Cosserat-Based Modeling and Control of Slender Soft Robots

Haihong Li

▶ **To cite this version:**

Haihong Li. Cosserat-Based Modeling and Control of Slender Soft Robots. Automatic. Centrale Lille Institut, 2023. English. NNT : 2023CLIL0015 . tel-04195946

HAL Id: tel-04195946
https://inria.hal.science/tel-04195946v1

Submitted on 5 Sep 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

CENTRALE LILLE

THESE

Présentée en vue
d'obtenir le grade de

DOCTEUR

En

Spécialité : Automatique, Génie Informatique,

Traitement du Signal et des Images

Par

Haihong LI

DOCTORAT DELIVRE PAR CENTRALE LILLE

Titre de la thèse :

Modélisation et Commande de Robots Souples Élancés Basées sur la Théorie de Cosserat

Cosserat-Based Modeling and Control of Slender Soft Robots

Soutenue le 28 Juin 2023 devant le jury d'examen :

Président	Jean-Pierre BARBOT	Professeur, ENSEA
Rapporteur	Yannick AOUSTIN	Professeur, Nantes University
Rapporteur	Holger VOOS	Professeur, University of Luxembourg
Examinateur	Catherine BONNET	Directrice de Recherche, INRIA Saclay
Examinateur	Jean-Pierre BARBOT	Professeur, ENSEA
Examinateur	Guoying GU	Professeur, Shanghai Jiao Tong University
Directeur de thèse	Gang ZHENG	Chargé de Recherche INRIA, HDR, INRIA Lille

Thèse préparée dans le Centre de Recherche en Informatique, Signal et Automatique de Lille,
CRIStAL, CNRS UMR 9189

Ecole Doctorale MADIS-631

Membre du Groupe des Écoles Centrale
Centrale Lille
Cité Scientifique – CS20048 – 59651 Villeneuve d'Ascq Cedex – France
Tél. + 33 (0)3 20 33 53 53 – http://centralelille.fr

Doctor of Philosophy

Cosserat-Based Modeling and Control of Slender Soft Robots

Haihong Li

Contents

Abstract

Soft robotics has become an emergent research area due to its distinct characteristics compared to conventional rigid counterparts. Being made from flexible and compliant material, soft robots present many advantages including high dexterity, safe interactions, and great adaptability. These inherent properties are useful for the current robotic applications, especially for grabbing fragile objects, environment exploration and so on. However, soft robots are characterized by a large number of degrees of freedom (DoFs) and highly nonlinear deformation, which makes their modeling and accurate control tough. As a result, scientific challenges such as the development of an exhaustive theoretical framework for dynamic modeling and real-time control have arisen, with consequent opportunities of new contributions in the field of soft robotics.

To contribute to this area of research, the present thesis develops a discrete modeling technique named piecewise linear strain (PLS) to solve the partial differential equations (PDEs) of Cosserat-based models for slender soft manipulators, based on which the associated analytical models are deduced. To validate the accuracy of the proposed Cosserat model, the static model of the conical cantilever rod under gravity as a simple example is simulated by using different discretization methods. Results indicate that PLS Cosserat model is comparable to the mechanical deformation behavior of real-world soft manipulator. Finally, three types of parameter identification schemes are established, and the simulation as well as experimental validation demonstrate that using these approaches can identify the model physical parameters with high accuracy.

Based on the proposed PLS Cosserat static model, this thesis presents different closed-loop control architectures for the end-effector position of the soft manipulator: the first is to control the soft manipulator within its individual sub-workspace, and the second is to achieve the global control of the end-effector position. For the second architecture, this thesis proposes two control strategies: one is to calculate (offline) the Jacobian matrix between the task space and the actuator space by unifying the Cosserat and neural network, the other estimates the generalized strain to (online) compute the Jacobian matrix. The performances of the static model-based control approaches have been experimentally validated on the studied soft manipulator.

Furthermore, this thesis proposes a general estimation-based control framework for the soft manipulator based on the PLS Cosserat dynamics. Under this framework, the strain and end-effector position control have been developed. With the knowledge of finite-time estimation of the states, different experiments have been implemented on the studied soft prototype. The simulation and experimental results indicate that the proposed controllers are capable of controlling the soft manipulator to rapidly and accurately track the desired strain and position.

Acknowledgments

First of all, I would like to express my thanks to my supervisor, Gang Zheng for presenting me with the opportunity to work on such an interesting subject, and also for his guidance, help, and sharing his experience with me. I am grateful.

Thanks to the China Scholarship Council (CSC) for the finance of my research.

Many thanks to all members of DEFROST team, and especially LingXiao Xun and Paul Chaillou for the help with the experimental platform. I am grateful for the time spent with them.

Many thanks to the reviewers of my manuscript and jury members.

I would like also to express my gratitude to my family and friends for their care and support.

Nomenclature

Symbol	Unit	Definition
$\dot{}$	—	Derivative with respect to time
$'$	—	Derivative with respect to space
$\widehat{}$	—	Mapping from $\mathbb{R}^6$ to $se(3)$
$\widetilde{}$	—	Mapping from $\mathbb{R}^3$ to $so(3)$
X	m	$\in [0, L] \subset \mathbb{R}$ Abscissa along the soft manipulator
t	s	$\subset \mathbb{R}$ Time
$\boldsymbol{R}$	—	$\in SO(3)$ Orientation matrix
$\boldsymbol{p}$	m	$\in \mathbb{R}^3$ Position vector
$\boldsymbol{g}(X)$	—	$= \begin{pmatrix} \boldsymbol{R} & \boldsymbol{p} \\ \boldsymbol{0}^\top & 1 \end{pmatrix} \in SE(3)$ Homogeneous transformation matrix
$\boldsymbol{K}(X)$	1/m	$\in \mathbb{R}^3$ Angular strain in the body frame
$\boldsymbol{Q}(X)$	—	$\in \mathbb{R}^3$ Linear strain in the body frame
$\boldsymbol{\Omega}(X)$	1/s	$\in \mathbb{R}^3$ Angular velocity in the body frame
$\boldsymbol{V}(X)$	m/s	$\in \mathbb{R}^3$ Linear velocity in the body frame
$\widehat{\boldsymbol{\xi}}(X)$	—	$= \begin{pmatrix} \widetilde{\boldsymbol{K}} & \boldsymbol{Q} \\ \boldsymbol{0}^\top & 0 \end{pmatrix} \in se(3)$: Strain twist matrix
$\boldsymbol{\xi}(X)$	—	$= \left(\boldsymbol{K}^\top, \boldsymbol{Q}^\top\right)^\top \in \mathbb{R}^6$ Strain twist
$\widehat{\boldsymbol{\eta}}(X)$	—	$= \begin{pmatrix} \widetilde{\boldsymbol{\Omega}} & \boldsymbol{V} \\ \boldsymbol{0}^\top & 0 \end{pmatrix} \in se(3)$: Velocity twist matrix
$\boldsymbol{\eta}(X)$	—	$= \left(\boldsymbol{\Omega}^\top, \boldsymbol{V}^\top\right)^\top \in \mathbb{R}^6$ Velocity twist
$\mathrm{ad}_{\boldsymbol{\xi}(X)}$	—	$= \begin{pmatrix} \widetilde{\boldsymbol{K}} & \boldsymbol{0}_{3\times3} \\ \widetilde{\boldsymbol{Q}} & \widetilde{\boldsymbol{K}} \end{pmatrix} \in \mathbb{R}^{6\times6}$: Adjoint representation of the strain twist
$\mathrm{ad}_{\boldsymbol{\eta}(X)}$	—	$= \begin{pmatrix} \widetilde{\boldsymbol{\Omega}} & \boldsymbol{0}_{3\times3} \\ \widetilde{\boldsymbol{V}} & \widetilde{\boldsymbol{\Omega}} \end{pmatrix} \in \mathbb{R}^{6\times6}$: Adjoint representation of the velocity twist
$\mathrm{d}X$	m	Infinitesimal material element
ρ	kg/m^3	Material density
$R(X)$	m	Cross-sectional radius

