0.43\), \(\alpha_2 = -0.195\), \(\theta_1 = 0.50\) | \(\mathbf{I}_x = 0.50i\), \(\mathbf{I}_y = 0.50j\), \(\mathbf{I}_z = 0.50k\) |
| 3 | \(\alpha_1 = 0\), \(d_1 = 0\), \(\alpha_2 = 0.235\), \(\theta_1 = 0.10\) | \(\mathbf{I}_x = 0.10i\), \(\mathbf{I}_y = 0.10j\), \(\mathbf{I}_z = 0.10k\) |
| Holonomic mobile base |

| N/A | N/A | N/A | N/A | N/A | 0 | 80 | \(\mathbf{I}_x = 40i\), \(\mathbf{I}_y = 40j\), \(\mathbf{I}_z = 40k\) |

A. M. Shafei and H. R. Shafei, “Dynamic modeling of tree-type robotic systems by combining \(3 \times 3\) $3\times3$ rotation and \(4 \times 4\) $4\times4$ transformation matrices,” Multibody System Dynamics, vol. 44, no. 4, pp. 367–395, Dec. 2018. [Online]. Available: http://link.springer.com/article/10.1007/s11081-010-9206-4

B. Siciliano and O. Khatib, Eds. New York, NY: Springer New York, 2005.

C. Tan, H. Zhao, and H. Ding, “Non-redundant inertial parameters determination for dynamic identification of branched articulated robots,” Industrial Robot: the international journal of robotics research and application, vol. 49, no. 6, pp. 1229–1241, Sep. 2022. [Online]. Available: https://www.emerald.com/insight/content/doi/10.1108/IR-12-2021-0296/full/html

D. Gries and F. B. Schneider, Eds. New York, NY: Springer New York, 2005.

E. J. Haug, “Topological Analysis of Multibody Systems for Recursive Dynamics Formulations,” Theoretic Models,” IEEE Transactions on Systems, Man, and Cybernetics, vol. 20, no. 5, pp. 1025–1048, 1990. [Online]. Available: http://ieeexplore.ieee.org/document/59967/

F. C. Park, B. Kim, J. Hong, “Geometric Algorithms for Robotic Manipulation,” CRC Press, 1994.

G. Baciu, J. Chou, and H. Kesavan, “Constrained multibody systems: an approach to vector dynamics,” Mechanism and Machine Theory, vol. 10, no. 1, pp. 57–75, Feb. 1975. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/0094114X88901097

H. K. Kesavan, “Multi-body systems with open chains: Graph-theoretic models,” Mechanism and Machine Theory, vol. 21, no. 3, pp. 273–284, Jan. 1986. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/0094114X86901035

I. Neubert and H. Lipson, “Soldercubes: A self-soldering self-reconfiguring modular robot system,” Autonomous Robots, vol. 40, no. 10, pp. 139–158, Jan. 2016. [Online]. Available: http://link.springer.com/10.1007/978-1-07295-303-0-3

J. M. Selig, Geometric Fundamentals of Robotics, D. Gries and F. B. Schneider, Eds. New York, NY: Springer New York, 2005.

K. Wang, P. Yin, H. Yang, and L. Wang, “Motion planning of rigid--flexible coupled robot,” International Journal of Advanced Robotic Systems, vol. 15, no. 3, p. 172988141877281, May 2018. [Online]. Available: http://www.journals.sagepub.com/doi/10.1177/1729881418772815

L. Wang, Y. Zhao, C. Wei, and D. An, “Multi-Branch Cellular Space Robot Mechanism Design and Climbing Behavior Research,” Journal of Mechanisms and Robotics, vol. 15, no. 5, p. 051016, Oct. 2023. [Online]. Available: https://asmedi digitalcollection.asme.org/mechanismsrobotics/article/15/5/051016/1155866/Multi-Branch-Cellular-Space-Robot-Mechanism-Design

M. Ahmadizadeh, A. Shafei, and M. Fooladi, “Dynamic analysis of multiple inclined and frictional impact–contacts in multi-branch robotic systems,” Applied Mathematical Modelling, vol. 91, pp. 24–42, Mar. 2021. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0307904X20305370

N. X. Liu, B. You, R. Wang, Y. Zhao, C. Wei, and D. An, “Multi-Branch Cellular Space Robot Mechanism Design and Climbing Behavior Research,” Journal of Mechanisms and Robotics, vol. 15, no. 5, p. 051016, Oct. 2023. [Online]. Available: https://asmedi digitalcollection.asme.org/mechanismsrobotics/article/15/5/051016/1155866/Multi-Branch-Cellular-Space-Robot-Mechanism-Design

