Exact derivative propagation method to compute the generalized compliance matrix for continuum robots: Application to concentric tubes continuum robots

Guillaume Lods, Benoit Rosa, Bernard Bayle, Florent Nageotte

Available online at www.sciencedirect.com

ScienceDirect

Mechanism and Machine Theory 00 (2025) 1–32

MMT

Exact Derivative Propagation Method to compute the Generalized Compliance Matrix for Continuum Robots: Application to Concentric Tubes Continuum Robots

Guillaume Lods[1,*], Benoit Rosa[1], Bernard Bayle[1], Florent Nageotte[1]

[1] *ICube Laboratory, UMR 7357 CNRS - University of Strasbourg, France*

Abstract

This paper introduces the concept of Generalized Compliance for continuum robots, specifically for those modeled with the Cosserat Rod theory. Unlike existing models based on tip compliance, the proposed approach considers interactions along the entire body of the flexible robot. The paper also presents a novel method referred to as the Low-Level Derivative Propagation Method, which is designed for the computationally efficient derivation of the Generalized Compliance matrix. The proposed method streamlines calculations and reduces integration time. The presented method, which is general and applies to various types of continuum robot models, is demonstrated on the case of a Concentric Tubes Continuum Robot. We provide detailed derivations of the equations and computation techniques leading to the derivation of the Generalized Compliance matrix, as well as a large-scale numerical validation of the method. The code used in this paper will be made public upon acceptance of the paper.

Keywords: Continuum robots, Differential kinematics, Generalized Compliance, Cosserat theory, Derivative Propagation Method.

1. Introduction

Continuum robots are thin, slender, flexible structures that conform to curvilinear paths and are intrinsically compliant. In opposition to standard rigid-link robots, their mechanical structure allows them to bend passively upon contact with the environment, which is a strong asset in medical contexts. For this reason, the development of continuum robots for medical and surgical applications has been a very active area of research in the last two decades [1], with applications in laryngeal surgery [2], urology [3], gastrointestinal surgery [4], brain surgery [5], and many others [6]. Various mechanical structures of continuum robots have been proposed in the literature: Concentric Tubes Continuum Robots (CTCR) [7, 8], Tendon-Actuated Continuum Robots (TACR) [9, 10], Concentric Push-Pull Robots (CPPR) [11], Multi-backbone Continuum Robots (MBCR) [12] and Magnetic Continuum Robots (MCR) [13]. Each of these mechanical structures stands out for its ability to be miniaturized, the mechanical stability of shape control, the diversity of the reachable shapes, or also by their ease of manufacturing [1].

In order to control such continuum robots, models linking the joint motor input to the robot shape are required. Historically, continuum robot models were based on geometrical features (e.g. the so-called constant curvature model [14]) and have then evolved to more advanced models that take distributed and tip loads into account [15].

*Corresponding author.
Email address: g.lods@unistra.fr (Guillaume Lods)

Such models, however, are more complex and require numerical resolution methods. Computing differential kinematics is also challenging, and the robot Jacobian calculation is often derived under some constant curvature assumptions [2, 16], or by using approximation schemes [17, 18].

A correct robot Jacobian enables the formulation of a control algorithm to follow a given trajectory in the absence of external forces. However, when external forces are exerted either along the shaft or at the tip of continuum robots, it becomes imperative to consider these forces, given that such structures undergo significant deflection under load. For this reason, continuum robot models have been developed including both distributed forces along the robot and tip forces [7, 8, 19, 20]. Such models can be used for estimating applied loads on the robot using an external measurement such as vision sensors [21, 22] or fiber Bragg gratings [23], for instance. The Compliance Matrix, the inverse of the stiffness matrix, is very useful to quantify the interaction between force and shape. It defines the variation of the robot shape resulting from a variation of the wrench applied to the robot. There has been recently a growing interest in the use of the Compliance Matrix for the stiffness control of continuum robots, the design of their mechanical structures and in the development of force sensing algorithms.

Stella et al. [24] designed and experimentally assessed a control algorithm allowing a TACR to have a prescribed stiffness at the end-effector. This was made possible by coupling the robot Jacobian with the end-effector Compliance Matrix in the control strategy. In the context of MBCR, Bajo et al. proposed a hybrid force/position control framework exploiting specific features of the MBCR structure [25]. Another approach for Compliance control, inspired by the concept of manipulability, consists in analyzing Compliance ellipsoids. This approach has been applied to pneumatic-driven soft robots in [26] and to CTCR in [27]. Since compliance relates externally applied forces to robot deformation, the Compliance Matrix can be used indirectly in a force-based control algorithm. This idea was developed and experimentally validated by Black et al. [28] on a Parallel Continuum Robot where the Compliance Matrix is used for estimating the force applied by the end-effector. An innovative use of compliance is also presented in [29], where a Compliance distribution index provides indications for the mechanical design of a soft rod. By analyzing this index, one can find a singular material distribution to carry actuating forces from the base to the tip while achieving a given tip displacement.

However, to the best of the authors' knowledge, formal definitions of the Compliance Matrix only consider force applied at the tip, which can be seen as a historical legacy of the rigid-links robots models. The existing Compliance Matrix representation is therefore not well adapted to the modeling of continuum robots [7, 8, 19, 20] since non-tip forces have to be considered [21, 22, 23]. This observation has led to the first contribution of this article: to propose a generalized definition of compliance for which the forces producing robot deformations are no longer restricted to the tip but can be applied anywhere along the robot's shaft.

A standard numeric technique to compute the Compliance Matrix is to do an approximation using finite differences on the robot model [17, 18]. Many papers in the literature have proposed to define the compliance by a set of N matrices computing the locally-linearized displacement $\delta T(s)$ at arc-length s under a tip wrench (force and/or torque). This was done, for instance, in [26, 30, 31]. It is worth pointing out here that this first extension of the concept of compliance matrix already brings a layer of complexity. Computing the arc-length discretized compliance using finite difference schemes requires 6 evaluations of the continuum robot BVP (Boundary Value Problem), each of them requiring several evaluations of the IVP (Initial Value Problem). This computational problem has been pointed out by Rucker et al. and was the main reason for developing a derivative propagation method in [30]. In our paper, we go one step further by relaxing the assumption of "tip forces only".

In this paper the models considered are the most advanced ones built on the Cosserat rod theory [32] as it is done for instance in [7, 8, 9, 10, 19, 20, 21, 33]. These models use the set of Cosserat Ordinary Differential Equations (ODE) to compute the propagation of stress and strain along the robot neutral fiber. Two approaches are possible: one can consider the whole continuum robot as an unique Cosserat rod [31, 34, 35, 36] or one can consider multiple Cosserat rods (one for each interacting rod) and rearrange the equations to obtain a robot model that is more faithful to the robot internal structure specificities [8, 10, 37, 38]. For the unique Cosserat rod case, solving the model consists in integrating once an Initial Value Problem (IVP) from on side of the robot to the other. For the multiple Cosserat rods case, the known boundary conditions of the ODE are split between the two ends of the robot. Therefore solving the model consists in solving a Boundary Value Problem (BVP) using a shooting method [39, 40] which optimizes a residual computed at each step using the IVP. Since using a finite differences approximation on the robot model can be time-consuming, some analytical methods have been developed in order to speed up the Compliance Matrix computation.

Smoljkic et al. have proposed a method for the unique Cosserat rod case which consists in expanding the ODE with the tangent problem (i.e. applying directly the partial derivative w.r.t. the tip wrench on the ODE), which yields the tip Compliance matrix after an additional integration [31]. Such a method, called a Derivative Propagation Method (DPM), ensures a high computing efficiency and is geometrically exact. However it is not exploitable for the multiple Cosserat rods case since the optimization residual is not involved in the partial derivatives propagation. This limitation has been overcome by Rucker et al. with another formulation of the DPM [30]. It differs from [31] because the partial derivatives are applied on three extra matrices and not directly on the robot model. After integrating the new equations, the partial derivative of the residual is included in the final tip Compliance computation. This method has opened up promising avenues for active structures such as MBCR, CTCR, TACR and CPPR. Nevertheless the computation time has been increased, while still only considering forces applied at the tip.

Because these DPM have been initially developed for the tip Compliance Matrix, they are not adapted for the Generalized Compliance matrix for which it is required to compute as many derivators as there are points on the robot. This issue has led to the second contribution of this paper: to develop a new DPM specially designed for the new definition of the Compliance Matrix. This method combines the best of the two existing methods [31, 30] and incorporates some mathematical optimizations to keep computation time low.

The paper is organized as follows. In section 2, after presenting the necessary mathematical derivations for Cosserat-based continuum robot modeling, we show how the method proposed in [30] can be adapted to the Generalized Compliance matrix computation, as well as the limitations posed by such a direct adaptation in terms of computational efficiency. We then propose an alternative formulation, termed as Low-Level Derivative Propagation Method (LLDPM). In section 3 we detail an application of the proposed LLDPM to a CTCR model, including the full derivation of the method for easy replication by interested readers. A large-scale numerical validation as well as a computation time analysis is then carried out in Section 4 to show the accuracy and performances of the proposed method. Finally, some interesting properties of the Generalized Compliance matrix are discussed in Section 5 and the conclusions and perspectives of this work are drawn in Section 6.

2. Computing Generalized Compliance Matrix on Continuum Robots

The following sections present the necessary notations. To enhance readability, all notations are listed in Appendix A.

2.1. Preliminaries: Continuum Robot Model

Let us consider a continuum robot, illustrated in Fig 1a, whose neutral axis is parameterized by a bounded curvilinear abscissa $s \in [0, L]$ in a fixed reference frame $\mathcal{R}_{B_0}$. This neutral axis is discretized into N points distributed according to a non-constant discretization step $\Delta(s)$. Every variable attached to the continuum robot will hold the indicator $\bullet_0$. The studied continuum robot is considered under external loads, composed of bending moments $\tau_0(s)$ and forces $f_0(s)$, applied to the neutral axis. These loads are composed of tip loads, denoted as $\tau_0(L) \in \mathbb{R}^3$ and $f_0(L) \in \mathbb{R}^3$, and distributed loads $\tau_0(s_0) \in \mathbb{R}^3$ and $f_0(s_0) \in \mathbb{R}^3$ applied over the range $s \in [l_{min}\ ,\ l_{max}]$. Every external load is defined in the fixed reference frame $\mathcal{R}_{B_0}$. As commonly seen in the literature, the external loads will be grouped into a wrench:

$$w_0(s) = \begin{bmatrix} \tau_0(s) \\ f_0(s) \end{bmatrix} \tag{1}$$

A body frame $\mathcal{R}_0(s)$ is attached to the robot and is described by the homogeneous transformation matrix $T_0(s) \in$ SE(3) defining the transformation between frame $\mathcal{R}_{B_0}$ and $\mathcal{R}_0(s)$. Transformation $T_0(s)$ includes a rotation matrix $R_0(s) \in$ SO(3) and a translation vector $p_0(s) \in \mathbb{R}^3$. Let $\hat{\bullet}$ / $\bullet^\wedge$ denote the skew-symmetric operator [41] associated with SO(3) and its extension to SE(3), respectively. For $u = [u_x\ \ u_y\ \ u_z]^T$ and $v = [v_x\ \ v_y\ \ v_z]^T \in \mathbb{R}^3$:

$$\hat{u} = \begin{bmatrix} 0 & -u_z & u_y \\ u_z & 0 & -u_x \\ -u_y & u_x & 0 \end{bmatrix} \tag{2a}$$

$$\begin{bmatrix} v \\ u \end{bmatrix}^{\wedge} = \begin{bmatrix} 0 & -u_z & u_y & v_x \\ u_z & 0 & -u_x & v_y \\ -u_y & u_x & 0 & v_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{2b}$$

For further use, we also define $\bullet^{\vee}$ the inverse operator of $\bullet^{\wedge}$.

The evolution of $T_0(s)$ can be described by its linear rate of change $v_0(s) \in \mathbb{R}^3$ and the angular rate of change $u_0(s) \in \mathbb{R}^3$. By defining $\dot{\bullet}$ as the derivative $\frac{d}{ds}$, the evolution of $T_0(s)$ is given by the two differential equations:

$$\dot{R}_0(s) = R_0(s)\hat{u}_0(s) \tag{3a}$$

$$\dot{p}_0(s) = R_0(s)v_0(s) \tag{3b}$$

It is common to find the notation $\xi_0(s)$ to represent the change of rate of $T_0(s)$ as:

$$\dot{T}_0(s) = T_0(s)\hat{\xi}_0(s) \tag{4}$$

where

$$\xi_0(s) = \begin{bmatrix} v_0(s) \\ u_0(s) \end{bmatrix} \tag{5}$$

Since continuum robots are flexible structures, it is required to describe the correlation between internal forces, external loads, and $\mathcal{R}_0(s)$ rates of change. Let us define the robot's internal bending moment $m_0(s) \in \mathbb{R}^3$ and its internal force $n_0(s) \in \mathbb{R}^3$. Every internal moment or force is defined in the local body frame $\mathcal{R}_0(s)$. The application of the Cosserat rod theory to the robot neutral axis provides two differential equations resulting from the application of Newton's first law on a given infinitesimally small robot cross-section:

$$\dot{m}_0(s) = \hat{u}_0(s)m_0(s) - \hat{v}_0(s)n_0(s) - R_0(s)^T\dot{\tau}_0(s) \tag{6a}$$

$$\dot{n}_0(s) = \hat{u}_0(s)n_0(s) - R_0(s)^T\dot{f}_0(s) \tag{6b}$$

In addition to the Cosserat equations, some assumptions can be made for the studied continuum robot. By combining equations (3a), (3b), (6a), (6b) and the robot-specific assumptions (e.g. interactions between the different rods constituting the robot or actuation input), one can describe the robot behavior by a set of ODE in the form:

$$\dot{y}(s) = \mathcal{F}(y, s) \tag{7}$$

where $\mathcal{F}$ is a general function, and $y(s) \in \mathbb{R}^Y$ is a state vector (typically including curvatures and internal forces and moments). Computing the continuum robot model consists in integrating equation (7) along the robot.

If the initial state vector value $y(0)$ is fully known, one can integrate the ODE (7) directly. This first case is an Initial Value Problem (IVP). For continuum robots, in the general case, some components of $y(0)$, denoted $y_u(0) \subset y(0)$, are not known. On the other hand, additional constraints at the robot tip denoted $b \in \mathbb{R}^B$, may be known. This constitutes a Boundary Value Problem (BVP) where boundary conditions are split. Such problems can be solved using shooting methods [39] [40], in which values of $y_u(0) \in \mathbb{R}^U$ are optimized in an iterative process by integrating the corresponding IVP until b converges to 0 (see Fig. 1b.). The optimization process is typically stopped when $\|b\|_\infty \leq \epsilon$. The residual b is typically obtained by a static equilibrium at the robot tip, and is therefore composed of equilibrium of internal and external forces and moments at $s = L_i$ and $s = L$.

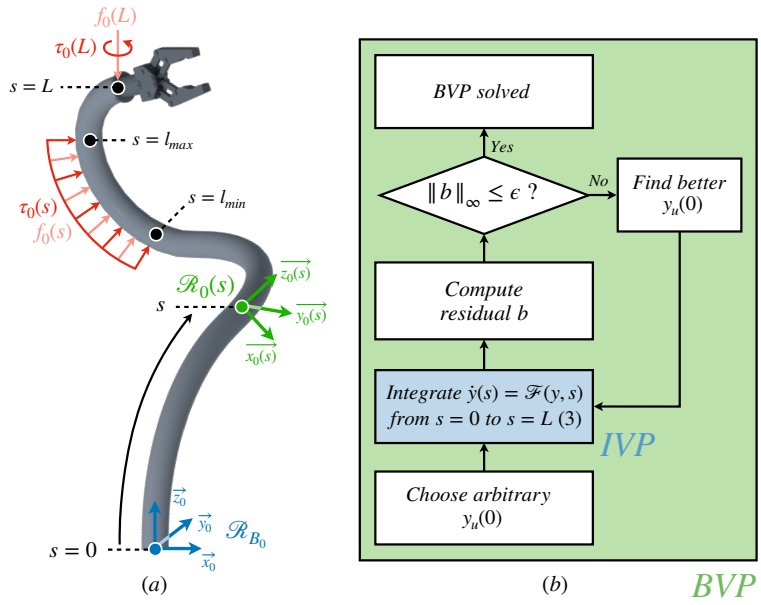


Figure 1: (a) Visualization of the base frame $\mathcal{R}_{B_0}$, the body frame $\mathcal{R}(s)$, the distributed loads $[\tau_0(s),\ f_0(s)]$ and the tips loads $[\tau_0(L),\ f_0(L)]$. (b) Principle of the shooting method used to solve the BVP.