Continued on next page

Table 1 – continued from previous page

Symbol	Unit	Definition
$A(X)$	m^2	Cross-sectional area
E	Pa	Young's modulus
ν	—	Poisson ratio
G	Pa	Shear modulus (For the isotropic material, $G = E/(2(1+\nu))$)
μ	$\mathrm{Pa}\cdot\mathrm{s}$	Viscosity modulus
n	—	Total number of DoFs
N	—	Total number of sections divided
n_u	—	Number of cables attached at the end of the soft manipulator
$\boldsymbol{\mathcal{I}}$	m^4	$(X) \in \mathbb{R}^{3\times3}$ second moment of the area tensor in the global frame
$\boldsymbol{\mathcal{J}}(X)$	m^4	$= \begin{pmatrix} J_x & 0 & 0 \\ 0 & J_y & 0 \\ 0 & 0 & J_z \end{pmatrix} \in \mathbb{R}^{3\times3}$ ($J_y(X)$ and $J_z(X)$ about the y-axis and z-axis, the polar area moment $J_x(X)$ about the x-axis. For a circular cross-section, $J_x = J_y + J_z$, and $J_y = J_z = \pi R^4/4$) : Second moment of the area tensor in the body frame
$\mathbf{I}_3$	—	3×3 identity matrix
$\boldsymbol{\mathcal{M}}(X)$	—	$= \begin{pmatrix} \rho\boldsymbol{\mathcal{J}}(X) & 0 \\ 0 & \rho\mathbf{I}_3 A(X) \end{pmatrix} \in \mathbb{R}^{6\times6}$: Cross-sectional mass matrix
$\boldsymbol{n}$	N	Internal force in the global frame
$\boldsymbol{m}$	$\mathrm{N}\cdot\mathrm{m}$	Internal torque in the global frame
$\overline{\boldsymbol{n}}$	N/m	Distributed force in the global frame
$\overline{\boldsymbol{m}}$	$\mathrm{N}\cdot\mathrm{m/m}$	Distributed torque in the global frame
$\boldsymbol{N}$	N	Internal force in the body frame
$\boldsymbol{M}$	$\mathrm{N}\cdot\mathrm{m}$	Internal torque in the body frame
$\overline{\boldsymbol{N}}$	N/m	Distributed force in the body frame
$\overline{\boldsymbol{M}}$	$\mathrm{N}\cdot\mathrm{m/m}$	Distributed torque in the body frame
$\boldsymbol{K}_{tb}(X)$	$\mathrm{N}\cdot\mathrm{m}^2$	$= \begin{pmatrix} G & 0 & 0 \\ 0 & E & 0 \\ 0 & 0 & E \end{pmatrix} \boldsymbol{\mathcal{J}}(X) \in \mathbb{R}^{3\times3}$: Stiffness matrix of the cross section at X for torsion and bending
$\boldsymbol{K}_{es}(X)$	$\mathrm{N}\cdot\mathrm{m}^2$	$= \begin{pmatrix} E & 0 & 0 \\ 0 & G & 0 \\ 0 & 0 & G \end{pmatrix} A(X) \in \mathbb{R}^{3\times3}$: Stiffness matrix of the cross section at X for elongation and shear
$\boldsymbol{D}_{tb}(X)$	$\mathrm{N}\cdot\mathrm{m}^2\cdot\mathrm{s}$	$= \begin{pmatrix} \mu & 0 & 0 \\ 0 & 3\mu & 0 \\ 0 & 0 & 3\mu \end{pmatrix} \boldsymbol{\mathcal{J}}(X) \in \mathbb{R}^{3\times3}$: Damping matrix of the cross section at X for torsion and bending
$\boldsymbol{D}_{es}(X)$	$\mathrm{N}\cdot\mathrm{s}$	$= \begin{pmatrix} 3\mu & 0 & 0 \\ 0 & \mu & 0 \\ 0 & 0 & \mu \end{pmatrix} A(X) \in \mathbb{R}^{3\times3}$: Damping matrix of the cross section at X for elongation and shear
$\boldsymbol{g}_r$	—	$\in \mathbb{R}^{4\times4}$: Transformation matrix between the inertial frame and the manipulator base frame

Continued on next page

Table 1 – continued from previous page

Symbol	Unit	Definition
$\boldsymbol{\mathcal{G}}$	—	$\in \mathbb{R}^6$: Gravitational acceleration twist
$\boldsymbol{q}(t)$	—	$= \left[\overline{\boldsymbol{\xi}}_0^\top \ \overline{\boldsymbol{\xi}}_1^\top \cdots \overline{\boldsymbol{\xi}}_N^\top\right]^\top \in \mathbb{R}^{6(N+1)}$: Generalized strain vector
$\boldsymbol{J}(\boldsymbol{q}, X)$	—	$\in \mathbb{R}^{6\times 6(N+1)}$ Body Jacobian matrix
$\dot{\boldsymbol{J}}(\boldsymbol{q}, \dot{\boldsymbol{q}}, X)$	—	$\in \mathbb{R}^{6\times 6(N+1)}$ Derivative of body Jacobian matrix w.r.t. time t
$\boldsymbol{\mathcal{P}}$	—	$= \left[\mathbf{I}_{6N\times 6N}^\top \ \mathbf{0}_{6\times 6N}^\top\right]^\top \in \mathbb{R}^{6(N+1)\times 6N}$: Selection matrix of the nodal strain twists for the purpose of making the increment of tip strain twist $\delta\overline{\boldsymbol{\xi}}_N$ equal to $\mathbf{0}$
$\boldsymbol{M}(\boldsymbol{q})$	—	$\in \mathbb{R}^{6N\times 6(N+1)}$ Mass matrix
$\boldsymbol{K}(\boldsymbol{q})$	—	$\in \mathbb{R}^{6N\times 6(N+1)}$ Stiffness matrix
$\boldsymbol{D}(\boldsymbol{q})$	—	$\in \mathbb{R}^{6N\times 6(N+1)}$ Viscosity matrix
$\boldsymbol{H}(\boldsymbol{q})$	—	$\in \mathbb{R}^{6N\times n_u}$ Actuation matrix
$\boldsymbol{G}(\boldsymbol{q})$	—	$\in \mathbb{R}^{6N\times 6}$ Gravitational matrix
$\underline{\boldsymbol{M}}(\boldsymbol{q})$	—	$\in \mathbb{R}^{6(N+1)\times 6(N+1)}$ Generalized mass matrix
$\underline{\boldsymbol{C}}(\boldsymbol{q}, \dot{\boldsymbol{q}})$	—	$\in \mathbb{R}^{6(N+1)\times 6(N+1)}$ Generalized Coriolis matrix
$\underline{\boldsymbol{K}}$	—	$\in \mathbb{R}^{6(N+1)\times 6(N+1)}$ Generalized stiffness matrix
$\underline{\boldsymbol{H}}$	—	$\in \mathbb{R}^{6(N+1)\times n_u}$ Generalized actuation matrix
$\underline{\boldsymbol{G}}(\boldsymbol{q})$	—	$\in \mathbb{R}^{6(N+1)}$ Generalized gravitational matrix
$\boldsymbol{T}$	N	$\in \mathbb{R}^{n_u}$: Cables' tension vector
$\delta\boldsymbol{q}_a$	—	Increment of generalized strain vector with respect to prior configuration except the strain twist of the end cross section
$\overline{\boldsymbol{\mathcal{P}}}$	—	$\in \mathbb{R}^{n_i(N+1)\times n_i N}$ Reduced generalized selection matrix.
$\boldsymbol{\xi}_{ia}^*$	—	$\in \mathbb{R}^{n_i}$: Vector field of the free strains of any interpolation node i allowed by the rod kinematics
$\boldsymbol{\xi}_{ic}^*$	—	$\in \mathbb{R}^{(6-n_i)}$: Vector field of the constrained strains of any interpolation node i
$\mathbf{B}_a,\ \mathbf{B}_c$	—	Complementary selection matrix of 1 and 0
n_i	—	Number of DoFs of the interpolation nodes allowed by the rod kinematics
$\boldsymbol{\lambda}(X)$	—	$\in \mathbb{R}^{(6-n_i)}$: Constrained wrench in charge of imposing the internal constraints for prohibited strains
$\otimes$	—	Kronecker tensor product
$\overline{\mathbf{B}}_a$	—	$= \mathbf{I}_{(N+1)\times(N+1)} \otimes \mathbf{B}_a$ Extended matrix of $\mathbf{B}_a$
$\overline{\boldsymbol{J}}(\overline{\boldsymbol{q}}, X)$	—	$\in \mathbb{R}^{6\times n_i(N+1)}$: Reduced body Jacobian matrix
$\|\cdot\|$	—	Euclidean norm of a vector or matrix
$(\cdot)^\vee$	—	Mapping from a matrix to a vector
$\mathcal{L}$	—	Lagrangian function
$\boldsymbol{d}_i(X)$	m	Local distance between the midline of soft rod and the cables
$\mathbf{t}_{ci}(X, t)$	—	Unit vector tangent to the cable path
$\boldsymbol{\mathcal{T}}$	—	$= \boldsymbol{\mathcal{T}}_1 \times \boldsymbol{\mathcal{T}}_2 \times \cdots \times \boldsymbol{\mathcal{T}}_{n_u}$ with $\boldsymbol{\mathcal{T}}_i = \left[T_{\min}^i, T_{\max}^i\right]$ being the minimal and maximal tension bounds of the ith cable for $i = 1, \cdots, n_u$
$\mathcal{W}_{_E}$	—	Workspace of the soft manipulator
$\boldsymbol{\Gamma}(\boldsymbol{q})$	—	$\in \mathbb{R}^{3\times n_u}$: Jacobian-based relation matrix between the actuator space (cables' tension) and the task space (end-effector position)
$\boldsymbol{\mu}$	—	$= \begin{bmatrix}\mu_1 & \mu_2 & \cdots & \mu_{2n_u}\end{bmatrix}$: Barrier parameter vector