O. Khatib, Ed. New York, NY: Springer New York, 2005.

P. N. Sheth and J. J. Uicker, “IMP (Integrated Mechanisms Program), A Computer-Aided Design Analysis System for Mechanisms and Linkage,” Journal of Engineering for Industry, vol. 94, no. 2, pp. 454–464, May 1972. [Online]. Available: http://asmedigitalcollection.asme.org/manufacturingscience/article/94/2/454/1432780/IMP-I ntegrated-Mechanisms-Program-A-ComputerAidedDesign-Analysis-System-for-Mechanism-and-Linkage

Q. Liu, B. You, R. Wang, Y. Zhao, C. Wei, and D. An, “Multi-Branch Cellular Space Robot Mechanism Design and Climbing Behavior Research,” Journal of Mechanisms and Robotics, vol. 15, no. 5, p. 051016, Oct. 2023. [Online]. Available: https://asmedi digitalcollection.asme.org/mechanismsrobotics/article/15/5/051016/1155866/Multi-Branch-Cellular-Space-Robot-Mechanism-Design

R. S. Hwang and E. J. Haug, “Topological Analysis of Multibody Systems for Recursive Dynamics Formulations,” Mechanisms and Structures
and Machines, vol. 17, no. 2, pp. 239–258, Jun. 1989. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/15397978909412818

[32] M. Althoff, A. Giusti, S. B. Liu, and A. Pereira, “Effortless creation of safe robots from modules through self-programming and self-verification,” Science Robotics, vol. 4, no. 31, p. eaaw1924, Jun. 2019. [Online]. Available: https://science.org/doi/10.1126/scirobotics.aaw1924

[33] A. Jain, “Multibody graph transformations and analysis,” Nonlinear Dynamics, vol. 67, no. 4, pp. 2779–2797, Mar. 2012. [Online]. Available: http://link.springer.com/article/10.1007/s11071-011-0188-y

[34] J. McPhee, C. Schmitke, and S. Redmond, “Dynamic Modelling of Mechatronic Multibody Systems With Symbolic Computing and Linear Graph Theory,” Mathematical and Computer Modelling of Dynamic Systems, vol. 10, no. 1, pp. 1–23, Mar. 2004. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/13504860408633180

[35] R. M. M. Orsino and T. A. Hess-Coelho, “A contribution on the modular modelling of multibody systems,” Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences, vol. 471, no. 2183, p. 20150080, Nov. 2015. [Online]. Available: https://royalsocietypublishing.org/doi/10.1098/rspa.2015.0080

[36] R. M. M. Orsino, “Recursive modular modelling methodology for lumped-parameter dynamic systems,” Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences, vol. 473, no. 2204, p. 20160891, 2017. [Online]. Available: https://royalsocietypublishing.org/doi/10.1098/rspa.2016.0891

[37] S. Kumar, K. A. V. Szadkowski, A. Mueller, and F. Kirchner, “An Analytical and Modular Software Workbench for Solving Kinematics and Dynamics of Series-Parallel Hybrid Robots,” Journal of Mechanisms, vol. 12, no. 2, p. 021114, Apr. 2020. [Online]. Available: https://asmedigitalcollection.asme.org/mechanismsrobotics/article/doi/10.1115/1.4059417/10274717/An-Analytical-and-Modular-Software-Workbench-for-

[38] T. A. Hess-Coelho, R. M. M. Orsino, and F. Malvezzi, “Modular modelling methodology applied to the dynamic analysis of parallel mechanisms,” Mechanism and Machine Theory, vol. 161, p. 104332, Jul. 2021. [Online]. Available: https://doi.org/10.1016/j.mechmachtheory.2021.104332

[39] A. Müller, “A note on the motion representation and configuration update in time-stepping schemes for the constrained rigid body,” BIT Numerical Mathematics, vol. 56, no. 3, pp. 995–1015, Sep. 2016. [Online]. Available: http://link.springer.com/10.1007/s10543-015-0580-y

[40] J. Yang, H. Peng, W. Zhou, J. Zhang, and Z. Wu, “A modular approach for dynamic modeling of multisegment continuum robots,” Mechanism and Machine Theory, vol. 165, p. 104429, Nov. 2021. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0094114X21001877

[41] J. Y. S. Luh, M. W. Walker, and R. P. C. Paul, “On-Line Computational Scheme for Mechanical Manipulators,” Journal of Dynamic Systems, Measurement, and Control, vol. 102, no. 2, pp. 69–76, Jun. 1989. [Online]. Available: https://asmedigitalcollection.asme.org/dynamicsystems/article/doi/10.1115/1.4059417/10274717/On-Line-Computational-Scheme-for-Mechanical-