2.2. *Rethinking the Compliance Matrix Definition*

To the best of the authors' knowledge, the Compliance matrix has always been defined as shape derivative w.r.t. tip loads for continuum robots [31, 30, 1]. Because application tasks are commonly defined at the robot's tip and knowing that the Compliance matrix is commonly used for task control loops, it was appropriate to define compliance as follows:

$$C(s) = \frac{\partial T_0^\vee}{\partial w_0\,(L)}(s) \in \mathbb{R}^{6\times 6} \tag{8}$$

The tip Compliance matrix $C(s)$ has interesting applications, for instance, to enable interaction control when the robot tip interacts with the environment. Nevertheless, its utility is limited in scenarios where interactions do not occur specifically at the robot tip, such as when seeking to control compliance with various contact points.

A new Compliance matrix definition is therefore proposed in this paper. This new Compliance matrix will be named Generalized Compliance matrix $C_{s_0}(s)$ in the following to prevent any confusion, and is defined as follows:

$$C_{s_0}(s) = \frac{\partial T_0^\vee}{\partial w_0\,(s_0)}(s) \in \mathbb{R}^{6\times 6} \tag{9}$$

Since $T_0(s)$ is defined all along the robot neutral axis, both definitions (8) and (9) are parameterized by s. $C_{s_0}(s)$ features a double arc-length discretization: for any couple of points $\{s, s_0\} \in \{[0, L] \times [0, L]\}$ along the robot, our definition includes the compliance associated with a wrench applied at s_0 and a displacement at s (where s and s_0 are not necessarily the same). The new Generalized Compliance matrix $C_{s_0}(s)$ (9) goes beyond the state of the art while maintaining the classical definition $C(s)$ as a special case of $C_{s_0}(s)$:

$$C(s) = C_{s_0=L}(s) \tag{10}$$

If one wants to compute the Generalized Compliance matrix $C_{s_0}(s)$ for any point of application s_0 of external forces, one needs an additional dimension to store all the new derivatives. A layout, illustrated by Fig 2a, is proposed to organize the $C_{s_0}(s)$ values properly. For future applications, $C_{s_0}(s)$ can be used to estimate changes of $T_0(s)$, denoted as $\Delta T_0(s)$ ($\Delta\bullet$ is used to indicate variations, here of shapes). An example is provided in Fig 2b.

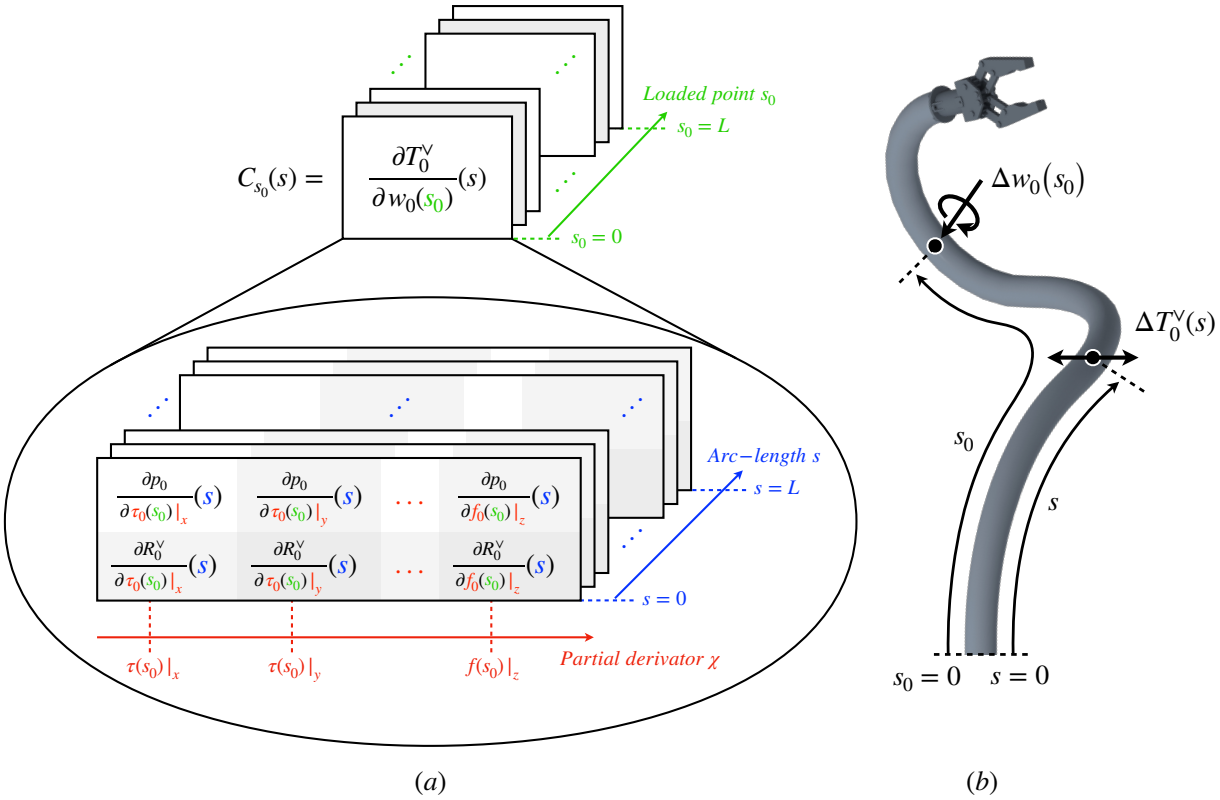


Figure 2: (a) The layout of the Generalized Compliance matrix $C_{s_0}(s)$ where each parameter refers to one matrix dimension. (b) Application using the Generalized Compliance matrix to estimate changes of homogeneous transformation matrix $\Delta T_0(s)$.

2.3. Adapting Existing Derivative Propagation Method for Generalized Compliance Matrix

Now that the Generalized Compliance matrix definition has been presented, the main challenge is to develop a method to compute it. Let us first start by describing the method developed by Rucker and. al. in [30] for the standard compliance matrix. The method starts by solving the BVP of a continuum robot for a given actuation input, yielding the value of $y_u(0)$ and the robot shape. Then, the method extends the state vector and proceeds to a single IVP integration to compute the Compliance, with the three new following ODEs. These differential equations, initially developed for Jacobians defined in the local frame $\mathcal{R}_0(s)$, have been adapted for definition in the global frame $\mathcal{R}_{B_0}$.

$$\dot{V}_\chi(s) = \frac{\partial \mathcal{F}}{\partial y(s)}(s)V_\chi(s) + \frac{\partial \mathcal{F}}{\partial T_0^\vee(s)}(s)E_\chi(s) + \frac{\partial \mathcal{F}}{\partial \chi}(s) \tag{11a}$$

$$\dot{E}_\chi(s) = \left(\hat{E}_\chi(s)\hat{\xi}(s)\right)^\vee + \left(T_0(s)\frac{\partial \hat{\xi}}{\partial \chi}(s)\right)^\vee \tag{11b}$$

$$\dot{\tilde{B}}_\chi(s) = \frac{\partial b}{\partial y(s)}V_\chi(s) + \frac{\partial b}{\partial T_0^\vee(s)}E_\chi(s) + \frac{\partial b}{\partial \chi} \tag{11c}$$

where χ is a dummy vector. In [30], it is defined as $\chi \in \{w_0(L),\ y_u(0)\}$ for computing the Compliance matrix. Additionally, we have:

$$V_\chi(s) = \frac{\partial y}{\partial \chi}(s) \in \mathbb{R}^{Y\times X} \tag{12a}$$

$$E_\chi(s) = \frac{\partial T_0^\vee}{\partial \chi}(s)\Big|_{b\neq 0} \in \mathbb{R}^{6\times X} \tag{12b}$$

$$B_\chi = \tilde{B}_\chi(s=L) = \frac{\partial b}{\partial \chi} \in \mathbb{R}^{B\times X} \tag{12c}$$

where X refers to the size of χ and $\bullet|_{b\neq 0}$ specifies that E_χ derivative does not guarantee the distal constraint $b = 0$. On the contrary, the Compliance matrix $C_L(s)$ is defined for the case where the constraint $b = 0$ is realized. But in this case, $\bullet|_{b=0}$ is omitted for clarity reasons. Computing the Compliance matrix requires the B_χ value which is computed thought the differential equation on $\tilde{B}_\chi$ (11c) in the method from [30]. Computing the tip Compliance matrix of the robot requires integrating equations (11a)-(11c) in order to combine them as follows:

$$C_L(s) = E_{w_0(L)}(s) - E_{y_u(0)}(s)B^\dagger_{y_u(0)}B_{w_0(L)}. \tag{13}$$

where $\bullet^\dagger$ refers to the Moore-Penrose pseudo-inverse operator.

In (13), which is the direct application of the method from [30], the Compliance matrix is computed only with respect to forces applied at the tip. Extending the method to the Generalized Compliance matrix is not straightforward. The computation of $B_{w_0(L)}$ could be done directly in [30] because the residual b contains the forces applied at the tip of the robot. Applying the method with a force applied at an arc length $s_0 \neq L$ would therefore require solving a BVP with 3-point split boundary conditions. Instead, we propose to redefine the problem as follows.

First, we define the dummy variable χ as :

$$\chi \in \{w_0(s_0),\ y_u(0)\} \tag{14}$$

To avoid the above-mentioned problem, when forces are not applied at the tip we compute partial derivatives with respect to $w_0(s_0) = [\tau_0(s), f_0(s)]$ with a point force/torque placed at arc length s_0. Such partial derivatives can be computed with literal expressions, because $\tau_0(s)$ and $f_0(s)$ appear explicitly in equations (6a) and (6b), which are used for constructing $\mathcal{F}$. This operation enables computing equations (12a) - (12c) with respect to the full vector χ defined in (14). The Generalized Compliance matrix can then be computed for each value of s_0 as :

$$C_{s_0}(s) = E_{w_0(s_0)}(s) - E_{y_u(0)}(s)B^\dagger_{y_u(0)}B_{w_0(s_0)}. \tag{15}$$

Note that in [30] the same derivative propagation method is applied for computing the robot Jacobian matrix. In our case, since we are mainly interested in the Generalized Compliance matrix, we have left the robot Jacobian-related part out of the equations. Our proposed method is however easily applicable to computing the robot Jacobian matrix. To do so, one needs to add the vector of robot joint variables $q \in \mathcal{R}^Q$ to the vector χ, leading to the computation of additional partial derivatives E_q, V_q, and B_q. Such terms are then assembled to compute $J(s)$ in a similar way as for $C_{s_0}(s)$:

$$J(s) = E_q(s) - E_{y_u(0)}(s)B^\dagger_{y_u(0)}B_q. \tag{16}$$

2.4. Towards the Development of a Dedicated Low-Level DPM

Integrating (11a) - (11c), as in [30], and computing (15) is carried out using literal equations, calculated by computing the associated partial derivatives of equation (7). As shown in [30], this is a computationally efficient method which ensures fast computational time. Applying the adapted method for computing the Generalized Compliance matrix, however, reveals several computational bottlenecks.

First, integrating (11a) - (11c) turns out to be unnecessary since $y(s)$ can be integrated thanks to a unique ODE, given by equation (7). It is possible to compute the partial derivatives of (7) directly as it has been done for a passive

Cosserat rod [31]. Computing the partial derivatives directly on (7) will intrinsically propagate the χ partial derivative all along the robot thanks to the s derivatives. This approach leads to the following second-degree ODE:

$$\frac{\partial}{\partial \chi}\left(\frac{dy}{ds}\right)(s) = \frac{\partial \mathcal{F}}{\partial \chi}(y, s) \in \mathbb{R}^{Y \times X} \tag{17}$$

Because the neutral axis discretization is independent of the external loads and joint values, partial derivatives can be reorganized as follows:

$$\frac{\partial}{\partial \chi}\left(\frac{dy}{ds}\right)(s) = \frac{d}{ds}\left(\frac{\partial y}{\partial \chi}\right)(s) \tag{18}$$

Combining equations (17) and (18) yields a new ODE which can be integrated along the arc length of the continuum robot. Once this is done, the values of $V_\chi(s)$, $E_\chi(s)$, and B_χ can be assembled in one pass without requiring to integrate them. Because the derivatives are applied directly to the state vector components, this method will be designated as a Low-Level Derivative Propagation Method (LLDPM).

Unlike a direct adaptation of [30], our proposed LLDPM minimizes the number of equations required to propagate the derivatives. Each of the quantities computed in (17) is needed behind the hood to evaluate high-level ODEs (11a) - (11b). A striking example is $\frac{\partial \xi}{\partial \chi}(s)$ which appears explicitly in (11b) and is also computed in (17) because ξ is part of the state vector y. While in our method only such low-level ODEs are integrated, the original DPM requires assembling them in the form of large matrices to be integrated. This introduces supplementary operations which turn out not to be necessary.

A second advantage of our proposed LLDPM is that it enables a smart adaptation of the integration bounds. [30] presents an integration process considering all the derivatives along the robot $s \in [0, L]$. We can observe, however, that the low-level partial ODE referring to $w_0(s_0)$ does not generate information before reaching the point where the load is applied, i.e. the integration is not relevant on the integration range $s \in [0, s_0]$. This means that partial derivatives related to $w_0(s_0)$ can be directly set to zero for a given integration point if the considered arc length s satisfies $s < s_0$. For example, at $s = 0$ the only derivative computed will be w.r.t. $w_0(s_0 = 0)$ while on the other boundary $s = L$ every external load $w_0(s_0)$, $s_0 \in [0,\ L]$ derivatives will be computed. Such a selection of the integration bounds is made possible by the fact that low-level partial derivatives are individually integrated in our LLDPM method. In the adapted DPM for Generalized Compliance matrix, such a selection of the integration bounds is not possible because low-level variables are concatenated in large high-level matrices V_χ and E_χ.

Let us define $N_{DPM(C_{s_0})}$ and $N_{LLDPM(C_{s_0})}$ the number of operations required to compute the Generalized Compliance matrix using respectively the DPM and the LLDPM. By multiplying the size of the matrices involved in the computation by the number of integration points to consider, the total number of operations required is estimated by equations (19a)-(19b).

$$N_{DPM(C_{s_0})} = N^2(6Y + 6B + 36) \tag{19a}$$

$$N_{LLDPM(C_{s_0})} = N^2 3Y + N3Y \tag{19b}$$

where it is recalled that N is the number of discretization points, Y is the size of $y(s)$, U is the size of $y_u(0)$ and B is the size of b. Using the LLDPM saves a large number of operations as the number of discretization points increases. In fact, subtracting (19b) from (19a) shows that the number of operations saved by our proposed method increases quadratically as the number of discretization points N increases:

$$N_{Gain(C_{s_0})} = N_{DPM(C_{s_0})} - N_{LLDPM(C_{s_0})} = N^2(3Y + 6B + 36) - N3Y \tag{20}$$

Table 1 shows the LLDPM theoretical computational cost when applying equation (20) to several models of continuum robots from the literature. This table demonstrates the benefits of the LLDPM for reducing the number of operations when computing the Generalized Compliance matrix $C_{s_0}(s)$.

Paper	Continuum Robot	N	Y	U	B	$\frac{N_{Gain(C_{s_0})}}{N_{DPM(C_{s_0})}}$
[30]	CTCR with 2 tubes	60	22	6	5	66%
[30]	CTCR with 3 tubes	60	24	9	6	66%
[10]	TACR with 6 tendons and 3 segments	30	18	6	6	69%
[9]	TACR with 3 tendons and 1 segment	40	27	6	6	65%
[38]	CPPR with 2 tubes *	100	40	10	4	60%
[38]	CPPR with 3 tubes *	100	53	12	8	60%
[12]	MBCR with 8 rods and 2 segments *	100	26	10	10	69%
[12]	MBCR with 12 rods and 3 segments *	100	30	12	12	68%

Table 1: Computation of equation (20) for representative continuum robot designs and models from the literature. When the number of discretization points was not specified in the paper, the default value $N = 100$ was used. Such cases are indicated with a $\bullet^*$.

Values in Table 1 indicate that as well as the Generalized Compliance matrix definition, it was essential to develop the dedicated LLDPM to compute it. Indeed, the LLDPM cuts the number of operations by half, compared to a simpler adaptation of existing DPM. Current developments in continuum robots push ahead with more complex robot architectures and combinations of different technologies. The most advanced robot models rely on the Cosserat equations, resulting in the introduction of more complex variables, and a higher number of operations. As a consequence, the more relevant the proposed LLDPM will be.