Continued on next page

Table 1 – continued from previous page

Symbol	Unit	Definition
$\boldsymbol{c}_l,\ b_l$	—	Center and width of the Gaussian kernel function of the l^{th} neuron

Table 1: Acronyms and glossary for the PLS Cosserat modeling and model-based controller design.

Chapter 1

Introduction

1.1 Motivation

1.1.1 Classification of Robotics

During the past few decades, robotics has been a rapidly growing field of engineering, mainly for safety-critical applications such as human assistance [1]. Specifically, robotics is a multi-disciplinary field that aims at the design, modeling, control, and applications of robots.

The most common robots are kinematically non-redundant, which have been widely used to repeatedly carry out a prescribed task in a well-defined environment with high accuracy. When a robot includes more degrees of freedom (DoFs) that are needed to effectuate a task (e.g., an arm with 7 DoFs performs a 6-DoF motion), it is called redundant [2]. For hyper-redundant robots, they are consisted of a large number of DoFs [3], and able to provide high flexibility in the unstructured environments. When the number of DoFs is even larger, the robot becomes a continuum robot which is the result of the evolution of robot design from the discrete mechanism made up of a series of rigid members to the mechanism with elastic structures capable of continuous deformation along their entire body [4]. Since an elastic structure is characterized by the distributed deformation and inherent compliance, the term soft robot is used to define the compliant robots with theoretically infinite number of DoFs [5]. It should be pointed out that continuum robots are capable of continuous deformations, but not all continuum robots are necessarily soft. A basic description on classification of robots based on the DoFs is shown in Fig. 1.1.

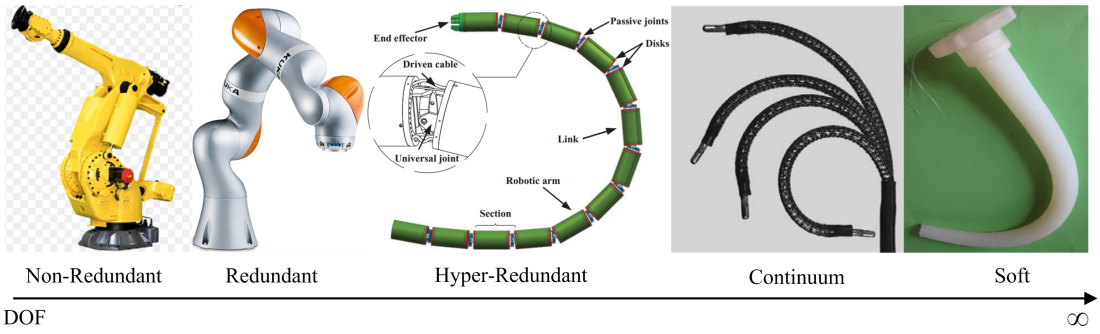


Figure 1.1: Classification of robots based on DoFs [6–10].

On the basis of the compliance of their constitutive materials, robots can be categorized as rigid or soft. Rigid robots are a class of robots whose motion accuracy is not reduced by the vibration and deformation of the structure owing to their stiff material and high-torque joints, which have been widely applied in the constrained environment and proved to increase productivity [11, 12]. Nevertheless, with the advent of new and more complex applications, conventional rigid robots suffer from many disadvantages in certain applications (especially in medical application) such as their high stiffness, limited DoFs and so on [13]. In addition, rigid robots have been regarded less practical when performing tasks in dynamic environments, and lack of flexibility as well as absorption of energy that make the interaction between machines and humans quite dangerous [14]. Despite their incredibly powerful and precise potential, rigid robots tend to be highly specialized and rarely exhibit the rich versatility of natural organisms.

In order to interact with the environment in a more safe and delicate way, there is a need for robots being flexible. Compared to rigid robots, the unprecedented properties of deformability, adaptation, sensitivity and agility enable the flexible robots to better achieve this goal. This motivates many researchers to seek a novel robotic design which is flexible and safe to handle new applications, leading to the rise of soft robotics.

1.1.2 Soft Robotics

Soft robotics, a sub-class of continuum robots [15], is an emergent research topic. In a general sense, soft robots are defined as robots which create motion by deforming their structures. One of the key ideas in designing soft robots is to employ deformable materials to provide safe contact or increase their accessibility, and thus soft robots can be roughly classified into two categories: one refers to the rigid body with soft actuators, and the other is for the deformable body owing to the flexibility of the material [3]. The first category can be regarded as an extension of conventional rigid robots, while the second one is completely new and more attractive since it provides robots with flexibility, for instance, to adapt their shapes to the tasks and their environments. All in all, this type of robots can easily achieve compliant and safe tasks due to its soft property [16].

Biological system in nature provides much inspiration of soft structures which are able to bend, extend, shear and twist. This involves elephant trunk, octopus, snail feet, mammalian tongue, inchworm, colonial anemone, etc. All of these soft structures change their orientation to reinforce their body parts for the purpose of interacting with the environments in any possible kinematic configuration, therefore they can be used in confined spaces [17]. Many researchers are motivated to invent bionic robots by using the morphology of these soft structures. Some have completely soft bodies, like caterpillar and octopus arm, while exhibiting complex behaviors. Some have hard bones, covered with a membrane or skin that then holds them together, such as birds and fish [18]. Clearly, the soft structures are critical to their unparalleled agility and maneuverability.

Many soft robots (see Fig. 1.2) have been designed during the last few years. For example, the design, fabrication, and characterization of multi-material bio-inspired soft octopus robot has been proposed by the inspiration of the simple yet surprising morphology of the octopus [19, 20]. A new type of elephant's trunk robot with very few driving constraints is introduced, which can achieve stable grasping of objects of different shapes and sizes [21]. Inspired by neuromechanical studies of crawling, the design and control of a novel soft robotic platform (Softworms) are described in [22]. An anthropomorphic robotic hand based on soft robotics is presented in [23], which significantly lowers the threshold for the field of grasping and manipulation research. Regarding the biomechanics of wing morphing in birds, the soft bio-hybrid morphing wings have been designed in [24]. The soft-bodied robotic fish is able to swim

along 3D trajectories with autonomous buoyancy control to observe marine life detailed in [25]. The robot Bat Bot, presented in [26], is a fully self-contained, autonomous flying robot that mimics such morphological characteristics of bat wings and uses the highly stretchable silicone-based membrane wings to best match the morphological characteristics of bat flight. As inspired by the cylindrical shape and the contraction motion of the ascidian, [27] introduces a soft robot that resembles this water animal, and it can crawl, tumble, and pick-to-place objects. The soft robotic fingers with embedded bones is proposed to improve the performance of a puppetry robot with haptic feedback in [28]. By combining the benefits of both the electroadhesive and soft pneumatic grippers, the PneuEA gripper presented in [29] is capable of picking not only the flexible materials such as porous cloth, but also fragile objects like light bulbs.