[42] B. V. Adorno, “Robot Kinematic Modeling and Control Based on Dual Quaternion Algebra – Part I: Fundamentals,” Tech. Rep., 2017. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01478223v1

[43] S. M. LaValle, Planning Algorithms. Cambridge: Cambridge University Press, 2006. [Online]. Available: http://planning.cs.uiuc.edu/

[44] J. Funder, R. H. Taylor, and R. P. Paul, “On Homogeneous Transforms, Quaternions, and Computational Efficiency,” IEEE Transactions on Robotics and Automation, vol. 6, no. 3, pp. 382–388, 1990.

[45] N. A. Aspragathos and J. K. Dimitros, “A comparative study of three methods for robot kinematics,” IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, vol. 28, no. 2, pp. 135–145, 1998.

[46] F. F. A. Silva, J. J. Quiroz-Omaña, and B. V. Adorno, “Dynamics of Mobile Manipulators using Dual Quaternion Algebra,” Journal of Mechanisms and Robotics, vol. 14, no. 6, p. 11, Jan. 2022. [Online]. Available: https://arxiv.org/abs/2007.08444

[47] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, Introduction to Algorithms. London: The MIT Press, 2009.

[48] H. Gazit and G. L. Miller, “An improved parallel algorithm that computes the BFS numbering of a directed graph,” Information Processing Letters, vol. 26, no. 2, pp. 61–65, Jun. 1988.

[49] C. E. Leiserson and T. B. Scharf, “A work-efficient parallel breadth-first search algorithm (or how to cope with the nondeterminism of reducers),” in Proceedings of the Twenty-Second Annual ACM Symposium on Parallelism in Algorithms and Architectures. Thira Santorini Greece: ACM, Jun. 2010, pp. 303–314.

[50] A. Buluç and K. Madduri, “Parallel breadth-first search on distributed memory systems,” in Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis. Seattle Washington: ACM, Nov. 2011, pp. 1–12.

[51] F. F. A. Silva and B. V. Adorno, “Whole-body Control of a Mobile Manipulator Using Feedback Linearization and Dual Quaternion Algebra,” Journal of Intelligent & Robotic Systems, vol. 91, no. 2, pp. 249–262, Aug. 2018. [Online]. Available: https://link.springer.com/article/10.1007/s10846-017-0686-3

[52] A. F. Roemer, S. P. N. Singh, and M. Freese. “V-REP: A versatile and scalable robot simulation framework,” in 2013 IEEE/RSJ International Conference on Intelligent Robots and Systems. Vilamoura-Algarve, Portugal: IEEE, Oct. 2012, pp. 5026–5033. [Online]. Available: http://ieeexplore.ieee.org/document/6696520/

[53] E. Todorov, T. Erez, and Y. Tassa. “MuJoCo: A physics engine for model-based control,” in 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems. Vilamoura-Algarve, Portugal: IEEE, Oct. 2012, pp. 5026–5033. [Online]. Available: http://ieeexplore.ieee.org/document/6386109/

[54] B. V. Adorno and M. Marques Marinho. “DQ Robotics: A Library for Robot Modeling and Control,” IEEE Robotics & Automation Magazine, vol. 28, no. 3, pp. 102–116, Sep. 2021. [Online]. Available: https://ieeexplore.ieee.org/document/9136790/

[55] A. Gilat and V. Subramaniam. Numerical Methods for Engineers and Scientists: An Introduction with Applications Using MATLAB, L. Ratts and H. Ellis, Eds. Hoboken, USA: John Wiley & Sons, 2014.

[56] A. Ferrari, A. G. Cutti, and A. Cappello. “A new formulation of the coefficient of multiple correlation to assess the similarity of waveforms measured synchronously by different motion analysis protocols,” Gait & Posture, vol. 31, no. 4, pp. 540–542, Apr. 2010. [Online]. Available: http://dx.doi.org/10.1016/j.gaitpost.2010.02.009

[57] K. Bouyarmane, K. Chappellot, J. Vaillant, and A. Kheddar. “Quadratic Programming for Multirobot and Task-Space Force Control,” IEEE Transactions on Robotics, vol. 35, no. 1, pp. 64–77, Feb. 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8328498/

[58] B. V. Adorno. “Two-arm Manipulation: From Manipulators to Enhanced Human-Robot Collaboration [Contribution à la manipulation à deux bras]: Des manipulateurs à la collaboration homme-robot,” Ph.D. dissertation, Université de Montpellier, 2011.