In the next section, the proposed LLDPM will be applied to a CTCR model. For this kind of robot, several tubes are involved, each of them modeled as a Cosserat rod. This assumption makes the $y(s)$ components have different integration ranges that do not always correspond to the full $s \in [0,\ L]$. Therefore some optimization can be added to the integration process by considering only the relevant $y(s)$ components depending on the current evaluation point s.

3. Computing Generalized Compliance Matrix on Concentric Tubes Continuum Robots

3.1. From the assumptions to the definition of state vector y

CTCR are composed of several elastic precurved concentric tubes creating an overall deformable and controllable shape, designated as the "robot". This continuum robot architecture is based on the elastic interactions of the tubes, which are commonly made in Nitinol. Each tube is actuated in translation and rotation at its base, allowing the deployed part miniaturization. Numerous works have already been presented regarding aspects of the CTCR such as design, fabrication, model, sensing, and control [42, 43]. Because the application of LLPDM is directly dependent on the quasi-static model (7), the assumptions used in this study will be redefined here. However, the model provided here is very close to the ones presented in [7] and [8].

Let us define $nbT \in \mathbb{N}^+$ the total number of tubes that form the CTCR and $nbT(s) \in \mathbb{N}^+$ the number of tubes involved at a given s. Each tube is allocated an index denoted by $i \in [\![1, nbT]\!]$, which defines its position in the concentric assembly: $i = 1$ is the shortest one while $i = nbT$ refers to the longest (and thinnest) one. Index $i = 0$ refers to variables associated with the assembled robot. In addition to the Bishop frame $\mathcal{R}_0(s)$, each tube i has its body frame $\mathcal{R}_i(s)$ that is used to measure the twist $\theta_i(s)$ defined as:

$$\theta_i(s) = \langle \overrightarrow{x_0}(s), \overrightarrow{x_i}(s) \rangle \tag{21}$$

Each tube is expected to be straight for length $L_{ri} \in \mathbb{R}$ then precurved with a constant radius of curvature $R_{ci} \in \mathbb{R}$ up to the distal end, which corresponds to length $L_{ci} \in \mathbb{R}$. The total length of the tube $L_i \in \mathbb{R}$ corresponds to the sum of the straight and curved sections:

$$L_i = L_{ri} + L_{ci} \tag{22}$$

The natural 3D curvature of the tube is designated by $u_i^*(s) \in \mathbb{R}^3$ while $u_i(s) \in \mathbb{R}^3$ represents the curvature after tube deformation. The undeployed part of the tube is constrained to be straight by the actuation unit, even for precurved

sections. Each tube is actuated by two motors: it is rotated by an angle denoted $\theta_{ci} \in \mathbb{R}$ and translated by a curvilinear length denoted $\beta_{ci} \in \mathbb{R}$.

When the tubes are concentrically assembled, they should share the robot shape $p_0(s)$ and curvature $u_0(s)$. If we assume that there is no mechanical play between the tubes, each tube should exhibit the same curvature at any curvilinear abscissa where they are deployed. Furthermore, the driving control has to guarantee that the deployment respects the following conditions:

$$0 \leq \beta_{c1} \leq ... \leq \beta_{ci} \leq ... \leq L = \beta_{cnbT} \tag{23}$$

Finally, tube cross-sections are not expected to undergo plastic deformations: they keep a perfectly circular section, and their neutral line stay centered on their section.

It is necessary to define a constitutive law (also known as a stress-strain law) of the tubes to estimate mechanical behavior resulting from tube interactions. It is assumed that Nitinol keeps an elastic and linear behavior, i.e. the angular stress is proportional to the internal bending moment. It is also expected that the shear and elongation of the tubes are negligible compared to bending and twisting (also known as the Kirchhoff assumption). Finally, the manufacturing process is assumed to be sufficiently controlled to suggest that the Nitinol material used is homogeneous and isotropic along the tubes. This leads to consider a diagonal and constant stiffness matrix $K_i \in \mathbb{R}^{3\times3}$ defined as follows (24):

$$K_i = \begin{bmatrix} kx_i & 0 & 0 \\ 0 & kx_i & 0 \\ 0 & 0 & kz_i \end{bmatrix} \tag{24}$$

All the information contained in the defined assumptions can be translated into equations. With the body frame definitions and Cosserat equations (6a), (6b), (3a) and (3b), it is then possible to define a state vector $y(s)$ as follows:

$$y(s) = \begin{bmatrix} \theta_1(s) \\ ... \\ \theta_{nbT}(s) \\ u_1(s)|_z \\ ... \\ u_{nbT}(s)|_z \\ m_0(s) \\ n_0(s) \\ R_0(s)^\vee \\ p_0(s) \end{bmatrix} \in \mathbb{R}^{2nbT+12} \tag{25}$$

that can be described by the following ODE, which is the developed form of ODE (7):

$$\dot{\theta}_i(s) = u_i|_z(s) \tag{26a}$$

$$\dot{u}_i|_z(s) = \frac{kx_i}{kz_i} u_i(s)\Big|_{x,y}^T \begin{bmatrix} u_i^*(s)|_y \\ -u_i^*(s)|_x \end{bmatrix} \tag{26b}$$

$$\dot{m}_0(s) = \hat{u}_0(s) m_0(s) - \hat{e}_z n_0(s) - R_0(s)^T \dot{\tau}_0(s) \tag{26c}$$

$$\dot{n}_0(s) = \hat{u}_0(s) n_0(s) - R_0(s)^T \dot{f}_0(s) \tag{26d}$$

$$\dot{R}_0(s) = R_0(s)\ \hat{u}_0(s) \tag{26e}$$

$$\dot{p}_0(s) = R_0(s)\ e_z \tag{26f}$$

where

$$u_0(s)\big|_{x,y} = \left(\left(\sum_{i\in nbT_0(s)} K_i\right)^{-1}\left(\sum_{i\in nbT_0(s)} Rot_z(\theta_i(s))K_i u_i^*(s)\right) + m_0(s)\right)\Bigg|_{x,y} \tag{27a}$$

$$u_0(s)\big|_z = 0 \tag{27b}$$

$$u_i(s)\big|_{x,y} = \left(Rot_z(\theta_{iT}(s))^T\, u_0(s)\right)\Big|_{x,y} \tag{27c}$$

$$u_i^*(s) = Rot_z\left(\theta_{ci}\right)\begin{bmatrix}\phi_i(s)\\ 0\\ 0\end{bmatrix} \tag{27d}$$

$$\phi_i(s) = \begin{cases} \frac{1}{R_{ci}} & if\ max\left(0,\ \beta_{ci} - L_{ci}\right) \le s \le \beta_{c_i} \\ 0 & if\ not \end{cases} \tag{27e}$$

$$e_z = [0\ 0\ 1]^T \tag{27f}$$

Vector e_z is the z-axis unit vector, $\bullet^T$ refers to the standard transpose operator, $Rot_z(\ )$ operator is the 3D rotation matrix around the z-axis and $\phi_i(s)$ describes the tubes piecewise constant curvature. The indicator $|_{x/y/z}$ is used to specify particular vector components when it is required.

Because the robot body frame $\mathcal{R}_0(s)$ is defined as a Bishop frame, the z-component of the robot curvature $u_0(s)|_z$ stays null all along the robot (27b). However because each tube can twist, the z-component of individual curvature $u_i(s)|_z$ has to be estimated by equation (26b). It can be noticed that equations (26a) and (26b) are defined for each tube and not for the robot $i=0$. This means that the ranges on which they will be integrated have to be adapted depending on the tubes deployment. The integration range will be maximum for tube $i=nbT$ and minimum for tube $i=1$.

3.2. Definition of the Distal Residual b

Choosing a direction for the integration defines the known $y(0)$ components. The known boundary conditions are shown in Fig 3. Therefore, choosing to integrate ODE (7) along s^+ creates a sub-state vector $y_u(0)$, defined by:

$$y_u(0) = \begin{bmatrix} u_1(0_1)|_z \\ \dots \\ u_{nbT}(0_{nbT})|_z \\ m_0(0) \\ n_0(0) \end{bmatrix} \tag{28}$$

that groups the unknown values of $y(0)$. All the initial values of $y(0)$ are provided in Table 2.

Derivatives	**Abscissa** s	**Dimensions**	**Value**
Global rotation matrix $R_0(s)$	$s = 0_0$	$\mathbb{R}^{3\times3}$	I_3
Global position vector $p(s)$	$s = 0_0$	$\mathbb{R}^3$	$(\beta_{c3} - L_3)\, e_z$
Twist of tubes $\theta_i(s)$	$s = 0_i$	$\mathbb{R}$	0
Twist curvature of tubes $u_i(s)\|_z$	$s = 0_i$	$\mathbb{R}$	unknown
Global internal bending moment $m_0(s)$	$s = 0$	$\mathbb{R}^3$	unknown
Global internal force $n_0(s)$	$s = 0$	$\mathbb{R}^3$	unknown

Table 2: Values of the initial state vector $y(0)$ components; $s=0_0$ refers to the first point of the robot, $s=0_i$ refers to the first point of the tube i and $s=0$ refers to the first point of the deployed robot.

As explained in Section 2.1 the shooting method used to solve the CTCR quasi-static model is based on the optimization of a residual b. This residual is built on the known distal values of $y(L)$ and compares the integration values

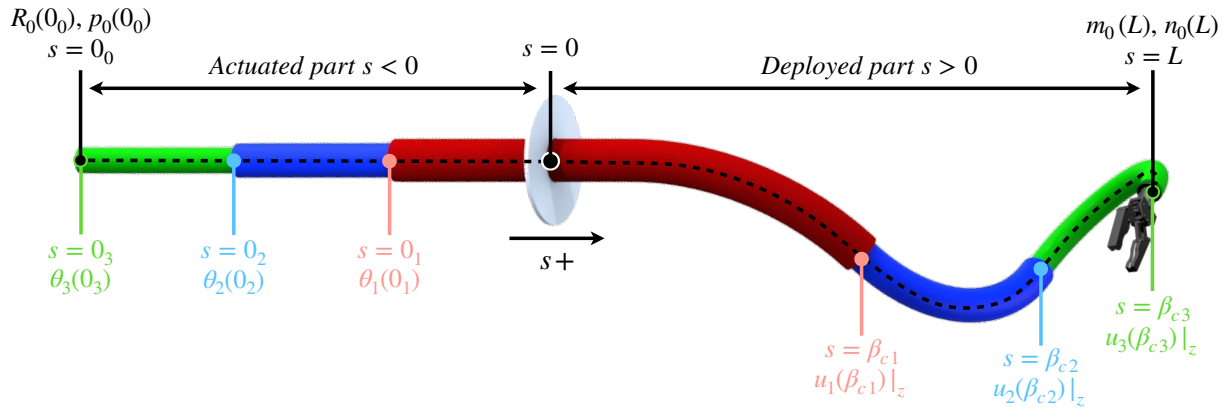


Figure 3: Illustration of the known values of the state vector (25) for a CTCR composed of 3 tubes.

and the ones expected by the distal condition. Since tube $i \in [\![1, nbT-1]\!]$ is concentrically assembled to the others, the z-component of its distal curvature $u_i(L_i)|_z$ is mechanically not constrained. Assuming that tubes do not have a precurved twist, i.e. $u_i^*(s)|_z = 0$, $u_i(L_i)|_z$ has to be null to guarantee the tube twist equilibrium. The distal end of the robot is also mechanically free and has to be balanced. However, some tip loads can be applied to the tip of the robot, which means that the internal forces should be equal to the tip loads. The same consideration can be made for the longest tube $i = nbT$ creating a specific constraint on $u_{nbT}(L)|_z$. These three mechanical constraints can be used to formulate the residual b as follows:

$$b = \begin{bmatrix} u_1(\beta_{c1})\big|_z \\ \dots \\ u_i(\beta_{ci})\big|_z \\ \dots \\ u_{nbT-1}(\beta_{cnbT-1})\big|_z \\ u_{nbT}(L)\big|_z - \frac{[0\,0\,1]}{kz_{nbT}} R_{nbT}(L)^T \tau_0(L) \\ m_0(L) - R_0(L)^T \tau_0(L) \\ n_0(L) - R_0(L)^T f_0(L) \end{bmatrix} \in \mathbb{R}^{nbT+6} \tag{29}$$

where

$$R_{nbT}(s) = R_0(s) Rot_z(\theta_{nbT}(s))^T \tag{30}$$

3.3. Low-Level Derivatives for the LLDPM

Computation of (17) requires to compute the partial derivatives of equations (26a) to (26f) w.r.t. χ. The general derivatives expressions are provided in the following equations:

$$\frac{d}{ds}\left(\frac{\partial \theta_i}{\partial \chi}\right)(s) = \frac{\partial u_i|_z}{\partial \chi}(s) \tag{31a}$$

$$\begin{aligned}\frac{d}{ds}\left(\frac{\partial u_i|_z}{\partial \chi}\right)(s) = \frac{kx_i}{kz_i}\Bigg(& \frac{\partial u_0}{\partial \chi}(s)^T Rot_z(\theta_i(s)) \begin{bmatrix} u_i^*(s)|_y \\ -u_i^*(s)|_x \\ 0 \end{bmatrix} \\ & + u_0(s)^T \hat{e}_z Rot_z(\theta_i(s)) \frac{\partial \theta_i}{\partial \chi}(s) \begin{bmatrix} u_i^*(s)|_y \\ -u_i^*(s)|_x \\ 0 \end{bmatrix} \\ & + u_0(s)^T Rot_z(\theta_i(s)) \begin{bmatrix} \frac{\partial u_i^*|_y}{\partial \chi}(s) \\ -\frac{\partial u_i^*|_x}{\partial \chi}(s) \\ 0 \end{bmatrix} \Bigg)\end{aligned} \tag{31b}$$

$$\frac{d}{ds}\left(\frac{\partial m_0}{\partial \chi}\right)(s) = \frac{\partial \hat{u}_0}{\partial \chi}(s) m_0(s) + \hat{u}_0(s)\frac{\partial m_0}{\partial \chi}(s) - \hat{e}_z \frac{\partial n_0}{\partial \chi}(s) - \frac{\partial R_0}{\partial \chi}(s)^T \dot{\tau}_0(s) - R_0(s)^T \frac{\partial \dot{\tau}_0}{\partial \chi}(s) \tag{31c}$$

$$\frac{d}{ds}\left(\frac{\partial n_0}{\partial \chi}\right)(s) = \frac{\partial \hat{u}_0}{\partial \chi}(s) n_0(s) + \hat{u}_0(s)\frac{\partial n_0}{\partial \chi}(s) - \frac{\partial R_0}{\partial \chi}(s)^T \dot{f}_0(s) - R_0(s)^T \frac{\partial \dot{f}_0}{\partial \chi}(s) \tag{31d}$$

$$\frac{d}{ds}\left(\frac{\partial R_0}{\partial \chi}\right)(s) = \frac{\partial R_0}{\partial \chi}(s)\hat{u}_0(s) + R_0(s)\frac{\partial \hat{u}_0}{\partial \chi}(s) \tag{31e}$$

$$\frac{d}{ds}\left(\frac{\partial p_0}{\partial \chi}\right)(s) = \frac{\partial R_0}{\partial \chi}(s) e_z \tag{31f}$$

where

$$\begin{aligned}\frac{\partial u_0}{\partial \chi}(s)\Big|_{x,y} = & \left(\frac{\partial}{\partial \chi}\left(\sum_{i \in nbT_0(s)} K_i\right)^{-1} \sum_{i \in nbT_0(s)} \left(Rot_z(\theta_i(s)) K_i u_i^*(s) + m_0(s) \right)\right. \\ & + \left(\sum_{i \in nbT_0(s)} K_i\right)^{-1} \sum_{i \in nbT_0(s)} \Bigg(\hat{e}_z\, Rot_z(\theta_i(s)) \frac{\partial \theta_i}{\partial \chi}(s) K_i u_i^*(s) \\ & + Rot_z(\theta_i(s)) K_i \frac{\partial u_i^*}{\partial \chi}(s) \\ & \left. + \frac{\partial m_0}{\partial \chi}(s) \Bigg) \right)\Bigg|_{x,y}\end{aligned} \tag{32a}$$

$$\frac{\partial u_i}{\partial \chi}(s)\Big|_{x,y} = \left(\left(\hat{e}_z Rot_z(\theta_i(s)) \frac{\partial \theta_i}{\partial \chi}(s)\right)^T u_0(s) + Rot_z(\theta_i(s)) \frac{\partial u_0}{\partial \chi}(s)\right)\Big|_{x,y} \tag{32b}$$

$$\frac{\partial u_i^*}{\partial \chi}(s) = \hat{e}_z Rot_z(\theta_{ci}) \frac{\partial \theta_{ci}}{\partial \chi} \begin{bmatrix} \phi_i(s) \\ 0 \\ 0 \end{bmatrix} + Rot_z(\theta_{ci}) \begin{bmatrix} \frac{\partial \phi_i}{\partial \chi}(s) \\ 0 \\ 0 \end{bmatrix} \tag{32c}$$

$$\frac{\partial \phi_i}{\partial \chi}(s) = \begin{cases} \pm\infty & if\ \chi = \beta_{ci}\ and\ s \in \{max(0,\ \beta_{ci} - L_{ci}),\ \beta_{ci}\} \\ 0 & if\ not \end{cases} \tag{32d}$$

Particular attention must be paid to the equation (32c) which contains the partial derivative $\frac{\partial \phi_i}{\partial \chi}(s)$. Because of the piecewise constant curvature assumption, $\phi_i(s)$ shows a discontinuity on $s \in \{\max(0,\ \beta_{ci} - L_{ci}),\beta_{ci}\}$ (see definition

(27e)). Since $d\beta_{ci} = -ds$, equations (32a)-(32d) show undefined values which can be quite challenging to handle with an analytic approach. Using the neutral axis discretization step $\Delta(s)$, it is possible to estimate these values using a local mean value approximation near the problematic abscissas $s \in \{max\,(0,\ \beta_{ci} - L_{ci})\,,\ \beta_{c_i}\}$. It can also be noticed that the equations (31d) and (31c) include derivatives $\frac{\partial \dot{\tau}_0}{\partial \chi}(s)$ and $\frac{\partial \dot{f}_0}{\partial \chi}(s)$. Here $\dot{\tau}_0(s)$ and $\dot{f}_0(s)$ refer to the distribution of the external loads and it is assumed that there is a single punctual load applied for each discretized piece $[s,\ s + \Delta(s)]$ of the robot. Therefore, the distribution of loads can be expressed as follows:

$$\dot{\tau}_0(s) = \frac{\tau_0(s)}{\Delta(s)} \tag{33a}$$

$$\dot{f}_0(s) = \frac{f_0(s)}{\Delta(s)} \tag{33b}$$

All the initial values and particular derivatives required to integrate equations (31a)-(31f) are summarized in Table 3.