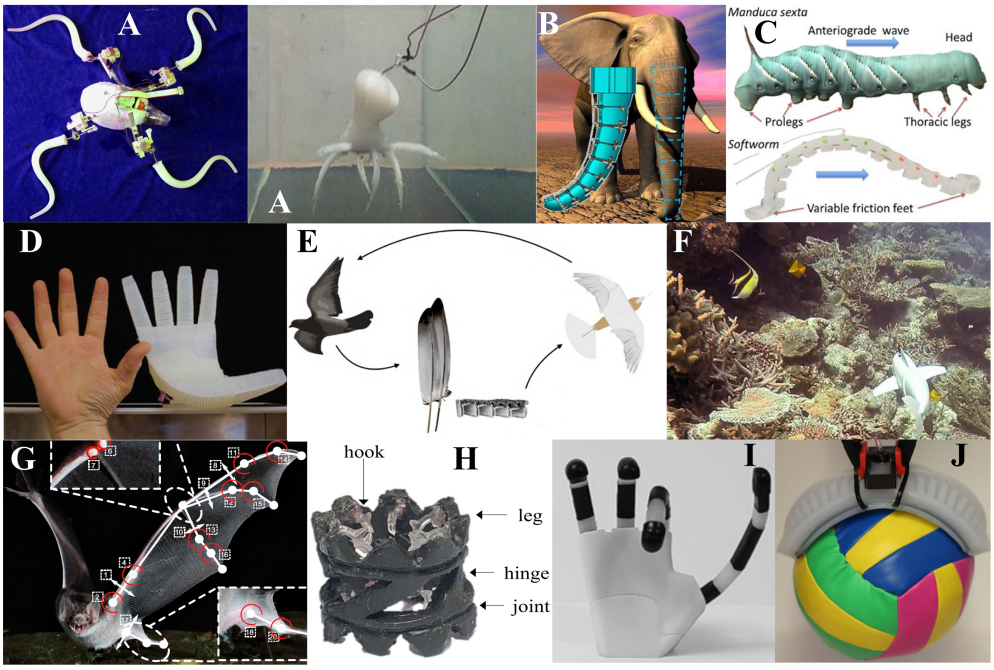


Figure 1.2: Inspirations from nature for soft robots: A- octopus robot [19, 20]. B- elephant trunk arm [21]. C- softworm robot [22]. D- bio-inspired hand robot [23]. E- soft bio-hybrid aerial robots [24]. F- robotic fish [25, 30]. G- Bat Bot [26]. H- ascidian-inspired soft robot [27]. I- bone-inspired soft robot fingers [28]. J- PneuEA gripper [29].

The deformability of the soft material offers flexibility and compliance, which facilitates safe human–robot interaction in comparison to the rigid robotics. There is no doubt that such kind of robots are promising, especially in the autonomous intelligent robotic field where the robots require to operate in cluttered environments. Moreover, these robots can obtain dexterous motion in confined spaces by applying their infinite DoFs, which enables them to be used for the new industrial fields.

Despite the advantages mentioned above, the mechanics of soft robots is complicated due to geometric nonlinearities and continuous deformations, leading to additional difficulties in getting the relatively accurate models in the analytical form. Therefore, it is essential to develop the appropriate modeling approaches for soft robots to fully realize their potential benefits, such as computationally inexpensive, sufficiently accurate model, fast and precise control.

1.2 Scientific Challenges

As an academic research field, soft robotics is highly interdisciplinary. Although many design methods have been proposed, the accuracy and efficiency of soft robotics remains limited by the difficulties of actuation, modeling and control of the deformable system. Consequently, there exist some significant challenges that require further scientific exploration with regard to actuation, modeling and control.

One of the main challenges in the field of soft robotics is the development of effective actuation systems. In traditional robots, actuation is often achieved through the use of rigid motors, gears, and joints. However, these systems are not well-suited to soft robots, as they can cause damage to the soft and delicate materials that are used in their construction. The actuation of any soft robot plays a key role in altering its position and orientation in its workspace to reach the desired objective, which leads to applications including grasping, climbing or moving in a specific trajectory.

Theoretically, the kinematics and dynamics of a soft body can only be captured and reconstructed with infinite number of sensors that correspond to infinite number of system states, which produces highly coupled nonlinear system and incorporates other unexpected complexities. Although many researchers have been trying to develop the mathematical modeling methods capable of obtaining the kinematic and dynamic models of the soft robots, the development of the accurate, general, and yet simple modeling tools for the dynamics of such high-dimensional systems is still a challenging task.

In terms of the control of the soft robots, it is a nontrivial task since elastic deformation of soft manipulators results in almost infinite-DoF motions (bending, extension, torsion, shear, etc), and the material properties exhibit nonlinear characteristics that restrict accurate control [4]. Therefore, soft robots with deformable body also present a great challenge for controller design, particularly with regard to model-based control.

1.3 State of the Art

The accurate description of the morphology of the soft structures has prompted the need for specialized modeling methods. Some significant advancements have been made in designing robots that mimic biologically inspired structures, and how they interact with the environments. However, it is still a difficult problem when considering the modeling and control of such soft robots. Therefore, there is an urgent need to develop efficient modeling methods and model-based feedback controllers.

This thesis is concerned with the development of the discrete Cosserat modeling method and the static as well as dynamic model-based control design. Accordingly, an exhaustive state of the art aiming at the modeling approaches and control design of the soft manipulator will be presented.

1.3.1 Mathematical Modeling of Soft Robots

The soft robots are consisted of infinite number of DoFs and characterized by the complex mechanics due to their natural compliance, which makes their kinematics and dynamics modeling highly nonlinear and more complicated. In the following, the kinematic and dynamic modeling methods available for soft robotics have been summarized (see Fig. 1.3).

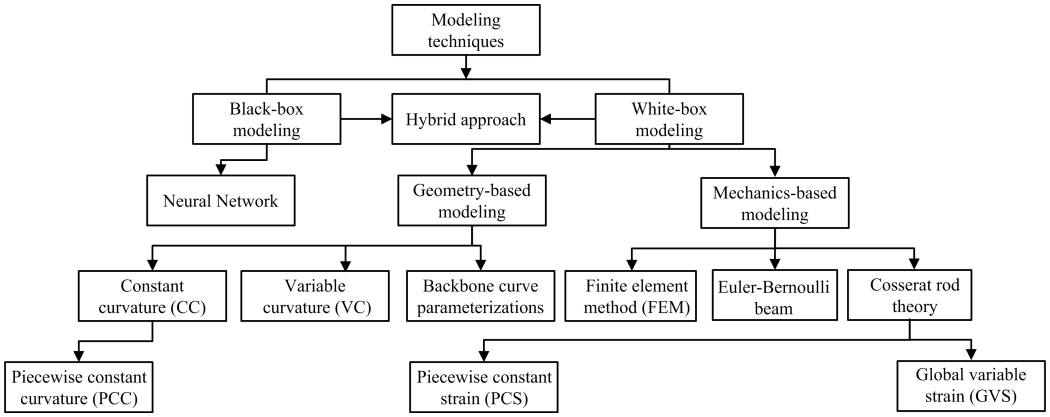


Figure 1.3: Summary of modeling approaches for soft robotics.

In general, modeling approaches can be classified into three categories: the black-box models, white-box models, and hybrid approaches. The black-box models such as modeling of soft robots based on different neural network architectures [31–33] show remarkable performance. However, the training phase of neural network needs to collect enough input-output data sets in order to get precisely approximated model, which is obviously time-consuming.