Initial values			
Partial derivatives	**Abscissa** s	**Dimensions**	**Value**
$\frac{\partial u_i(s)\|_z}{\partial u_j(0_j)\|_z}$	$s = 0_i$	$\mathbb{R}$	δ_{ij}
$\frac{\partial \theta_{ci}}{\partial \theta_{cj}}$	$s = 0_i$	$\mathbb{R}$	δ_{ij}
$\frac{\partial m_0}{\partial m_0(0)}(s)$	$s = 0$	$\mathbb{R}^{3\times3}$	I_3
$\frac{\partial n_0}{\partial n_0(0)}(s)$	$s = 0$	$\mathbb{R}^{3\times3}$	I_3
Particular values			
Partial derivatives	**Abscissa** s	**Dimensions**	**Value**
$\frac{\partial \phi_i}{\partial \beta_{cj}}(s)$	$s = max\,(0,\ \beta_{ci} - L_{ci})$	$\mathbb{R}$	$-\delta_{ij}\frac{R_{ci}}{\Delta(s)}$
$\frac{\partial u_i}{\partial \beta_{cj}}(s)\|_{x,y}$	$s = max\,(0,\ \beta_{ci} - L_{ci})$	$\mathbb{R}^2$	$-\delta_{ij}\frac{u_i(s+\Delta(s))-u_i(s)}{\Delta(s)}\|_{x,y}$
$\frac{\partial u_i^*}{\partial \beta_{cj}}(s)$	$s = max\,(0,\ \beta_{ci} - L_{ci})$	$\mathbb{R}^3$	$-\delta_{ij}\frac{u_i^*(s+\Delta(s))-u_i^*(s)}{\Delta(s)}$
$\frac{\partial u_0}{\partial \beta_{cj}}(s)\|_{x,y}$	$s = max\left(0,\ \beta_{cj} - L_{cj}\right)$	$\mathbb{R}^2$	$-\frac{u_0(s+\Delta(s))-u_0(s)}{\Delta(s)}\|_{x,y}$
$\frac{\partial \phi_i}{\partial \beta_{cj}}(s)$	$s = \beta_{c_i}$	$\mathbb{R}$	$\delta_{ij}\frac{R_{ci}}{\Delta(s)}$
$\frac{\partial u_i}{\partial \beta_{cj}}(s)\|_{x,y}$	$s = \beta_{c_i}$	$\mathbb{R}^2$	$\delta_{ij}\frac{u_i(s+\Delta(s))-u_i(s)}{\Delta(s)}\|_{x,y}$
$\frac{\partial u_i^*}{\partial \beta_{cj}}(s)$	$s = \beta_{c_i}$	$\mathbb{R}^3$	$\delta_{ij}\frac{u_i^*(s+\Delta(s))-u_i^*(s)}{\Delta(s)}$
$\frac{\partial u_0}{\partial \beta_{cj}}(s)\|_{x,y}$	$s = \beta_{c_j}$	$\mathbb{R}^2$	$\frac{u_0(s+\Delta(s))-u_0(s)}{\Delta(s)}\|_{x,y}$
$\frac{\partial \tau_0}{\partial \tau_0(s_0)}(s)$	$s = s_0$	$\mathbb{R}^{3\times3}$	I_3
$\frac{\partial f_0}{\partial f_0(s_0)}(s)$	$s = s_0$	$\mathbb{R}^{3\times3}$	I_3
$\frac{\partial \dot{\tau}_0}{\partial \tau_0(s_0)}(s)$	$s = s_0$	$\mathbb{R}^{3\times3}$	$\frac{1}{\Delta(s)}I_3$
$\frac{\partial \dot{f}_0}{\partial f_0(s_0)}(s)$	$s = s_0$	$\mathbb{R}^{3\times3}$	$\frac{1}{\Delta(s)}I_3$

Table 3: Initial and particular derivatives values required to integrate equations (31a)-(31f).

where δ_{ij} refers to the Kronecker delta, $s = 0_i$ refers to the first point of tube i, $s = 0$ refers to the first point of the

deployed robot, $s = max(0,\ \beta_{ci} - L_{ci})$ refers to the first point of tube i precurved part, $s = \beta_{ci}$ refers to the last point of tube i and $s = s_0$ refers to the loaded point abscissa of the studied external loads.

After integrating ODE (31e) and (31f), $E_\chi(s)$ value can be computed at once from its definition:

$$E_\chi(s) = \begin{bmatrix} \frac{\partial R_0}{\partial \chi}(s) & \frac{\partial p_0}{\partial \chi}(s) \\ 0 & 1 \end{bmatrix}^\vee \tag{34}$$

B_χ is composed of the state vector partial derivatives computed from ODE (31a)-(31f). The following B_χ expression comes out deriving b expression (29):

$$B_\chi = \frac{\partial b}{\partial \chi} = \begin{bmatrix} \frac{\partial u_1}{\partial \chi}(\beta_{c1})|_z \\ \dots \\ \frac{\partial u_i}{\partial \chi}(\beta_{ci})|_z \\ \dots \\ \frac{\partial u_{nbT-1}}{\partial \chi}(\beta_{cnbT-1})|_z \\ \frac{\partial u_{nbT}}{\partial \chi}(L) - \frac{(0\ 0\ 1]}{kz_{nbT}} \left[\frac{\partial R_{nbT}}{\partial \chi}(L)^T \tau_0(L) + R_{nbT}(L)^T \frac{\partial \tau_0}{\partial \chi}(L) \right) \\ \frac{\partial m_0}{\partial \chi}(L) - \left(\frac{\partial R_0}{\partial \chi}(s)^T \tau_0(L) + R_0(L)^T \frac{\partial \tau_0}{\partial \chi}(L) \right) \\ \frac{\partial n_0}{\partial \chi}(L) - \left(\frac{\partial R_0}{\partial \chi}(s)^T f_0(L) + R_0(L)^T \frac{\partial f_0}{\partial \chi}(L) \right) \end{bmatrix} \tag{35}$$

where

$$\frac{\partial R_{nbT}}{\partial \chi}(s) = \frac{\partial R_0}{\partial \chi}(s) Rot_z(\theta_{nbT}(s))^T + R_0(s) \left(\hat{e}_z Rot_z(\theta_{nbT}(s)) \frac{\partial \theta_{nbT}}{\partial \chi}(s) \right)^T \tag{36}$$

Finally, the Generalized Compliance matrix $C_{s_0}(s)$ can be computed from equation (15). A summarizing scheme for computing $C_{s_0}(s)$ using the LLDPM is provided in Fig 4. This scheme can also be applied to compute $J(s)$ by using the partial derivatives associated to $\chi \in \{\theta_{ci}\ ,\ \beta_{ci}\}$.

4. Large-scale numerical validation of the Generalized Compliance Matrix Model

In this section we present a large-scale numerical validation of the Generalized Compliance matrix and robot Jacobian matrices as computed using our LLDPM method.

4.1. Overall validation procedure and metrics

Let us define Ω_{robot} as a set of parameters defining the robot features of a given CTCR, and $\Omega_{control}$ which includes all parameters that can change during a control loop (control inputs but also external forces):

$$\Omega_{robot} = \{L_{ri}\ ,\ L_{ci}\ ,\ R_{ci}\ ,\ K_i\} \tag{37a}$$

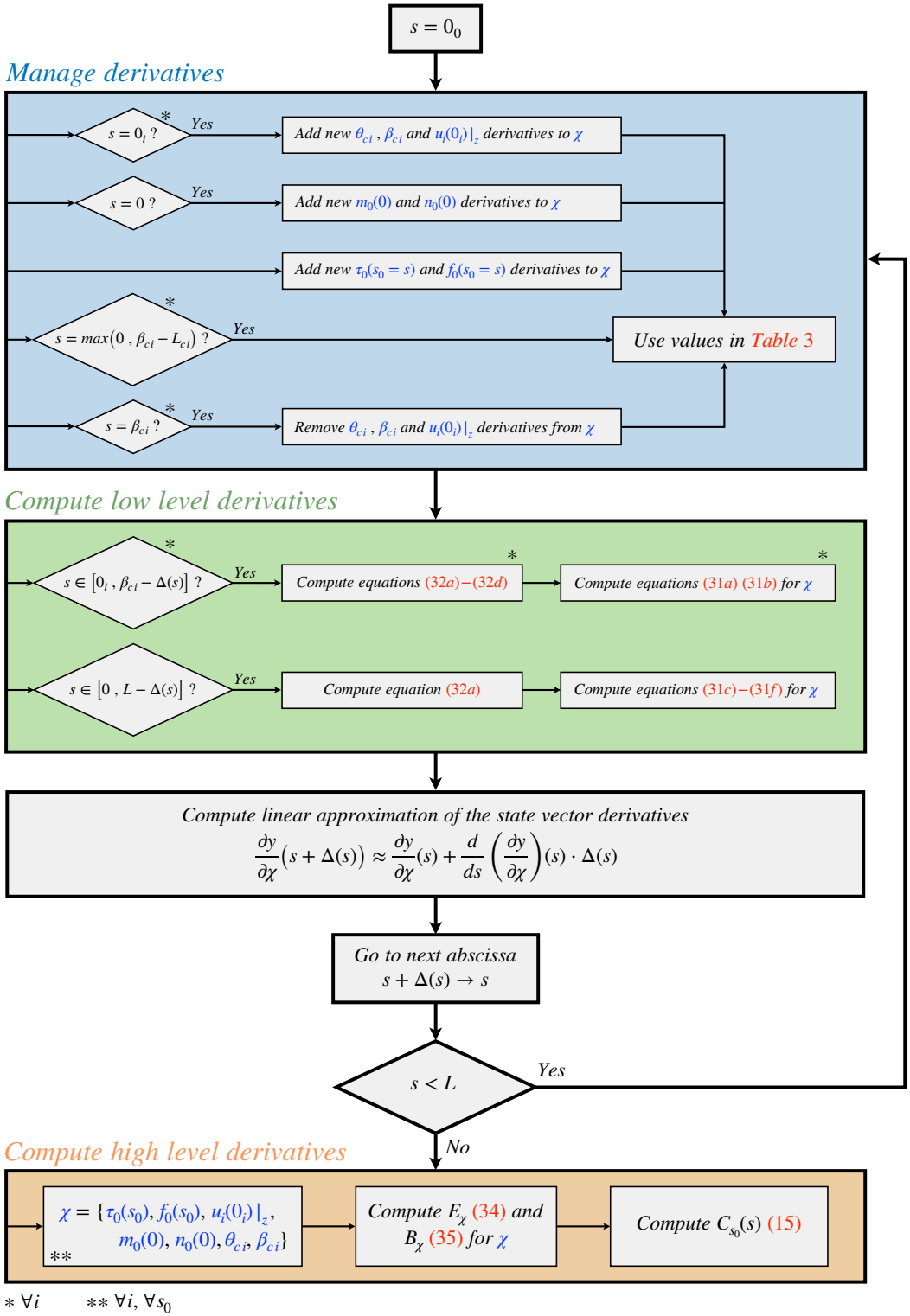
$$\Omega_{control} = \{\theta_{ci}\ ,\ \beta_{ci}\ ,\ \tau_0(s_0)\ ,\ f_0(s_0)\}\,. \tag{37b}$$

Given a couple $\Omega_{robot}, \Omega_{control}$ one can compute the robot 3D shape $p^*(s)$ (the subscript $_0$ is omitted in the following for clarity):

$$p^*(s) = \mathcal{M}_{CTCR}(\Omega_{robot}, \Omega_{control}, s)\,, \tag{38}$$

where the function $\mathcal{M}_{CTCR}(\ \ )$ implements the model described in section 3.1. Let us now consider a change $\Delta\Omega_{control} = \{\Delta\theta_{ci}, \Delta\beta_{ci}, \Delta\tau_0(s_0), \Delta f_0(s_0), \Delta\tau_0(L), \Delta f_0(L)\}$ (see Fig 5). One can solve again the model and obtain a deformed shape :

$$p_{model}(s) = \mathcal{M}_{CTCR}(\Omega_{robot}, \Omega_{control} + \Delta\Omega_{control}, s)\,. \tag{39}$$



$* \ \forall i$ $** \ \forall i, \forall s_0$

Figure 4: Principle of the LLDPM applied to a CTCR to compute the Compliance matrix $C_{s_0}(s)$.

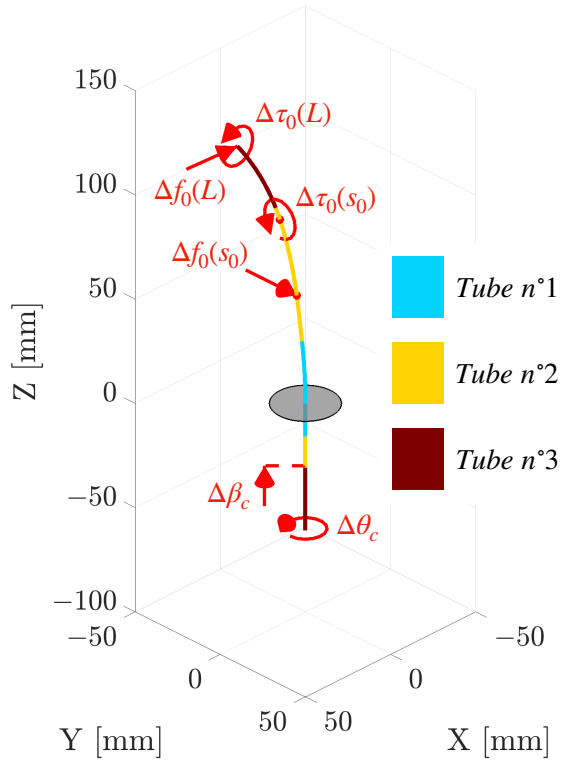


Figure 5: Visualization of the variations applied to the CTCR shape $\Delta\Omega_{control}$.

Alternatively, assuming that $\Delta\Omega_{control}$ is small, one can exploit the Generalized Compliance matrix $C_{s_0}(s)$ and the robot Jacobian $J(s)$ to compute the deformed shape :

$$p_{jacob}(s) \approx p^*(s) + \sum_{0\leq s_0\leq L} C_{s_0}(s)\Big|_{p_0(s)} \begin{bmatrix}\Delta\tau_0(s_0)\\ \Delta f_0(s_0)\end{bmatrix} + \sum_{i\in nbT(s)} J(s)\Big|_{p_0(s)} \begin{bmatrix}\Delta\theta_{ci}\\ \Delta\beta_{ci}\end{bmatrix} \tag{40}$$

where $\bullet|_{p_0(s)}$ specifies that only the position components of $C_{s_0}(s)$ and $J(s)$ are used.

To summarize, given an initial robot configuration we compute the nominal shape $p^*(s)$ as well as two deformed shapes when considering joint variables or external forces variations: $p_{model}(s)$ by solving a new BVP and $p_{jacob}(s)$ using a linear approximation of the deformations and exploiting the Generalized Compliance matrix and robot Jacobian matrices computed using our proposed LLDPM method.

To compare these deformed shapes, the distance $d(\ )$ will be used to quantify the differences. Given two shapes $p_1(s)$ and $p_2(s)$, it is defined as:

$$d\left(p_1(s), p_2(s)\right) = \max_s \left\| p_1(s) - p_2(s) \right\|_2 \tag{41}$$

Operator $d(\ )$ can be used to compute three distances for a given simulation with inputs Ω_{robot}, $\Omega_{control}$, and $\Delta\Omega_{control}$:

$$\mathcal{D}^*_{model} = d\left(p_{model}(s), p^*(s)\right) \tag{42a}$$

$$\mathcal{D}^*_{jacob} = d\left(p_{jacob}(s), p^*(s)\right) \tag{42b}$$

$$\mathcal{D}^{jacob}_{model} = d\left(p_{jacob}(s), p_{model}(s)\right) \tag{42c}$$

An illustration of the three shapes and the distances $\mathcal{D}^*_{model}$, $\mathcal{D}^*_{jacob}$, $\mathcal{D}^{jacob}_{model}$ is given in Fig. 6. In most cases, this distance matches the distance between the tips of the two shapes.

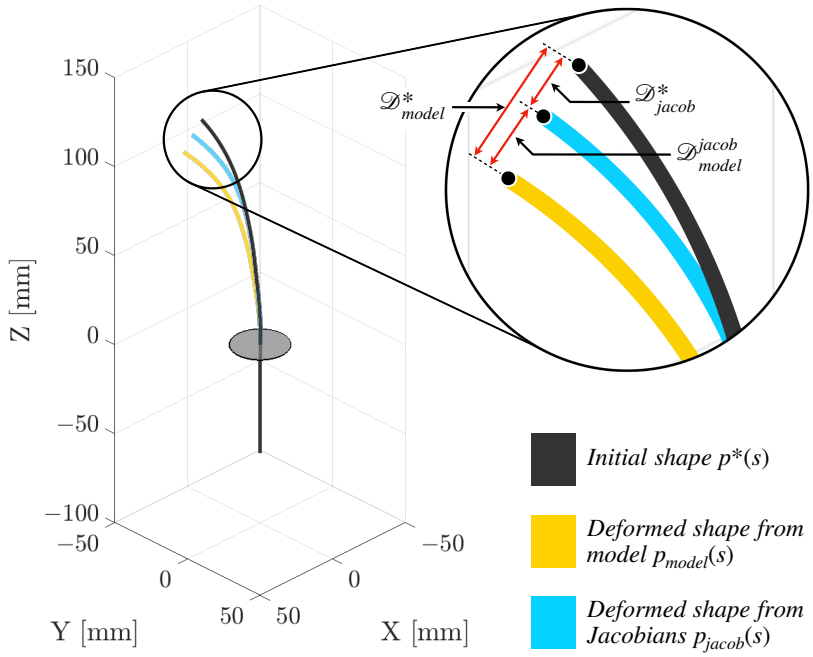


Figure 6: Distances measured between the initial shape $p^*(s)$, the deformed shape resulting from the Jacobians $p_{jacob}(s)$ and the one resulting from the model $p_{model}(s)$

4.2. CTCR Model Parameters and implementation

The validation will be done for a single CTCR. This robot has been chosen to maximize the set of joint values that converge through the BVP solving, i.e. joint values that lead to stable shapes (i.e. avoiding elastic instabilities). The chosen CTCR is composed of three tubes that are described in Table 4.

Feature	**Units**	**Value**		
		Tube $n°1$	Tube $n°2$	Tube $n°3$
Straight length L_{ri}	mm	15	75	135
Curved length L_{ci}	mm	30	45	45
Total length L_i	mm	45	120	180
Radius of curvature R_{ci}	mm	150	100	50
Stiffness kx_i	N.mm^2	$1.5 \cdot 10^6$	$4.0 \cdot 10^5$	$2.0 \cdot 10^5$
Poisson's ratio	-	0.3	0.3	0.3

Table 4: Parameters Ω_{robot} of the robot simulated for the validation.

Shooting method parameters can have a major impact on the BVP solving efficiency. In this study, we implement the model in MATLAB R2023a [44]. The BVP are solved using a Levenberg-Marquardt algorithm. Using equations (31a)-(31f) and (35) we compute $B_{y_u(0)} = \frac{\partial b}{\partial y_u(0)}$ which is supplied to the solver as the optimization Jacobian of the objective function, and avoids resorting to numerical differentiation. The stopping criterion for convergence of the algorithm is defined as $\|b\|_\infty < \epsilon$. Since all the ODE are integrated using a first-order scheme, a finer discretization leads to an increase in the number of integration steps. Therefore, a coarse discretization is used all along the robot neutral axis, and a finer one is used near the discontinuity points $s \in \{0_i,\ max(0,\ \beta_{ci} - L_{ci}),\ \beta_{ci}\}$. All the model parameters are listed in Table 5.

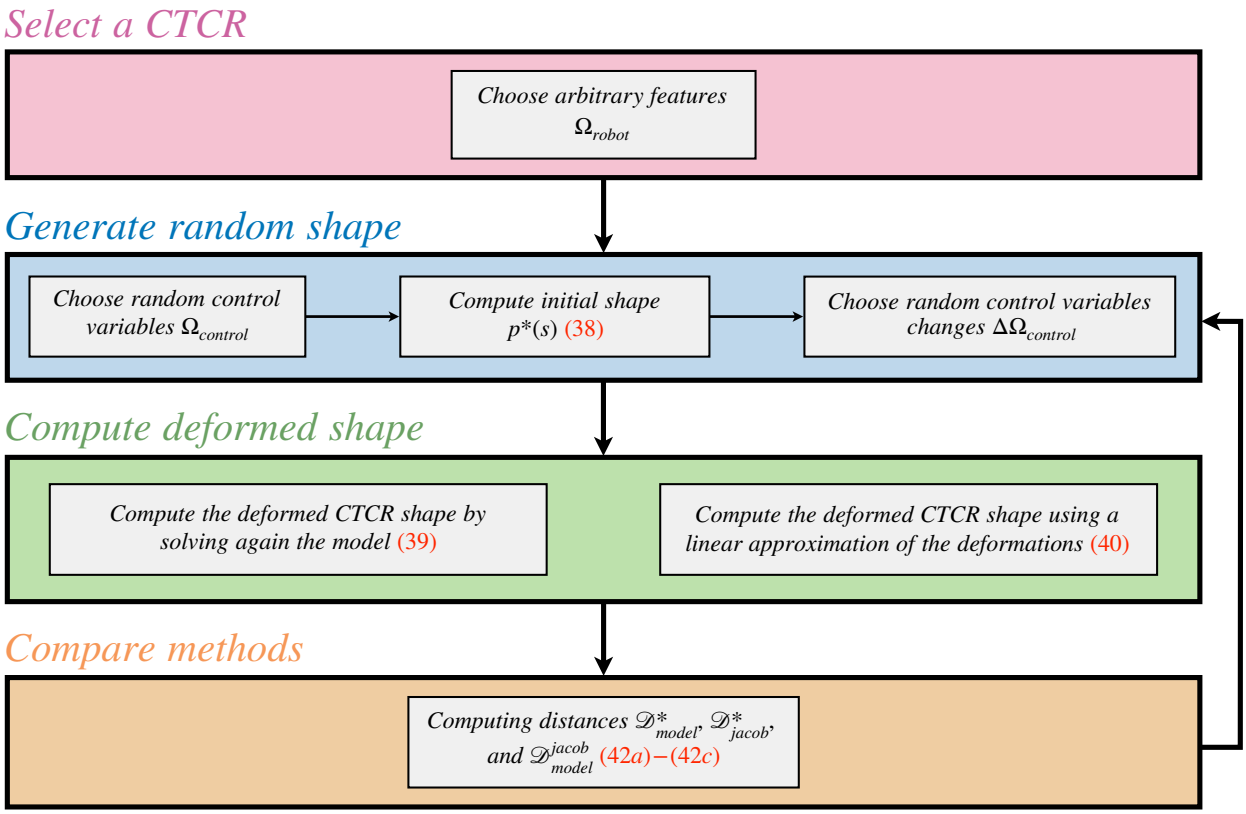


Figure 7: Validation procedure used to compare the deformations estimated solving two times the BVP and by using the robot differential kinematics.

Settings	**Units**	**Value**
Stopping criterion ϵ [1]	-	10^{-20}
Discretization step $\Delta(s)$ away from discontinuity points	mm	2
Number of points before and after each discontinuity points	-	5
Discretization step $\Delta(s)$ for discontinuity points	mm	10^{-7}
Total number of discretization points	-	175
Initial $y_u(0)$ value	-	0

Table 5: Parameters of the shooting method used to solve the CTCR model and discretization parameters.

4.3. Large-scale Numerical Validation on Simulated CTCR

To validate that the LLDPM equations provide accurate and robust results for $C_{s_0}(s)$ and $J(s)$, the validation procedure will be carried out on a wide range of random initial shapes $p^*(s)$. First, a set of random values for the set $\Omega_{control}$ is drawn, so as to challenge the LLDPM results on various state vector initial values $y_u(0)$. Joint values θ_{ci} and β_{ci} are randomly sampled in the range $[-\pi, \pi]$ and $[0, L_{nbT}]$ respectively. For values of $\tau_0(s_0)$ and $f_0(s_0)$, we

[1] A stopping criteria $\epsilon \leq 10^{-5}$ is practically sufficient for solving a standard BVP. However, in the next section 4.3, the CTCR shape variations are measured for several parameters variations such as $\Delta\theta_{ci} = 10^{-6}$ rad. In order to make the impact of the threshold ϵ on shape variations negligible, a very conservative value $\epsilon = 10^{-20}$ has been used.

distinguish two cases, which will be referred to in the following as the *unloaded* and the *preloaded* cases. In the unloaded case, we simply set $\tau_0(s_0)$ and $f_0(s_0)$ to zero for all values of $s_0 \in [0, L]$. In the preloaded case, we consider that external forces are applied to the CTCR, yielding a reference shape $p^*(s)$ deformed by static external forces. The rest of the procedure depicted on Figure 7 is then applied to validate the LLDPM in preloaded conditions. Values of $\tau_0(s_0)$ and $f_0(s_0)$ for $s_0 \in [0, L]$ are uniformly drawn such that the norm of the different elements do not exceed the values listed in Table 6. Such preload magnitudes have been chosen so that they can induce significant deformations. One should notice that the external loads $\tau_0(s_0)$ and $f_0(s_0)$ are included in both the robot model (7) and the LLDPM computations (17).

Once all values are set, we can construct the set of initial control variables $\Omega_{control}$, and compute $p^*(s)$ by solving the BVP. One then needs to generate the $\Delta\Omega_{control}$ set of values in order to be able to compute $p_{model}(s)$ and $p_{jacob}(s)$. Similarly to the procedure described above, variations $\Delta\Omega_{control}$ are uniformly sampled. However, we applied specific care in ensuring that variations of individual components of $\Delta\Omega_{control}$ (i.e. $\Delta\theta_{c_i}$, $\Delta\beta_{c_i}$, etc.) generate shape deformations with the same order of magnitude. Balancing the individual deformations guarantees that the overall deformed shape is not weighted too heavily by particular components of $C_{s_0}(s)$ or $J(s)$. Preliminary simulations have been done to define properly the $\|\Delta\Omega_{control}\|$ extremal values listed in Table 6.

Variations ranges

Description	**Designation**	**Units**	**Maximum**
Rotation of the tubes	$\|\|\Delta\theta_{c_i}\|\|$	rad	$2 \cdot 10^{-6}$
Translation of the tubes	$\|\|\Delta\beta_{c_i}\|\|$	mm	$5 \cdot 10^{-5}$
Tip bending moment	$\|\|\Delta\tau_0(L)\|\|$	N.mm	$8 \cdot 10^{-1}$
Tip force	$\|\|\Delta f_0(L)\|\|$	N	$2 \cdot 10^{-2}$
Abscissa of the loaded point	s_0	mm	L
Non-tip bending moment	$\|\|\Delta\tau_0(s_0)\|\|$	N.mm	$4 \cdot 10^{-1}$
Non-tip force	$\|\|\Delta f_0(s_0)\|\|$	N	$8 \cdot 10^{-3}$

Preload ranges

Description	**Designation**	**Units**	**Maximum**
Tip bending moment	$\|\|\tau_0(L)\|\|$	N.mm	$1 \cdot 10^{2}$
Tip force	$\|\|f_0(L)\|\|$	N	$1 \cdot 10^{0}$
Preloaded section length	$\|\|l_{max} - l_{min}\|\|$	mm	$2 \cdot 10^{1}$
Non-tip bending moment	$\|\|\tau_0(s_0)\|\|$	N.mm	$1 \cdot 10^{3}$
Non-tip force	$\|\|f_0(s_0)\|\|$	N	$1 \cdot 10^{1}$

Table 6: Summary of the deviation between the initial shape $p^*(s)$ and the deformed shape resulting from the Jacobians $p_{jacob}(s)$ and the model $p_{model}(s)$ for an unloaded CTCR.

The simulated variations include variations on the tip loads, on a single punctual non-tip load with a random application point s_0, and the joint values. This means that the presented large-scale validation uses simultaneously $C_{s_0=L}(s)$, $C_{s_0\neq L}(s)$ and $J(s)$. For both loaded and unloaded CTCR, 5000 random shapes $p^*(s)$ are simulated, each of them is deformed and estimated through $p_{model}(s)$ and $p_{jacob}(s)$ and the distances $\mathcal{D}^*_{model}$, $\mathcal{D}^*_{jacob}$ and $\mathcal{D}^{jacob}_{model}$ are systematically computed. An overview of the CTCR shapes $p^*(s)$ sampled for our validation is shown in Fig 8. The simulated deformations for the unloaded and preloaded CTCR are plotted in Fig 9. The simulation statistics are also provided in Table 7.

It can be noticed that the relative difference between the two methods $\mathcal{D}^{jacob}_{model}$ is at least one order of magnitude lower than the absolute deformations $\mathcal{D}^*_{model}$ and $\mathcal{D}^*_{jacob}$ for 99% of the unloaded robots and 90% of the preloaded ones. Instabilities can appear with preloaded continuum robots, which makes the preloaded CTCR success rate

surprisingly promising. It can therefore be modestly admitted that the LLDPM provides precise estimations of the CTCR deformations compared to simulations based exclusively on the robot model.

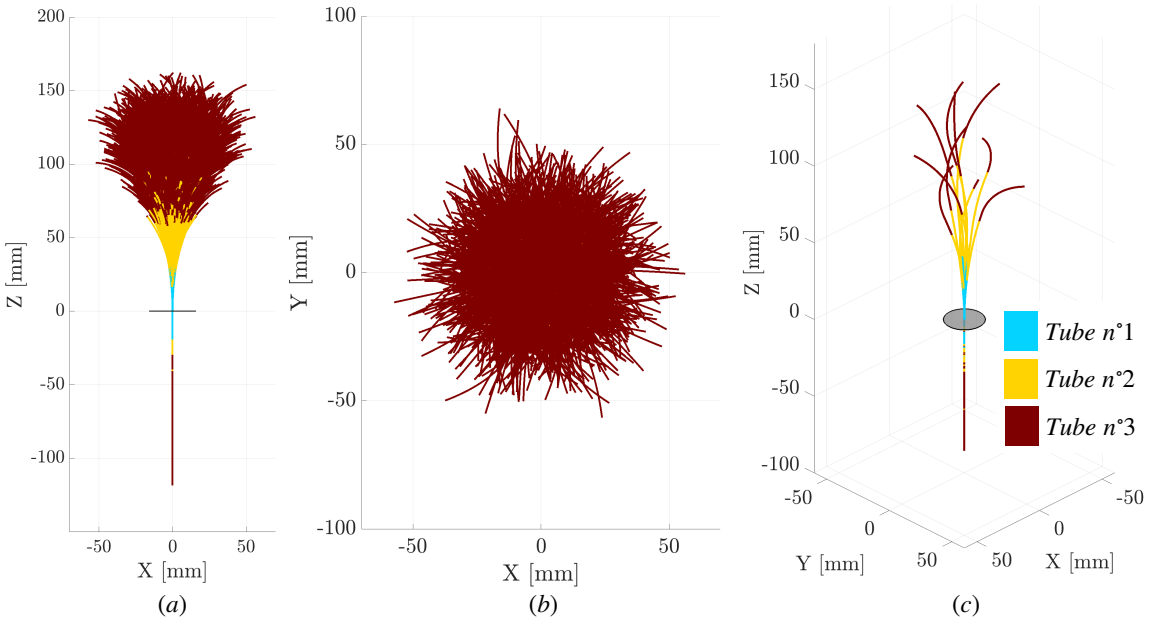


Figure 8: Overview of the CTCR shapes $p^*(s)$ used for the validation process. a) and b) show respectively the side view and top view of the 5000 shapes. c) shows a sample of 10 shapes highlighting the wide variety of simulated shapes, including "C-shapes" and "S-shapes".