One of the white-box modeling approaches towards kinematics of the soft robotics consists of geometry-based models, which mainly includes constant curvature (CC) [34], variable curvature (VC) [35], and backbone curve parameterization. The CC approximation has been validated experimentally in many soft continuum robots [36, 37]. For multi-section soft continuum manipulators, each constant-curvature-assumed section can be connected together to generate the piecewise constant curvature (PCC) model [38], which however is valid only for specific design and actuation due to the limitation of CC assumption [39]. On the contrary, VC kinematic frameworks combined with elastic theories are more desirable for slender objects when extreme loading conditions exist [40]. Using the parameterization of backbone curves, the kinematic model can be deduced for a hyper-redundant robot [41, 42]. In [43], the kinematic model of the soft robot is obtained by employing the geometric information. Nevertheless, geometry-based models may not accurately capture the physical behavior of objects under complex loads and boundary conditions, as they may not consider the effects of forces, deformations, and other mechanical properties.

As another promising approach, the mechanics-based modeling techniques mainly including the finite element method (FEM) and beam theory have been widely used for mechanical modeling of soft robots. In general, the soft body in FEM is discretized into finer mesh elements with nodes (treated as independent particles), and the motion of a soft robot is then described via those particles [44]. In soft robotics, the beam theory provides a simplified mathematical framework for the mechanical behaviors of soft robots that can be approximated as slender, flexible beams. [45] developed a quasi-static bending model from a geometrically exact Euler–Bernoulli formulation that generalized a sequence of soft arm designs to predict the result of design changes. Timoshenko beam model allows for the effect of shear deformation, which has been investigated and applied in soft robot modeling [46]. The Cosserat beam, as a powerful alternative to the 3D FEM, assumes that all particles move in the same manner as rigid body, and can provide a systematic modeling framework which is the necessary condition for developing the structural designs and control architectures of these soft robots [47–49]. However, the modeling technique via the Cosserat beam theory is governed by the nonlinear partial

differential equations (PDEs), which results in complicated models and poses difficulties while designing and practically implementing closed-loop feedback control strategies. Despite this, many researchers have developed the different Cosserat models which can capture the dynamics of the robot with large deformations. In [50], a Cosserat geometrically exact dynamic model of a soft manipulator driven by cables was developed. The dissertation about the problems of statics, dynamics, and stability for continuum robots with slender elastic rod on the basis of Cosserat was exhaustively presented in [51]. Based on the work of [51], a new implicit dynamics framework was put forward for solving the PDEs of Cosserat-based model applied to soft continuum robots, aiming at addressing the issue of computational difficulties [52]. In [53], the authors presented the piecewise constant strain (PCS) model for multi-section soft manipulator dynamics. To reduce the dimension of the deduced system, a dynamic Cosserat modeling approach via the strain nonlinear parameterization (i.e., global variable strain (GVS)) was proposed in [54].

Apart from the above modeling techniques, the hybrid methods by unifying learning-based and model-based approaches have also been applied to establish the kinematic and dynamic models of soft robots. For example, [55] proposed the FEM-based training of artificial neural networks for modular soft robots, which enabled learning of kinematic models for simulation and control purposes. In [56], the authors introduced the hybrid models that consisted of both an analytical model and a learned error model. This hybrid approach combined the strengths of both modeling techniques to significantly improve modeling accuracy.

In the literature, the most widely used approaches are FEM [44], PCC [9], and PCS Cosserat method [53]. In the following, the advantages and disadvantages of these discrete methods together with GVS are explored in detail, and then their deformation behaviors are compared.

1.3.1.1 Finite Element Method

FEM-based method which is formulated as ways of approximate solutions of PDEs has been used to model physical behavior of soft robots [40, 57]. In FEM of the deformable body, the deformable domain of the structure is discretized into smaller finite elements using a specific mesh geometry, consisting of finite DoFs, as shown in Fig. 1.4, and then the behavior of the deformable domain is interpolated by measuring variation of the associated nodal displacements.

In engineering, FEM is widely used and regarded as a reliable tool to verify the modeling result, since it discretizes the space in a very generic manner without introducing restrictive assumptions. By selecting finer mesh, FEM can be regarded as one of the most accurate modeling methods comparing to others. To put it another way, FEM is the powerful tool that is rather generic, allowing the modeling and control of a large number of soft robotic systems, including in design, external forces, actuation mechanisms [39]. To solve the forward and inverse kinematic problem, the discrete-time kinematics based on real-time FEM was deduced [58, 59]. In [60], the FEM was used to get a dynamic model of the soft robot, just to cite a few.

Nevertheless, to obtain good modeling precision, FEM demands that the number of nodes should tend to infinity, which negatively increases the dimension of the system and leads to a higher computation complexity. Moreover, FEM models for large-deformation 3D nonlinear elasticity always entail unnecessary computational expense when modeling long, slender arms since general deformations of the cross sections are included.

Such system analysis and controller design using full order models of infinite dimensions are difficult to achieve, and all states of the FEM model are not possible to be measurable when performing the closed-loop control. To overcome these drawbacks, model-order reduction techniques which project the large dimensional system generated by FEM to a smaller subspace have been developed [61–64], aiming at decreasing the number of DoFs and consequently improving the computational efficiency. However, this improvement in turn may decrease the

modeling precision.

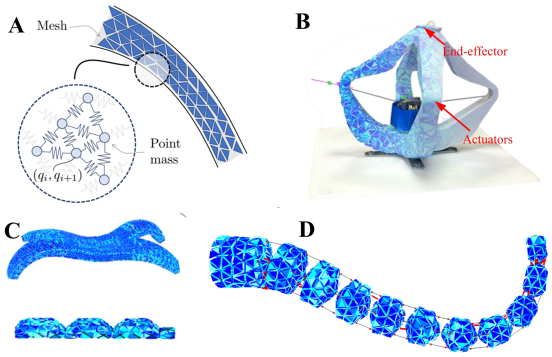


Figure 1.4: FEM modeling approach. A- Geometric description of FEM model [65]. B- FEM model of the parallel robot [44]. C- FEM model of multi-gait soft robot [66]. D- Finite element meshes of soft trunk-like arm [62].

1.3.1.2 Piecewise Constant Curvature

The PCC approach was initially put forward for kinematics, and later extended to the dynamics of soft robots. It describes a soft robot with a finite number of arcs parameterized by three quantities (curvature κ, arc length l, and bending plane ϕ), resulting in the reduced-order and relatively simple mathematical models, as shown in Fig. 1.5. As for the principle of PCC, all strains are ignored except curvature, which itself is assumed to be piecewise constant and variable in time. This method has proven to be a very useful technique with a wide range of applications. Examples include design [67], kinematics and dynamics modeling [68], kinematic control [38, 69, 70], feedforward dynamic control [71], feedback dynamic control [72, 73].

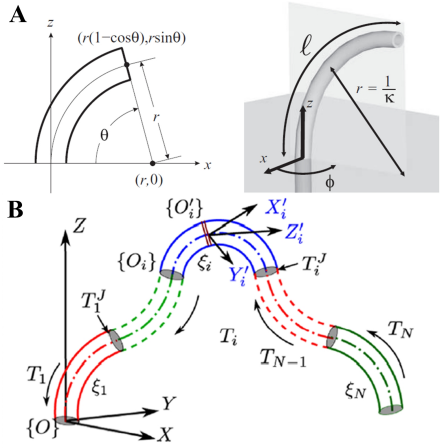


Figure 1.5: PCC modeling technique. A- PCC parameterization [9]. B- Schematic of the PCC model for a multisection continuum robot [68].

Despite this success, when the conditions of external loads or contact with the environments are considered, the validity of the CC assumption is relatively conservative and unsuitable for

many applications [74]. Therefore, the CC assumption is valid only for the specific design and actuation manner, and it can not be regarded as generic. Furthermore, it does not allow torsional deformation which is fundamental to cope with non-negligible external loads [53]. These flaws can potentially produce critical behaviors in the real-world scenario.