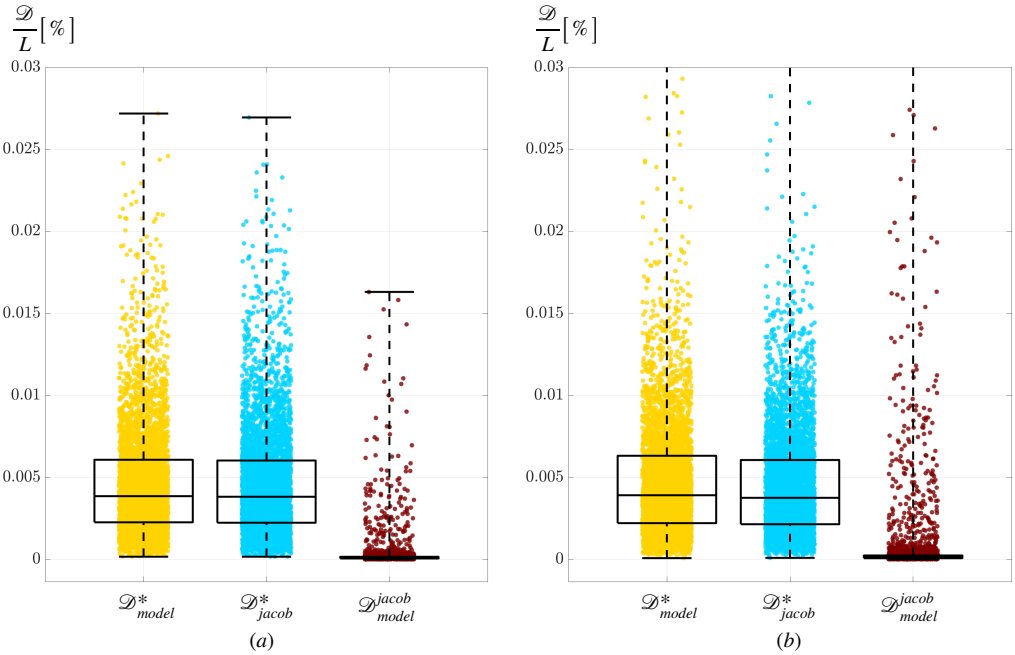


Figure 9: Deformations simulated using the BVP ($\mathcal{D}^*_{model}$) and by using $C_{s_0}(s)$ and $J(s)$ resulting from the LLDPM ($\mathcal{D}^*_{jacob}$) . The difference between these two estimations is measured using the relative distance $\mathcal{D}^{jacob}_{model}$. The results are shown for 5000 random shapes of an unloaded CTCR (a) and for a preloaded CTCR (b).

Unloaded CTCR			
Measurements	$\frac{\mathcal{D}^*_{model}}{L}[\%]$	$\frac{\mathcal{D}^*_{jacob}}{L}[\%]$	$\frac{\mathcal{D}^{jacob}_{model}}{L}[\%]$
Minimum [%]	$1.7 \cdot 10^{-4}$	$1.7 \cdot 10^{-4}$	$7.5 \cdot 10^{-6}$
Median [%]	$3.9 \cdot 10^{-3}$	$3.8 \cdot 10^{-3}$	$1.1 \cdot 10^{-4}$
90th Percentile [%]	$9.0 \cdot 10^{-3}$	$8.9 \cdot 10^{-3}$	$2.5 \cdot 10^{-4}$
99th Percentile [%]	$1.7 \cdot 10^{-2}$	$1.7 \cdot 10^{-2}$	$4.2 \cdot 10^{-3}$
Maximum [%]	$2.7 \cdot 10^{-2}$	$2.7 \cdot 10^{-2}$	$1.6 \cdot 10^{-2}$
Preloaded CTCR			
Measurements	$\frac{\mathcal{D}^*_{model}}{L}[\%]$	$\frac{\mathcal{D}^*_{jacob}}{L}[\%]$	$\frac{\mathcal{D}^{jacob}_{model}}{L}[\%]$
Minimum [%]	$8.5 \cdot 10^{-5}$	$1.0 \cdot 10^{-4}$	$8.7 \cdot 10^{-6}$
Median [%]	$4.0 \cdot 10^{-3}$	$3.8 \cdot 10^{-3}$	$1.4 \cdot 10^{-4}$
90th Percentile [%]	$9.6 \cdot 10^{-3}$	$9.0 \cdot 10^{-3}$	$5.2 \cdot 10^{-4}$
99th Percentile [%]	$3.1 \cdot 10^{-2}$	$1.5 \cdot 10^{-2}$	$2.7 \cdot 10^{-2}$
Maximum [%]	$5.6 \cdot 10^{0}$	$3.6 \cdot 10^{-2}$	$5.6 \cdot 10^{0}$

Table 7: Statistics of the absolute deformations $\mathcal{D}^*_{model}$ and $\mathcal{D}^*_{model}$ and the relative difference $\mathcal{D}^*_{model}$ between the two estimations. The results have been produced by 5000 random simulations on an unloaded robot (a) and on a preloaded robot (b).

4.4. Computational performance assessment

The whole idea behind the proposed LLDPM arises from a need to optimize the number of operations involved in integrating the partial derivatives when computing the Generalized Compliance matrix. It has been estimated in section 2.4 that the number of operations saved can rise to almost 70%, and increases as the robot's discretization becomes finer. This gain only takes into account the integration operations and, therefore, does not give a full view of the computational performance. The performance may be impacted by logical operations required to manage the partial derivatives (see the "Manage derivatives" block in Figure 4), but also by various numerical operations in the implementation itself. This naturally suggests directly comparing the computation times of the different methods when computing $C_{s_0}(s)$. In order to do so, we implemented and compared four methods from the state of the art based on Finite Differences (FD) and Derivative Propagation Methods (DPM) :

- $FD\ BVP(C_{s_0})$ is the typical FD method from the state of the art [17, 18], adapted for computing $C_{s_0}(s)$. It consists in computing directly $C_{s_0}(s)$ using FD (43). It requires solving a new BVP problem for each couple $\{s, s_0\}$ in order to estimate $C_{s_0}(s)$:

$$C_{s_0}(s) \approx \frac{T_0^\vee\Big(s, w_0(s_0) + \Delta w_0(s_0)\Big) - T_0^\vee\Big(s, w_0(s_0)\Big)}{\Delta w_0(s_0)} \tag{43}$$

- $FD\ IVP(C_{s_0})$ is the adaptation of the *IVP Finite Differences* method from [30] for the Generalized Compliance matrix. This method performs FD at the IVP level to estimate variations of $E_{w_0(s_0)}(s)$ and $B_{w_0(s_0)}$ based on equations (44a)-(44b). $C_{s_0}(s)$ is then computed using (15). No BVP evaluations are required in the FD scheme, it is then expected to be faster than $FD(BVP)$.

$$E_{w_0(s_0)}(s) \approx \left.\frac{T_0^\vee\Big(s, w_0(s_0) + \Delta w_0(s_0)\Big) - T_0^\vee\Big(s, w_0(s_0)\Big)}{\Delta w_0(s_0)}\right|_{b\neq 0} \tag{44a}$$

$$B_{w_0(s_0)} \approx \frac{b\Big(w_0(s_0) + \Delta w_0(s_0)\Big) - b\Big(w_0(s_0)\Big)}{\Delta w_0(s_0)} \tag{44b}$$

- $DPM(C_{s_0})$: This method has been also adapted from the one introduced by Rucker et al. in [30]. As discussed in section 2.3, the values of $E_{w_0(s_0)}(s)$ and $B_{w_0(s_0)}$ are computed by integrating the literal partial derivatives through equations (11a)-(11c). Then $C_{s_0}(s)$ is computed using (15).

- $LLDPM(C_{s_0})$: This is the method proposed in section 2.4 of this paper. It consists in first integrating the Low-Level Partial Derivatives using equations (17), then reconstructing at once the $E_{w_0(s_0)}(s)$ and $B_{w_0(s_0)}$ values and finally computing $C_{s_0}(s)$ using (15).

The computation time required to compute $C_{s_0}(s)$ using a method $\mathcal{M}$ (where $\mathcal{M}$ is one of the above-mentioned methods) is denoted $T_{\mathcal{M}(C_{s_0})}$. Similarly to the gain of operations defined in (20), the computation time saved using the LLDPM compared to the DPM is denoted $T_{Gain(C_{s_0})}$. These four methods were implemented in the same framework in MATLAB and converted to C++ through the MATLAB Coder application and MEX functions. Figure 10 shows the results for a 3-tubes CTCR in a given random configuration, for a number of discretization points N ranging from 30 to 100 (which is the range observed in the literature, see Table 1). For each N, the average computational time for 10 runs was measured for each method. The laptop computer used for this comparison is a MacBook Pro with a 3.22 GHz 10 cores Apple M1 Pro processor.

As expected, one can see in Fig. 10 that the FD-based methods are much slower than the derivative propagation methods. Among the derivative propagation methods, our proposed LLDPM method consistently outperforms the DPM method adapted from [30] by a large margin. As can be seen in Fig. 10b, the computational time gain increases with the number of discretization points, which is perfectly in line with our theoretical derivations in section 2.4. For 100 points, the LLDPM computes the Generalized Compliance matrix almost an order of magnitude faster (70%) than the adaptation of the DPM. With $N = 100$, the LLDPM could compute $C_{s_0}(s)$ at 100 Hz while the DPM would be limited to 30 Hz. In comparison, FD methods compute $C_{s_0}(s)$ at 0.1 to 1 Hz.

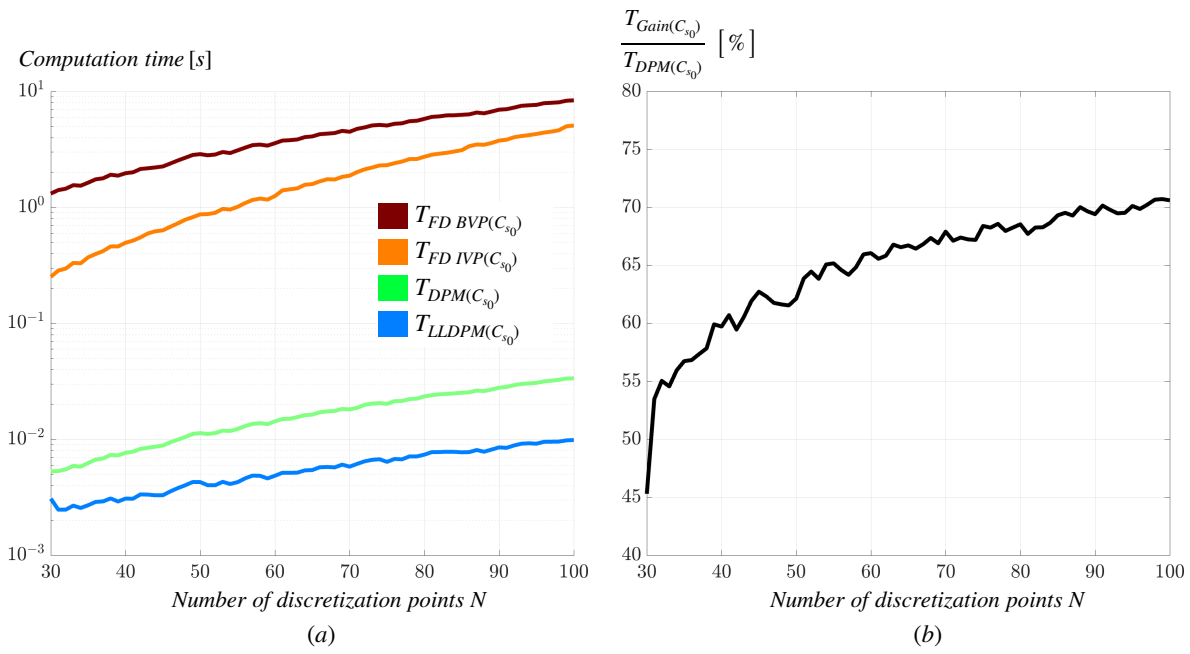


Figure 10: a) Evolution of the Generalized Compliance matrix computation time for the four possible methods presented in 4.4 w.r.t. the number of discretization points N for a 3-tube CTCR. b) Evolution of the computation time saving when using the LLDPM compared to an adaptation of the DPM method from [30].

5. Discussion: Generalized Compliance Matrix Properties and Applications

The results obtained with our large-scale numerical validation in section 4 show that our proposed LLDPM method robustly and accurately computes the Generalized Compliance matrix $C_{s_0}(s)$ and the robot Jacobian $J(s)$ in most cases, with a computational efficiency outperforming existing methods from the state of the art. Some particular cases, however, exhibit larger errors, especially in the preloaded case where 10% of the simulations show a relative error in the same order of magnitude as the displacement. An advanced study could be carried out to quantify the correlation between $y_u(0)$, Δf_0, and the linearization domain properties of $C_{s_0}(s)$ and $J(s)$. Besides, external forces applied in the preloaded case can push the Cosserat rod model of CTCRs to its limits, as such forces may bring the robot closer to singularities and elastic instabilities [19]. One should also note that the observed relative errors have a maximum of 5.6% (see Table 7), which is close to the errors obtained by Cosserat rod CTCR models when validated against real-world data [8, 17]. Such errors are typically due to non-linear phenomena such as friction or non-linear material elasticity, which are not included in our continuum robot model. While a few continuum robot models include friction [45], they are quite complex and often specific to a given implementation. Extending our method to such models is therefore beyond the scope of the present paper. Despite this limitation, the proposed LLDPM method could serve as a basis for several applications, typically in feedback control [15], design optimization [46] or motion planning. It would be interesting, in future work, to investigate how the Generalized Compliance matrix behaves in the presence of such phenomena.

The large-scale validation was carried out to emphasize the performances obtained when using $C_{s_0}(s)$ and $J(s)$ to estimate CTCR deformations. To avoid an exponential increase in validation cases, we included a unique external load $w_0(s_0)$ per simulation, with s_0 randomly drawn in $[0, L]$. A continuum robot is, however, typically fixed at its base and free at its end, which means a force applied at or close to the tip should generate larger deformations than a force applied at or close to the base. Fig. 11 shows results of simulations for a single load case with values of $s_0 \in [L/4, L/2, 3L/4, L]$, for various values of $\|\Delta f_0(s_0)\|$. As before, each data point in Fig. 11a is the average error ($\mathcal{D}_{jacob}^{model}$ normalized by the length of the robot) for 50 simulations in random configurations. These simulations show that the error for tip loads is constantly 10% higher than for proximal loads up to 1 N magnitude force. Beyond this threshold, the tip error increases faster than the proximal ones. This correlation was expected since the greater the amplitude of the forces, the greater the deformations and, as a result, the greater the error induced by the linearization of $C_{s_0}(s)$. However, the errors stay within acceptable bounds even for large forces, up to 100 N if applied on the lower part of the robot (see Fig. 11b).