1.3.1.3 Piecewise Constant Strain

PCS Cosserat modeling technique [53, 75, 76] extends the PCC assumption to more general rod kinematics while providing a dynamic model. It employs a finite set of piecewise constant strains with discontinuities happening at fixed points along the rod (see Fig. 1.6) to model the deformation of the soft robots, and provides advantages to use with not only a few state variables, but also a relatively high modeling precision.

Recently, the PCS Cosserat dynamic models have been increasingly applied to robots such as soft gripper [77] and Fin-Ray finger [78], etc. With regards to the soft robots modeled through the PCS approximation, the number of sections to be divided depends on the designer of the model as a result of application-specific considerations. This technique is able to exactly approximate the continuum formulation when the arm is divided into more sections, yet the dimension of state variables and computational cost will highly increase. Consequently, a trade-off between model order reduction and accuracy should be established.

From practical point of view, the deformation field of any certain section under external forces is obviously not constant along the soft arm [65]. To this end, the authors proposed a novel variable-strain parameterization by the discretization of the continuous Cosserat rod model onto a finite set of strain basis functions in order to generalize the PCS method to the case of discrete non-constant strains (i.e., geometric variable strain) [79]. However, this approach has not been extended to the multi-section dynamic case.

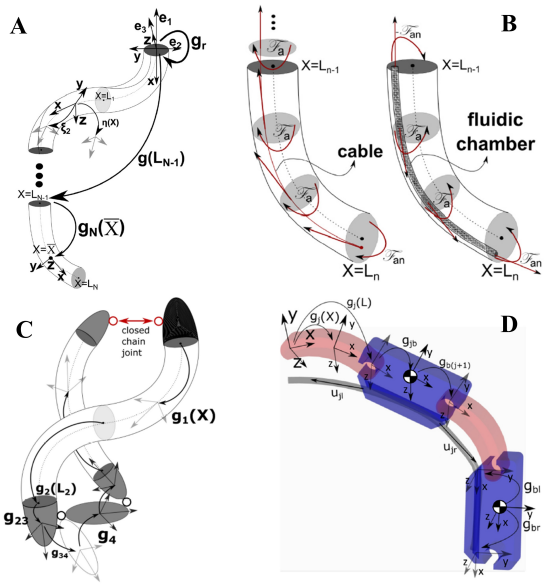


Figure 1.6: PCS Cosserat modeling approach and its applications. A- Depiction of the PCS kinematics [53]. B- Cable and fluidic actuation for PCS model [76]. C- Fin-Ray Finger [78]. D- Modular structure of a gripper [77].

1.3.1.4 Global Variable Strain

The global variable strain (GVS) method [54] proposed to globally approximate the strain field as $\boldsymbol{\xi}(X,t) = \boldsymbol{\Phi}(X)\boldsymbol{q}(t)$ where $\boldsymbol{\Phi}(X) = (\Phi_1, \Phi_2, \cdots, \Phi_n)$ defines n basis functions to parameterize the strain space, which is numerically simple with a good accuracy and less DoFs, and has been proved by the simulation of a flying rod, as shown in Fig. 1.7. However, this technique leads to computational complexities owing to high-order basis functions required when modeling quite complicated deformation such as global buckling behaviors.

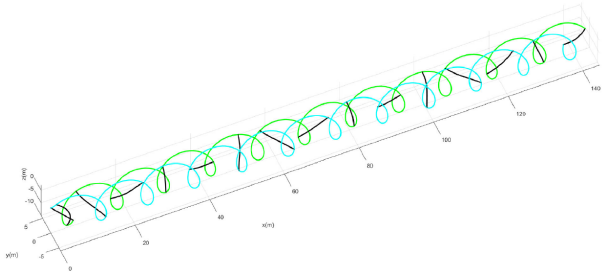


Figure 1.7: GVS method [54].

1.3.1.5 Comparison of Aforementioned Discrete Modeling Approaches Together with GVS

FEM is one of the preferred numerical tools to model the kinematics and dynamics of a soft robot. However, the model of soft robot generated by FEM is of very large dimension, which leads to difficulty in designing model-based controllers. Moreover, it is difficult to obtain the constitutive laws of materials for soft robots with complex shapes or special materials [32].

The PCC modeling approach provides an effective and simple method for the soft robotic modeling, which greatly reduces the number of variables needed and has been widely adopted in the soft robotics community. However, the adopted CC assumption are not always valid, especially when the robot is subject to out-of-plane external loads. Moreover, the material properties of the robot are not considered which is significantly important for the dynamic modeling of soft robots.

As an extension of PCC, the PCS considers both the material properties of a robot and geometric nonlinearity. Clearly, this assumption requires fine spatial discretization which yields relatively high dimensional system. Such a method can provide enough precision and work well for numerical simulation which might take time. Nevertheless, PCS will be a big issue when designing model-based controller.

One potential advantage of using GVS over PCS is that the dimension of the resulting dynamic model is quite low, and it also can provide comparable precision if the number of basis function is high enough. Yet, the choice of the number of basis function is highly dependent on external disturbances. For example, the external disturbance such as concentrated loads will lead to local strain mutation, and the GVS approximation method will be difficult to guarantee the local fitting precision. Moreover, all the kinematics are no more analytically integrable in contrast to PCS, but numerically reconstructed with a quaternion-based integrator. In other words, the deduced model is not anymore analytical, and will not be very friendly for control design.

1.3.1.6 Model Parameter Identification

Once the model has been deduced, it is important and necessary to identify the physical parameters (such as stiffness, damping, and mass, etc.) of the model conveniently and accurately. The parameter identification is a crucial step in validating the accuracy and reliability of the model. By comparing the model's predictions with experimental data obtained from the actual soft robot, parameter identification helps to validate the model's precision and ensure that it is representative of the real-world behaviors of the soft robot.

Numerous model parameter identification techniques have been proposed by researchers. Several major contributions for dynamic model identification and its applications in the robotics control have been reported in [80, 81]. These studies assumed that the kinematic models were accurate, however, some typical investigations presented uncertainties of kinematic models [82, 83]. The parameters identification of static model is more obtainable than the dynamic model in practical applications since it just needs information of joint position rather than joint velocity and acceleration. [84] established a framework for the special actuator's modeling using Cosserat rods along with experimental data-driven estimates of the actuation and stiffness parameters. In [85], a material characterization approach for identifying the parameters for material models via the target application was proposed to improve their use in the modeling of soft continuum robots. For the work of [86], the authors introduced a procedure of the static model identification towards general serial articulated manipulator and presented an application of the identified static model to validate the proposed methods. In the literature, most of existing parameter identification methods based on Cosserat rod theory were proposed for specific robots, and there is a lack of general approach to identify physical parameters for slender soft robots based on the discrete Cosserat models.

1.3.2 Control of Soft Robots

Due to their inherent compliance and high flexibility properties, soft robots present significant challenges to develop high-performance control techniques compared to rigid counterparts. Generally, control design for soft robots can be achieved using model-free and model-based strategies, which will be described in what follows.

1.3.2.1 Model-Free Control

Soft robots present a set of internal nonlinearities and external uncertain disturbances, which makes it difficult to establish an analytical model that can be used to develop the control schemes. In addition, limited by the accuracy of the kinematic and dynamic models (e.g., uncertainties in kinematic parameters, structural deformation, and external loads), the performances (e.g., accuracy and robustness) of the model-based control methods are not always satisfactory [5]. Accordingly, the model-free control methodologies are proposed to control soft robots using machine learning techniques or empirical approaches, which has become a suitable tool to well control the soft robots with the complexities of nonlinear behaviors.

In [87], the authors proposed an efficient exploratory learning method of inverse kinematics on an elephant trunk robot, and provided the first functioning control concept for the robot platform. This learning approach together with integration of feedback control allows accurate and reliable end-effector positioning in the practical operation. To minimize variation in the actuator space, [88, 89] attempted to learn the differential inverse kinematic model through local mappings, which allowed multiple solutions to the global inverse kinematic problem and works even if some actuators are nonfunctional after the learning process. Another method was presented in [90] to estimate the robot's Jacobian and accomplish path tracking for a continuum

robot by adaptive Kalman filter approximation. In [33], the authors learned a model of a soft robot's forward kinematics and carried out gradient-based optimization to find optimal control inputs, which however limits the type of network used for forward kinematics learning and requires more time to compute the Jacobian for determining inverse kinematics solutions. For the work of [91], a method was proposed to train both the forward kinematic model and its Jacobian as two neural networks in order to solve the inverse kinematics in a fast speed. This technique can effectively solve the inverse kinematics and realize control purpose of kinematic tasks, but there is one major drawback that the static model will introduce large errors in the tracking tasks when performing the dynamic behavior of the soft robot. [92] proposed a model-free control method via reinforcement learning to control a multi-segment soft manipulator in 2D plane. In [93], a data-based control framework was introduced to solve the soft robot underwater locomotion problem using deep reinforcement learning.

Model-less control is an approach that learns the manipulator's Jacobian and adapts to constraints in the environment in a safe manner. For example, [94] presented a model-less method for controlling a tendon-driven continuum manipulator based on empirical estimates of the robot Jacobian, and employed optimal control in closed-loop task space feedback to obtain the control inputs. However, during interaction with the environment using the end-effector, the presence of tip-constraints can lead to ill-conditioned Jacobian estimates. To address this issue, [95] defined a hybrid control approach for simultaneously controlling end-effector position and force while maintaining the minimalist approach to model-less control. [96] proposed a closed-loop model-less feedback tracking control scheme for soft manipulators with Jacobian matrix adaption. This framework is independent of a specific kinematic model and has wide applicability for different kinds of arms.

Different learning-based strategies have also been used to develop controllers for dynamic control of soft robots. In [97], the authors presented a machine learning based approach for development of dynamic models of the soft robotic manipulator and a trajectory optimization method for open loop predictive control of the manipulator in task space. Due to the computational cost of analytical models, [98] used a type of recurrent neural network to learn the forward dynamic model, and then open loop control policies were sampled on the under-actuated pneumatically-driven soft manipulator by using the learned model and trajectory optimization algorithm. Based on artificial neural network (ANN), two types of robust controllers were proposed with complete mathematical proofs, which can always guarantee the exponential convergence of the end-effector of soft robot to a desired reference, however, training phase to collect data is time-consuming [32]. [99] established the learning nonlinear dynamic models of soft robots for model predictive control with neural networks. Using the developed dynamic model and trajectory-optimization techniques, [100] found the locally-optimal open-loop policies which enable the system to perform dynamic manipulation of grabbing. For the work of [101], the controller depending on the learning of an inverse static model was provided, which was used as a baseline controller for trajectory tracking tasks, and the data collected from this task was then employed to learn an inverse dynamics model using a long short-term memory network.

1.3.2.2 Model-Based Control

The model-free approaches to develop control schemes lie in the easy modeling accuracy and low sensory requirements. Despite their gratifying advancements, collecting a representative and sufficient amount of data is still challenging for most learning-based control approaches for soft robots. In addition, stability analysis and theoretical convergence proofs are difficult to provide for the learning-based controllers. These factors motivates researchers to investigate the model-based controllers. Many researchers have proposed in-depth investigations of various

control techniques based on different modeling methods. One straightforward approach to model and control the soft robots is to obtain the simplified approximate models of the robotic motion. Most works on model-based control still tend to use the aforementioned discrete models.

The PCC models can be seen as an oversimplification of the real soft robots, however, these modeling methods have been proven to be quite feasible with a vast range of control applications. [102] explored a kinematic model of the soft manipulator using the concept of the PCC, and designed an adaptive controller for image-based visual servoing to control the end-effector position. For the elastomer-based manipulator using the PCC assumption, [103] developed a control algorithm that can control the manipulator's configuration along the realizable kinematic curvature trajectories. Under the PCC assumption, the contributors of [73] presented a closed-loop control architecture for the purpose of achieving accurate curvature and bending control. The dynamic model via the PCC was adopted by [104] to realize static posture regulation, but this control strategy is limited in performing the dynamic tasks. In [105], the authors developed the PCC model-based dynamic feedback control of a planar soft robot for trajectory tracking and surface following, enabling a wide range of dynamic tasks. [70] presented an alternative state representation which can address all issues PCC suffers from, and derived the model-based controller. Yet, there are no experiments on real robots to test the use of the proposed parameterization.

Moving to a more common case, FEM has been widely employed in the modeling of soft robots. The strategies for inverse kinematics-based control [44] directly used FEM models. In [59], the authors employed FEM and visual tracking technique to achieve the feedback control for soft robots. [106] used asynchronous FEM and quadratic programming algorithm to obtain the inverse solution that is utilized to control the actuators. Nevertheless, the high-dimensional FEM makes the real-time controllers design tough, and restricts its practical application in feedback control, which motivates the development of FEM model reduction methods well suitable for control design [62]. To design a controller usable in practice with less number of sensors, the FEM model reduction technique was employed in [107] where a state observer is added to control the dynamics of cable-driven soft robots around a stable equilibrium point. Based on this work, the authors proposed reduced order linear parameter varying models via the FEM, and conducted nonlinear controller design for these models [108]. In order to circumvent the drawbacks of FEM, [16] established the link between the simplified model and the FEM, and designed a robust controller to achieve position control of the soft tripod robot. In [109], the authors proposed a novel approach for model-based optimal control of soft robots via order-reduction FEM without loss in modeling accuracy.

Some representative achievements about Cosserat-rod-based controllers have also been made. In [110], the Cosserat rod model was used to design robust controller for continuum robots for the first time. The approach was proposed for performing stiffness control on continuum robots modeled by the Cosserat rod [111]. Although this controller provided good dynamic performance and stability, it did not provide any theoretical convergence proof. [112] presented a coupled Cosserat rod model for parallel continuum robots to realize real-time kinematic control, however, no control experiments were conducted. The authors of [113] coupled the Cosserat kinematic model with integrated sensing and the efficient Jacobian matrix to provide a practically feasible, closed-loop control system for application to the soft continuum manipulators. Typical works on discrete Cosserat rod model-based controllers have also been reported. [98] presented a PCS model-based reinforcement learning algorithm for the global closed-loop predictive control of the soft manipulator. Since controllers employing second-order dynamic models tend to be computationally expensive, [114] proposed to reduce the PCS dynamic model under a first-order approximation assumption, and developed closed-loop controllers by the use of the simplification of the planning and sensor feedback. For the work of [39], the geometric variable strain [79]

approach was used for Jacobian-based inverse kinematic control of general soft manipulator. Nevertheless, this strategy ignored the influence of the external forces and lacked the experimental validation.

The hybrid control method combining model-based and learning-based control approaches has also emerged as a cutting-edge control strategy for soft robots, which can be regarded as an improvement of model-free control while incorporating the strengths of physics models. In [10], a supervised learning method to solve the inverse statics of the cable-actuated soft manipulator with non-constant curvature was introduced. For the work of [115], the authors put forward a hybrid modeling method which combined the first-principles model with machine learning methods to improve overall performance of the nonlinear model predictive controller. Based on FEM and ANN, [116] proposed a transfer learning scheme to minimize the effort of generating real data for neural network training, and achieved the fine control of a real soft pneumatic actuator. In [117], the authors introduced a model-based online learning and adaptive control algorithm for the wearable soft robotic glove, which enabled the soft glove to adapt to diverse hand conditions for reference tracking.

In many existing works on control of soft robots, there exists a big issue related to state estimation since most of controllers are proposed to be state feedback type when designing model-based control for soft robots, which assumes that all states of soft robots need to be known. Nevertheless, for many nonlinear dynamic systems, the state variables of the system for control design are not measurable in practice, which therefore promotes the development of the state estimation approaches. To ensure more precise model-based control, an appropriate state estimation technique will be necessary even though it is inherently challenging due to the infinite dimensionality of the system. One of the most widespread and successfully used techniques to solve the state estimation problem is the Kalman filter (KF) [118]. Besides, different kinds of nonlinear state estimators such as the moving horizon estimation [119] and extended KF [120] have also been proposed for many physical systems with nonlinear dynamics and constrained states. Examples of the state estimation for soft continuum robots include filtering method for estimating the shape and end-effector pose of a snake robot [121], real-time pose estimation approach for a tendon-driven continuum manipulator [122], force and shape estimation in soft robotics [123]. Despite these progresses, to our best knowledge, no state estimation approach via PLS Cosserat has been investigated for the control of the cable-driven soft manipulator.