The Generalized Compliance matrix being a locally linearized relation, one can, however, apply equation (40) with more than one external force along the robot. Fig. 12a shows results for an increasing number of loads $nb_{loads} \in \mathbb{N}^+$ on a random reference configuration. In this case, for each point s_0 where a force is applied, the force variation $\Delta f(s_0)$ has a random orientation and a fixed magnitude. Each data point is the average of 50 simulations. This example shows that the variations of magnitudes $\|\Delta f_0(s_0)\|$ have more impact on the deformations errors than the number of loads nb_{loads}. Interestingly, one can also note that the Generalized Compliance matrix can be used with $\|\Delta f_0(s_0)\| < 1$ N, or with larger forces if nb_{loads} is not too large. Even five forces of 10 N of magnitude applied to the robot generate a relative error of less than 1% of the robot length (as also shown Fig. 12a), which may be very interesting for control or planning applications. Increasing the applied forces further, however, yields large deviations, as shown in Fig. 12b.

In section 4.4, it has been shown that the LLDPM is more efficient than the state-of-the-art method adaptations to compute the Generalized Compliance matrix. The computational times have been obtained on C++ programs generated by the MATLAB Coder application. It may be possible to optimize the code to further increase the LLDPM computational efficiency. Furthermore, results show that the computational times are affected by the number N of discretization points (which was also apparent in our theoretical estimation in section 2.3). In our study, we imposed a constant discretization step $\Delta(s)$ of 2 mm, with added points close to discontinuities (see Table 5). In future work, one could investigate a non-constant and adaptive value of $\Delta(s)$, for instance as a function of the local curvature, in order to reduce the computational time of our method while minimally impacting its precision.

The proposed Generalized Compliance matrix definition is a general robot modeling concept that makes particular sense for medical continuum robots. Whereas in this paper we focused on CTCR, it would be valuable to develop the method for other types of continuum robots. In practice, the method can be adapted to any continuum robot model which can be put in the form of an ODE integrated from the base to the tip of the robot, and which is in the form $\dot{y}(s) = \mathcal{F}(y, s)$ (see details in sections 2.1 and 2.4). This is the case for many types of continuum robots such

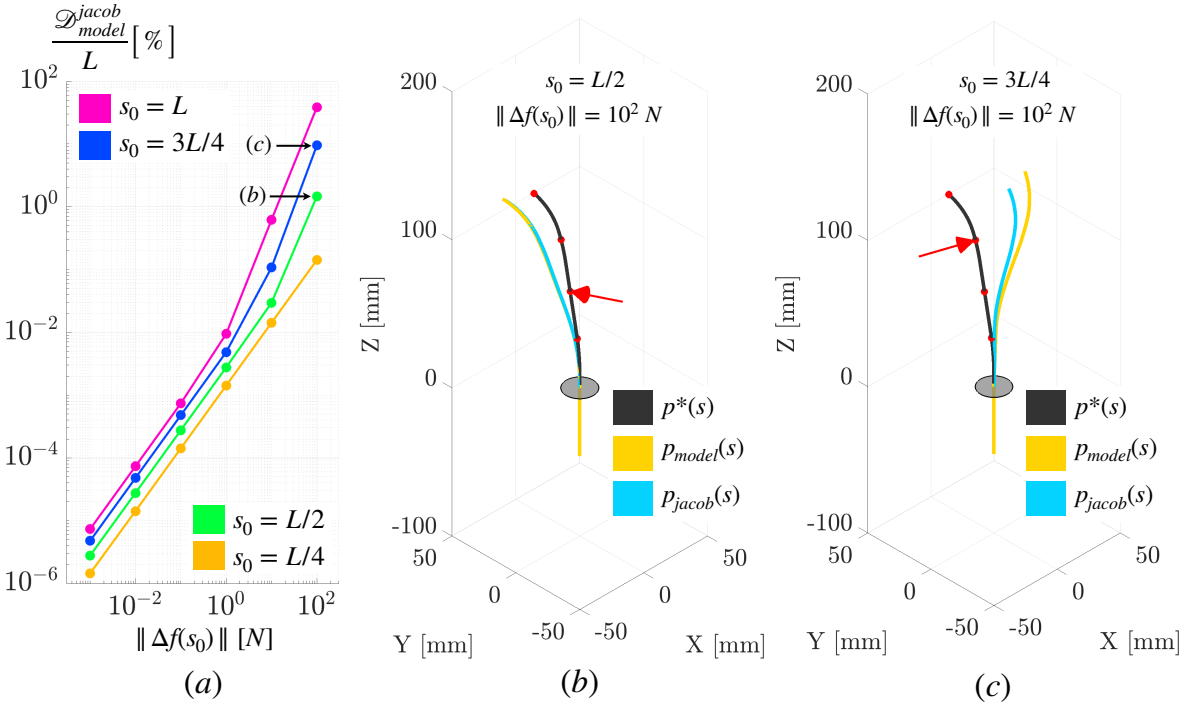


Figure 11: a) Investigations of the impact of $\|\Delta f_0(s_0)\|$ on the deformations error for several loaded points s_0. Representation of the resulting deformations for $100N$ magnitudes on middle point $s_0 = L/2$ (b) and top point $s_0 = 3L/4$ (c).

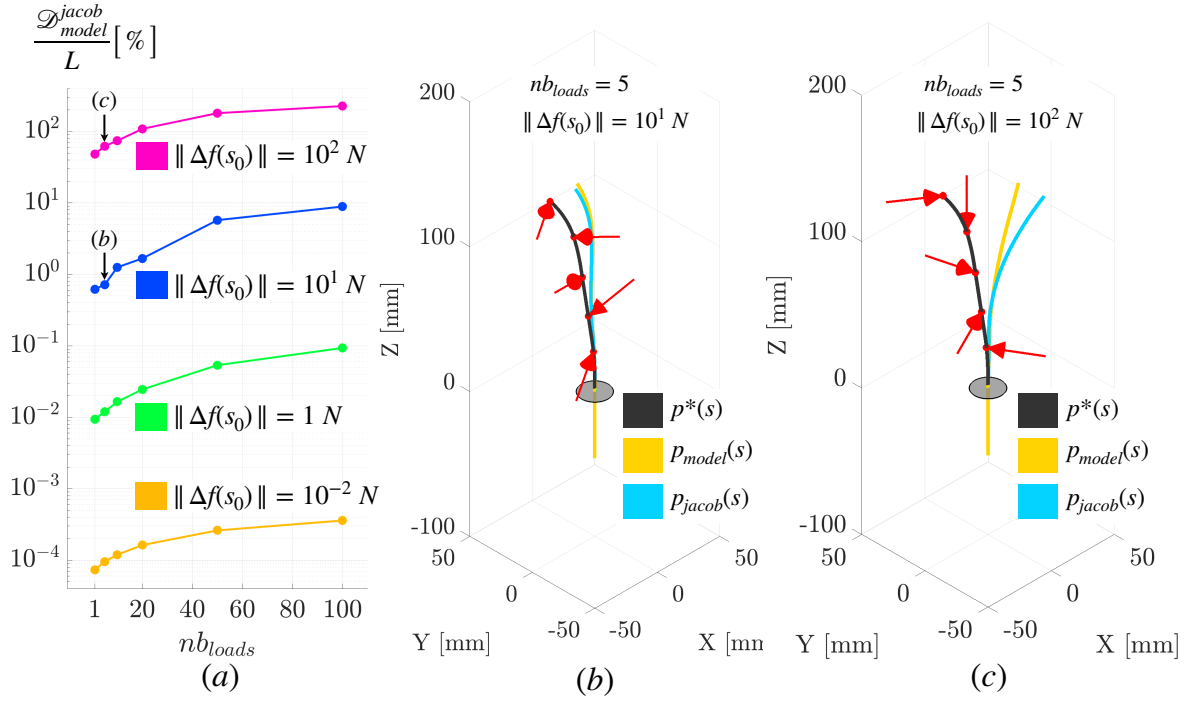


Figure 12: a) Investigations of the impact of the number nb_{loads} of external forces on the deformations error for several magnitudes $\|\Delta f_0(s_0)\|$. Representation of the resulting deformations for 5 applied forces of 10 N (b) and 100 N (c) magnitudes.

as MBCR [12], CPPR [38], parallel continuum robots [37] and pneumatically-driven soft and continuum robots [33]. However, some additional developments would be needed to apply the LLDPM for TACR since the ODE in the model is in the form $M(y, s)\dot{y} = \mathcal{F}(y, s)$, where M is a state-dependent mass matrix [47, 33]. This constitutes a problem for applying the LLDPM since no analytical expression for the inverse of M can be obtained, and model solving in the state of the art relies on numerically inverting it [47, 33]. The derivation should nevertheless be possible by relying on the relation $\frac{\partial}{\partial \chi}(M^{-1}) = -M^{-1}\frac{\partial M}{\partial \chi}M^{-1}$. However, this would rely on obtaining M^{-1} through numerical inversion, and it is unclear how this method would behave in terms of numerical stability. Such a study is therefore beyond the scope of the present paper. Besides, it would also be interesting to investigate the effect of large S-shape deformations due to external forces acting on TACR, as observed in [48], on the accuracy of the Generalized Compliance matrix computed through the LLDPM.

Other approaches than the usual Newtonian form can also be used for modeling continuum robots, such as Lagrangian-based methods frequently used in soft robotics. While it was shown recently that for TACR both approaches are mathematically equivalent [10], proposing a formulation of our LLDPM in the Lagrangian framework remains an open problem. Similarly, extension of the LLDPM method for other modeling frameworks (see for instance the work from [48] which applies the virtual works principle under a constant curvature assumption) and/or in the presence of stiffness modulating elements in the robot structure [48, 6] is an interesting future work.

Numerous applications considering non-tip forces and compliance analysis have been proposed in the state-of-the-art. All the existing non-tip force sensing algorithms correlate the robot shape deformations with the applied external force using iterative and numeric methods based directly on the robot model [21, 23, 35, 49]. Another approach recently developed by Ferguson et al. [50] consists in statistically estimating external forces using an adaptation of a continuous-time batch estimation while computing the robot shape. Using the Generalized Compliance matrix can simplify and speed up the force estimation problem by using a differential kinematics approach. Indeed the local wrench variation $\Delta w_0(s_0)$ is directly linked to the shape variation $\Delta T_0(s)$ through the relation $\Delta w_0(s_0) \approx C_{s_0}(s)^{-1}\Delta T_0^{\vee}(s)$ for all (s, s_0). This application scenario is currently under investigation and experimental studies are envisioned.

The proposed Compliance definition could also have applications in compliance control strategies. Since Compliance is intrinsically dependent on the robot shape, one could investigate the properties and distribution of Compliance according to force application points that are detected or foreseen in trajectory planning. Therefore, it may be possible to modulate the Compliance using an active robot-internal actuation such as electrodes along tubes of a CTCR [51] or leverage mechanisms on a rod-driven continuum robot [48]. More generally, the Generalized Compliance could be an asset for mechanical design optimization, similar to the work of Naselli et al. developed for soft robots [29] or Ha et al. for stability-based CTCR design optimization [19].

6. Conclusion

Since their emergence, flexible robots have been regarded as having significant potential, in particular for use in medical applications. Models such as the robot Jacobian and the tip Compliance matrices were soon adapted for controlling such robots. Unlike existing methods, which consider the effect of only tip forces and moments on the shape of the robot, this paper presents the concept of Generalized Compliance matrix. The Generalized Compliance matrix is a four-dimensional tensor containing the sensitivity of any point of the robot with respect to any forces and moments applied at any location of the robot.

Existing computation methods can be adapted for computing the Generalized Compliance matrix using a derivative propagation method, but they suffer from the growing number of derivatives concerned since they have not been designed for this purpose. In this work, we introduce a novel method termed Low-Level Derivative Propagation Method (LLDPM). This method has been specially developed for the Generalized Compliance matrix $C_{s_0}(s)$. The LLDPM aims to densify the propagated derivatives into the minimum of ODE and limit the time-consuming integration to the absolute minimum.

The Generalized Compliance matrix and the associated LLDPM are suitable for any Cosserat rod-based continuum robot model. In this paper, we present a case study using a widely studied Concentric Tube Continuum Robot (CTCR). All relevant equations and computation techniques are explained, which eases understanding and reproducibility. A large-scale numerical validation is also carried out on simulated CTCR, illustrating the LLDPM robustness and versatility. The computation time of the LLDPM has been compared to state-of-the-art methods, showing its computational efficiency and potential for real-time control applications.

The application of LLDPM to other widely used continuum robots (TACR, CPPR, and MBCR) and passive structures (Cosserat rods) could yield valuable insights. Finally, it is important to underline that both path planning and motion control of continuum robots could benefit from these developments, allowing for improved performance and accuracy.

Acknowledgments

This work was supported by French State Funds managed by the Agence Nationale de la Recherche (ANR), Grant ANR-18-CE19-0012 (MACROS). Research in medical robotics at the ICube laboratory is partially supported by Grants ANR-10-IAHU-02 (IHU Strasbourg), ANR-11-LABX-0004 (Labex CAMI) and ANR-10-IDEX-0002 (ITI HealthTech) under the framework of the French Investments for the Future Program.

Appendix A. Nomenclature

Operators		
Notation	**Definition sets**	**Description**
$\dot{\bullet}$	–	Derivation w.r.t. to the curvilinear abscissa s
$\hat{\bullet}$ / $\bullet^{\wedge}$	$\mathbb{R}^3$ / $\mathbb{R}^6 \longmapsto \mathfrak{so}(3)$ / $\mathfrak{se}(3)$	Skew symmetric operator
$\bullet^{\vee}$	$\mathfrak{so}(3)$ / $\mathfrak{se}(3) \longmapsto \mathbb{R}^3$ / $\mathbb{R}^6$	Inverse operator of the skew-symmetric operator
$\bullet^{\dagger}$	–	Moore-Penrose pseudo-inverse operator
$\bullet^{T}$	–	Transpose operator
$\| \bullet \|$	$\mathbb{R}^3 \longmapsto \mathbb{R}$	Euclidean standard norm
$\mathcal{F}(\ )$	$\left(\mathbb{R}^M, \mathbb{R}\right) \longmapsto \mathbb{R}^M$	State vector ODE
$Rot_z(\ )$	$\mathbb{R} \longmapsto \mathbb{R}^{3\times3}$	Rotation matrix around the z axis
$d(\ )$	$\left(\mathbb{R}^{3\times N}, \mathbb{R}^{3\times N}\right) \longmapsto \mathbb{R}$	Distance between two 3D curves discretized into N points

Table A.8: Nomenclature of the operators used in this paper.

Indicators	
Notation	**Description**
$\bullet^{*}$	Indicates that the variable refers to an initial state, usually before a deformation happens
$\bullet_i$	Indicates that the variable refers to the tube n°i
$\bullet_0$	Indicates that the variable refers to the tubes assembly
$\bullet\|_{x/y/z}$	Indicates a restriction to one or several components of a vector
$\bullet\|_{b\neq0}$	Indicates that the derivative does not ensure the constraint $b = 0$
$\bullet\|_{b=0}$	Indicates that the derivative ensures the constraint $b = 0$
$\Delta\bullet$	Indicates a variation
$\bullet_{model}$	Indicates that the variable has been computed using the robot model
$\bullet_{jacob}$	Indicates that the variable has been computed using one or several Jacobians

Table A.9: Nomenclature of the indicators used in this paper.

Continumm Robot Variables

Notation	Dimensions	Units	Description
s	$\mathbb{R}$	mm	Curvilinear abscissa counted from $\mathcal{R}_{B_0}$
L	$\mathbb{R}$	mm	Curvilinear length of the robot tip
s_0	$\mathbb{R}$	mm	Curvilinear abscissa describing a point where external loads are applied
$[l_{min}, l_{max}]$	$\mathbb{R}^2$	mm	Robot curvilinear range on which external loads are applied
$C_{s_0}(s)$	$\mathbb{R}^{6\times6}$	-	Generalized Compliance matrix
$C(s)$	$\mathbb{R}^{6\times6}$	-	Tip Compliance matrix $C(s) = C_{s_0=L}(s)$
$J(s)$	$\mathbb{R}^{6\times\bullet}$	-	Robot Jacobian
e_z	$\mathbb{R}^3$	-	z-axis unit vector
$\mathcal{R}_{B_0}$	-	-	General reference frame
$\mathcal{R}_0(s)$	-	-	Body frame parametrized by s and attached to the tubes assembly
$T_0(s)$	SE(3)	-	Homogeneous transformation matrix from $\mathcal{R}_{B_0}$ to $\mathcal{R}_0(s)$
$R_0(s)$	SO(3)	rad	Rotation matrix from $\mathcal{R}_{B_0}$ to $\mathcal{R}_0(s)$
$p_0(s)$	$\mathbb{R}^3$	mm	Translation matrix from $\mathcal{R}_{B_0}$ to $\mathcal{R}_0(s)$
$v_0(s)$	$\mathbb{R}^3$	rad.mm^{-1}	Linear rate of change of $\mathcal{R}_0(s)$
$u_0(s)$	$\mathbb{R}^3$	mm^{-1}	Angular rate of change of $\mathcal{R}_0(s)$
$\xi_0(s)$	$\mathbb{R}^6$	-	Vector that groups $\mathcal{R}_0(s)$ rates of change
$m_0(s)$	$\mathbb{R}^3$	N.mm	Internal bending moment of the robot
$n_0(s)$	$\mathbb{R}^3$	N	Internal force of the robot
$\tau_0(s)$	$\mathbb{R}^3$	N.mm	External bending moment applied to the robot
$f_0(s)$	$\mathbb{R}^3$	N	External force applied to the robot
$w(s_0)$	$\mathbb{R}^6$	-	Vector that groups external loads applied at $s = s_0$
$y(s)$	$\mathbb{R}^M$	-	State vector describing the robot behavior
$y_u(0)$	$\mathbb{R}^N$	-	Sub-vector of $y(0)$ containing the unknown initial values
b	$\mathbb{R}^U$	-	Distal residual ensuring static equilibrium of the robot if $b = 0$
ϵ	$\mathbb{R}$	-	Optimization stopping criterion
q	$\mathbb{R}^6$	-	Vector that groups the actuation variables
χ	$\mathbb{R}^X$	-	Dummy variable representing a given variable of derivation
N	$\mathbb{N}^*$	-	Number of discretization points
Y	$\mathbb{N}^*$	-	Size of $y(s)$
U	$\mathbb{N}^*$	-	Size of $y_u(0)$
B	$\mathbb{N}^*$	-	Size of b
Q	$\mathbb{N}^*$	-	Size of q
X	$\mathbb{N}^*$	-	Size of χ
$E_\chi(s)$	$\mathbb{R}^{6\times\bullet}$	-	Derivative of the homogeneous transformation $T_0(s)$
B_χ	$\mathbb{R}^{U\times\bullet}$	-	Derivative of the residual b
B_u	$\mathbb{R}^{U\times U}$	-	Optimization matrix used to solve the BVP
V_χ	$\mathbb{R}^{M\times\bullet}$	-	Derivative of the state vector $y(s)$
$\Delta(s)$	$\mathbb{R}$	mm	Discretization step at abscissa s
$\mathcal{M}_{CTCR}(\ )$	$-$	-	Representation of the CTCR model which gives back the robot shape

Table A.10: Nomenclature of the continuum robot variables used in this paper.

CTCR Variables

Notation	Dimensions	Units	Description
nbT	$\mathbb{N}^*$	-	Number of tubes
$nbT(s)$	$\mathbb{N}^*$	-	Number of tubes involved at abscissa s
0_i	$\mathbb{R}$	mm	Lowest abscissa of tube i
0_0	$\mathbb{R}$	mm	Lowest abscissa of the tubes assembly
L_{ri}	$\mathbb{R}$ a	mm	Straight length of tube i
L_{ci}	$\mathbb{R}$	mm	Curved length of tube i
L_i	$\mathbb{R}$	mm	Total length of tube i
R_{ci}	$\mathbb{R}$	mm	Radius of curvature of tube i
K_i	$\mathbb{R}^{3\times3}$	N.mm^2	Stiffness matrix of tube i cross section
kx_i	$\mathbb{R}$	N.mm^2	Value of K_i first and second directions
kz_i	$\mathbb{R}$	N.mm^2	Value of K_i third direction
$\theta_i(s)$	$\mathbb{R}$	rad	Twist tube i
$\phi_i(s)$	$\mathbb{R}$	mm^{-1}	Variable describing the tubes piecewise constant curvature
$u_i^*(s)$	$\mathbb{R}^3$	mm^{-1}	3D vector of the non-deformed tube i
$u_i(s)$	$\mathbb{R}^3$	mm^{-1}	3D vector of the deformed tube i
θ_{ci}	$\mathbb{R}$	rad	Rotation of tube i base
β_{ci}	$\mathbb{R}$	mm	Deployment of tube i
Ω_{robot}	-	-	Set of CTCR parameters defining the robot features
$\Omega_{control}$	-	-	Set of CTCR parameters defining variables of the control loop
nb_{loads}	$\mathbb{N}$	-	Number of punctual forces applied to the robot

Table A.11: Nomenclature of the CTCR variables used in this paper.

References

[1] P. E. Dupont, N. Simaan, H. Choset, C. Rucker, Continuum robots for medical interventions, Proceedings of the IEEE 110 (7) (2022) 847–870. doi:10.1109/JPROC.2022.3141338.

[2] N. Simaan, R. Taylor, P. Flint, A dexterous system for laryngeal surgery, 2004 IEEE International Conference on Robotics and Automation 1 (2004) 351–357. doi:10.1109/ROBOT.2004.1307175.

[3] R. J. Hendrick, S. D. Herrell, R. J. Webster, A multi-arm hand-held robotic system for transurethral laser prostate surgery, 2014 IEEE International Conference on Robotics and Automation (2014) 2850–2855doi:10.1109/ICRA.2014.6907268.

[4] F. Nageotte, L. Zorn, P. Zanne, M. De Mathelin, Stras: a modular and flexible telemanipulated robotic device for intraluminal surgery, Elsevier, 2020, pp. 123–146. doi:10.1016/B978-0-12-814245-5.00008-6.

[5] K. Price, J. Peine, M. Mencattelli, Y. Chitalia, D. Pu, T. Looi, S. Stone, J. Drake, P. E. Dupont, Using robotics to move a neurosurgeon's hands to the tip of their endoscope, Science Robotics 8 (82) (2023) eadg6042. doi:10.1126/scirobotics.adg6042.

[6] J. Burgner-Kahrs, D. C. Rucker, H. Choset, Continuum robots for medical applications: A survey, IEEE Transactions on Robotics 31 (6) (2015) 1261–1280. doi:10.1109/TRO.2015.2489500.

[7] J. Lock, G. Laing, M. Mahvash, P. E. Dupont, Quasistatic modeling of concentric tube robots with external loads, 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (2010) 2325–2332doi:10.1109/IROS.2010.5651240.

[8] D. C. Rucker, B. A. Jones, R. J. Webster III, A geometrically exact model for externally loaded concentric-tube continuum robots, IEEE transactions on robotics 26 (5) (2010) 769–780. doi:10.1109/TRO.2010.2062570.

[9] F. Janabi-Sharifi, A. Jalali, I. D. Walker, Cosserat rod-based dynamic modeling of tendon-driven continuum robots: A tutorial, IEEE Access 9 (2021) 68703–68719. doi:10.1109/ACCESS.2021.3077186.

[10] M. Tummers, V. Lebastard, F. Boyer, J. Troccaz, B. Rosa, M. T. Chikhaoui, Cosserat rod modeling of continuum robots from newtonian and lagrangian perspectives, IEEE Transactions on Robotics (2023). doi:10.1109/TRO.2023.3238171.

[11] K. Oliver-Butler, Z. H. Epps, D. C. Rucker, Concentric agonist-antagonist robots for minimally invasive surgeries, Medical Imaging 2017: Image-Guided Procedures, Robotic Interventions, and Modeling 10135 (2017) 270–278. doi:10.1117/12.2255549.

[12] Y. Chen, B. Wu, J. Jin, K. Xu, A variable curvature model for multi-backbone continuum robots to account for inter-segment coupling and external disturbance, IEEE Robotics and Automation Letters 6 (2) (2021) 1590–1597. doi:10.1109/LRA.2021.3058925.

[13] D. Lin, N. Jiao, Z. Wang, L. Liu, A magnetic continuum robot with multi-mode control using opposite-magnetized magnets, IEEE Robotics and Automation Letters 6 (2) (2021) 2485–2492. doi:10.1109/LRA.2021.3061376.

[14] R. J. Webster III, B. A. Jones, Design and kinematic modeling of constant curvature continuum robots: A review, The International Journal of Robotics Research 29 (13) (2010) 1661–1683. doi:10.1177/0278364910368147.

[15] M. T. Chikhaoui, B. Rosa, Modeling and control strategies for flexible devices, Elsevier, 2022, pp. 187–213. doi:10.1016/B978-0-12-821750-4.00008-6.

[16] T. Greigarn, N. L. Poirot, X. Xu, M. C. Çavuşoğlu, Jacobian-based task-space motion planning for mri-actuated continuum robots, IEEE Robotics and Automation Letters 4 (1) (2018) 145–152. doi:10.1109/LRA.2018.2881987.

[17] P. E. Dupont, J. Lock, B. Itkowitz, E. Butler, Design and control of concentric-tube robots, IEEE Transactions on Robotics 26 (2) (2009) 209–225. doi:10.1109/TRO.2009.2035740.

[18] R. Xu, A. Asadian, A. S. Naidu, R. V. Patel, Position control of concentric-tube continuum robots using a modified jacobian-based approach, 2013 IEEE International Conference on Robotics and Automation (2013) 5813–5818doi:10.1109/ICRA.2013.6631413.

[19] J. Ha, F. C. Park, P. E. Dupont, Elastic stability of concentric tube robots subject to external loads, IEEE Transactions on Biomedical Engineering 63 (6) (2015) 1116–1128. doi:10.1109/TBME.2015.2483560.

[20] F. Boyer, V. Lebastard, F. Candelier, F. Renda, Dynamics of continuum and soft robots: A strain parameterization based approach, IEEE Transactions on Robotics 37 (3) (2020) 847–863. doi:10.1109/TRO.2020.3036618.

[21] V. Aloi, K. T. Dang, E. J. Barth, C. Rucker, Estimating forces along continuum robots, IEEE Robotics and Automation Letters 7 (4) (2022) 8877–8884. doi:10.1109/LRA.2022.3188905.

[22] M. A. Diezinger, B. Tamadazte, G. J. Laurent, 3d curvature-based tip load estimation for continuum robots, IEEE Robotics and Automation Letters 7 (4) (2022) 10526–10533. doi:10.1109/LRA.2022.3194680.

[23] O. Al-Ahmad, M. Ourak, J. Vlekken, E. Vander Poorten, Fbg-based estimation of external forces along flexible instrument bodies, Frontiers in Robotics and AI 8 (2021) 718033. doi:10.3389/frobt.2021.718033.

[24] F. Stella, J. Hughes, D. Rus, C. Della Santina, Prescribing cartesian stiffness of soft robots by co-optimization of shape and segment-level stiffness, Soft Robotics 10 (4) (2023) 701–712. doi:10.1089/soro.2022.0025.

[25] A. Bajo, N. Simaan, Kinematics-based detection and localization of contacts along multisegment continuum robots, IEEE Transactions on Robotics 28 (2) (2011) 291–302. doi:10.1109/TRO.2011.2175761.

[26] J. Shi, A. Shariati, S.-A. Abad, Y. Liu, J. S. Dai, H. A. Wurdemann, Stiffness modelling and analysis of soft fluidic-driven robots using lie theory, The International Journal of Robotics Research 43 (3) (2024) 354–384. doi:10.1177/02783649231200595.

[27] M. Khadem, L. Da Cruz, C. Bergeles, Force/velocity manipulability analysis for 3d continuum robots, 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (2018) 4920–4926doi:10.1109/IROS.2018.8593874.

[28] C. B. Black, J. Till, D. C. Rucker, Parallel continuum robots: Modeling, analysis, and actuation-based force sensing, IEEE Transactions on Robotics 34 (1) (2017) 29–47. doi:10.1109/TRO.2017.2753829.

[29] G. A. Naselli, B. Mazzolai, The softness distribution index: towards the creation of guidelines for the modeling of soft-bodied robots, The International Journal of Robotics Research 40 (1) (2019) 197–223. doi:10.1177/0278364919893451.

[30] D. C. Rucker, R. J. Webster, Computing jacobians and compliance matrices for externally loaded continuum robots, 2011 IEEE International Conference on Robotics and Automation (2011) 945–950doi:10.1109/ICRA.2011.5980351.

[31] G. Smoljkic, D. Reynaerts, J. Vander Sloten, E. Vander Poorten, Compliance computation for continuum types of robots, 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (2014) 1066–1073doi:10.1109/IROS.2014.6942690.

[32] S. S. Antman, Problems in nonlinear elasticity, Springer, 1995. doi:10.1007/978-1-4757-4147-6.

[33] J. Till, V. Aloi, C. Rucker, Real-time dynamics of soft and continuum robots based on cosserat rod models, The International Journal of Robotics Research 38 (6) (2018) 723–746. doi:10.48550/arXiv.2309.09019.
[34] F. Renda, M. Giorelli, M. Calisti, M. Cianchetti, C. Laschi, Dynamic model of a multibending soft robot arm driven by cables, IEEE Transactions on Robotics 30 (5) (2014) 1109–1122. doi:10.1109/TRO.2014.2325992.
[35] V. A. Aloi, D. C. Rucker, Estimating loads along elastic rods, 2019 IEEE International Conference on Robotics and Automation (2019) 2867–2873doi:10.1109/ICRA.2019.8794301.
[36] M. Mahvash, P. E. Dupont, Stiffness control of surgical continuum manipulators, IEEE Transactions on Robotics 27 (2) (2011) 334–345. doi:10.1109/TRO.2011.2105410.
[37] A. L. Orekhov, V. A. Aloi, D. C. Rucker, Modeling parallel continuum robots with general intermediate constraints, 2017 IEEE International Conference on Robotics and Automation (2017) 6142–6149doi:10.1109/ICRA.2017.7989728.
[38] J. A. Childs, C. Rucker, A kinetostatic model for concentric push-pull robots, IEEE Transactions on Robotics (2023). doi:10.1109/TRO.2023.3327811.
[39] M. R. Osborne, On shooting methods for boundary value problems, Journal of mathematical analysis and applications 27 (2) (1969) 417–433. doi:10.1016/0022-247X(69)90059-6.
[40] L. F. Shampine, J. Kierzenka, M. W. Reichelt, et al., Solving boundary value problems for ordinary differential equations in matlab with bvp4c, Tutorial notes 2000 (2000) 1–27.
[41] R. M. Murray, Z. Li, S. S. Sastry, A mathematical introduction to robotic manipulation, CRC press, 2017. doi:10.1201/9781315136370.
[42] H. B. Gilbert, D. C. Rucker, R. J. Webster III, Concentric tube robots: The state of the art and future directions, 2016. doi:10.1007/978-3-319-28872-7_15.
[43] C. J. Nwafor, C. Girerd, G. J. Laurent, T. K. Morimoto, K. Rabenorosoa, Design and fabrication of concentric tube robots: a survey, IEEE Transactions on Robotics (2023). doi:10.1109/TRO.2023.3255512.
[44] T. M. Inc., Matlab version: 9.14.0.2337262 (r2023a) update 5 (2023).
URL `https://www.mathworks.com`
[45] J. Ha, G. Fagogenis, P. E. Dupont, Modeling tube clearance and bounding the effect of friction in concentric tube robot kinematics, IEEE Transactions on Robotics (2018). doi:10.1109/TRO.2018.2878906.
[46] C. Bergeles, A. H. Gosline, N. V. Vasilyev, P. J. Codd, P. J. del Nido, P. E. Dupont, Concentric tube robot design and optimization based on task and anatomical constraints, IEEE Transactions on Robotics (2015). doi:10.1109/TRO.2014.2378431.
[47] D. C. Rucker, R. J. Webster III, Statics and dynamics of continuum robots with general tendon routing and external loading, IEEE Transactions on Robotics 27 (6) (2011) 1033–1044. doi:10.1109/TRO.2011.2160469.
[48] C. Yang, S. Geng, I. Walker, D. T. Branson, J. Liu, J. S. Dai, R. Kang, Geometric constraint-based modeling and analysis of a novel continuum robot with shape memory alloy initiated variable stiffness, The International Journal of Robotics Research 39 (14) (2020) 1620–1634. doi:10.1177/027836492091392.
[49] Q. Qiao, G. Borghesan, J. De Schutter, E. Vander Poorten, Force from shape—estimating the location and magnitude of the external force on flexible instruments, IEEE Transactions on Robotics 37 (5) (2021) 1826–1833. doi:10.1109/TRO.2021.3062504.
[50] J. M. Ferguson, D. C. Rucker, R. J. Webster, Unified shape and external load state estimation for continuum robots, IEEE Transactions on Robotics (2024). doi:10.1109/TRO.2024.3360950.
[51] M. T. Chikhaoui, K. Rabenorosoa, N. Andreff, Kinematics and performance analysis of a novel concentric tube robotic structure with embedded soft micro-actuation, Mechanism and Machine Theory 104 (2016) 234–254. doi:10.1016/j.mechmachtheory.2016.06.005.