1.3.2.3 Summary

As for the model-free control methods, there is no need to define parameters in the configuration/joint space and consider the shape of the manipulator. Thus, arbitrarily complex kinematic or dynamic models can be developed based on the sample data. This is probably why model-free methods perform better for systems that are highly nonlinear, non-uniform, affected by gravity, or operating in unstructured environments where modeling are nearly impossible [10,87]. Despite its significant success, the model-free control method still exists some challenges. For instance, this approach requires a significant amount of data or samples to learn effective control policies, leading to time-consuming and impractical implementations in some real-world applications. Additionally, reinforcement learning-based control methods encounter the challenge of striking a balance between exploration (trying new actions to learn) and exploitation (using known good actions). Moreover, it is difficult for most model-free control methods to provide stability analysis and theoretical convergence proofs.

The extensive model-based control strategies that consider the complete kinematics and dynamics have become better and more reliable controllers for the soft robots. The static model-based control lies in the assumption of steady state, rendering soft robotic arms fail

to move quickly, whereas the dynamic model-based controllers take into account the dynamic behaviors of soft manipulators to achieve fast and efficient tracking.

1.4 Contributions of this thesis

1.4.1 Contributions and Organization of the Manuscript

The development of general and accurate modeling tools for soft robotic applications is a challenging task that many researchers have been trying to solve. Despite the great number of proposed solutions, many of the desired objectives are yet to be accomplished, especially for what concerns the applications of the proposed methods in the control design prospective. This is mainly due to the complex and highly nonlinear kinematics and dynamics associated with this type of soft robots. Nowadays, the most adopted models for the modeling and control of slender soft robots include the FEM, PCC, PCS and GVS models, with its consolidated benefits and drawbacks.

In this thesis, we propose a discrete Cosserat modeling method and design the various controllers based on the proposed model, which can be formulated by the following questions: Given the real strain of the slender soft manipulator, how to use the method via the Cosserat rod theory to approximate it more accurately? Given the proposed discrete Cosserat models, how to design the model-based controllers to achieve the strain control or end-effector position control of the slender soft manipulator?

To answer the above two questions, the manuscript has been organized into 6 chapters as follows (see Fig. 1.8):

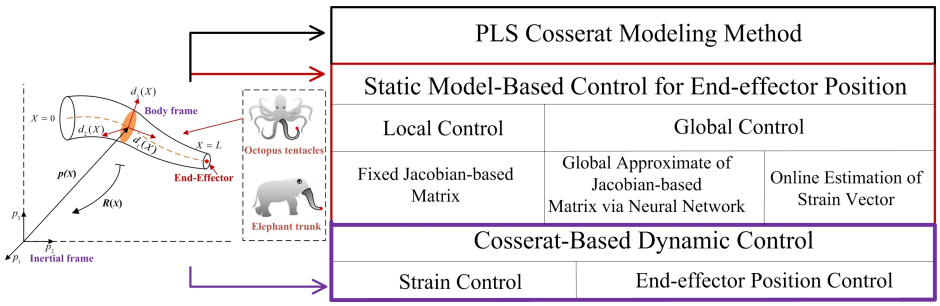


Figure 1.8: Manuscript Organization.

Chapter 2 commences with a complete description of theoretical background for modeling (see Appendix A), followed by briefly reviewing the continuous Cosserat models (i.e., kinematics, differential kinematics, statics and dynamics) in order to introduce the discretization method developed in the next chapter.

Chapter 3 proposes a discrete Cosserat modeling technique (i.e., PLS: piecewise linear strain) to approximate the strong form of the Cosserat models for the soft manipulator. Based on the PLS modeling approach, several parameter identification schemes have been presented to identify the geometric and material parameters of the model, which are verified by simulations as well as experiments on the investigated manipulator, and paves the way for the subsequent model-based controller design.

Next, the PLS Cosserat static model-based control scheme is established in Chapter 4 to achieve the end-effector position control for the slender soft manipulator in its workspace. Due

to the difficulty of the knowledge of the Jacobian matrix between the actuator space and the task space, we address such an issue from two aspects, a local controller and a global one. For the local controller, we use the constant Jacobian matrix to approximate the real Jacobian matrix. To realize the global control objective of the end-effector position, we propose two approaches that consist of an offline learning method to approximate the Jacobian matrix and online estimation of the generalized strain vector. Finally, the feasibility and robustness of these control schemes have been validated on the studied soft manipulator through extensive experimental tests.

Chapter 5 presents the strain and end-effector position control for the slender soft manipulator via the PLS Cosserat dynamics, respectively. The performances of the designed controllers have been validated by a series of simulations and experiments.

Finally, Chapter 6 provides the conclusions of the present thesis and presents the perspectives of future works, especially, the model parameter estimation from offline to online, the frictional contacts between soft manipulator and external environments, and also the orientation and position control of slender soft manipulator via PLS Cosserat.

1.4.2 Publications

The contributions of the present thesis are outlined in the following:

Journal Articles:

1. Haihong Li, Lingxiao Xun, and Gang Zheng, "Piecewise Linear Strain Cosserat Model for Soft Slender Manipulator." IEEE Transactions on Robotics, 2023, 39(3):2342-2359, doi:10.1109/TRO.2023.3236942.
   Summary: In this article, we proposed a discrete modeling technique named piecewise linear strain (PLS) to solve the PDEs of Cosserat-based models, based on which the associated analytical models are deduced.

2. Haihong Li, Lingxiao Xun, Gang Zheng, and Federico Renda. "Discrete Cosserat Static Model-Based Control of Soft Manipulator." IEEE Robotics and Automation Letters, 2023, 8(3): 1739-1746, doi:10.1109/LRA.2023.3243799.
   Summary: In this paper, the robust closed-loop control architecture based on the PLS Cosserat static model for the soft manipulator around its sub-workspace was established. To validate the performance of the proposed model-based control scheme, the point-to-point and time-varying trajectory tracking experiments under external disturbances have been conducted on the soft manipulator actuated by cables.

3. Haihong Li, Lingxiao Xun, Gang Zheng. "Global Control of Soft Manipulator by Unifying Cosserat and Neural Network." IEEE Transactions on Industrial Electronics. Under review.
   Summary: In this journal, we proposed a hybrid control strategy by unifying the PLS Cosserat model and radial basis function neural network to approximate the Jacobian matrix of any end-effector position in the whole workspace, and then designed a global control scheme based on the approximation of Jacobian matrix. The theoretical convergence proof and experimental validation were provided for the designed global controller.

4. Haihong Li, Lingxiao Xun, Gang Zheng, and Frédéric Boyer. "Cosserat-Based Dynamic Control of Soft Manipulator" IEEE Transactions on Robotics, 2023. Under review.
   Summary: In this paper, we proposed a general state estimation-based control framework for the soft manipulator based on the PLS Cosserat dynamic model for the first time.

Then, two types of control, namely strain control and end-effector position control, have been investigated. With the estimation of the system states, different experiments have been implemented on the studied soft prototype driven by the cables to further validate the performances (convergence speed, precision and robustness) of the control approaches.

Chapter 2

Continuous Cosserat Model

2.1 Introduction

Using the Lie group structure of rigid body motions to describe kinematic chains mathematically inspired researchers to construct modern screw theory by resorting to the basic linear algebra and matrix groups. By virtue of the modern screw theory, the modern differential geometry can be employed to handle an extensive range of problems about robots in an elegant and simple way, which has contributed to the development of representative textbooks by Murray et al. [124] for the rigid robots.

For the soft robots, the mathematical modeling is complicated due to the description of the continuously deformable 3D shape when subjected to external loads. However, we can use the classical beam theory, which allows the configuration of the robot to be defined by considering two-dimensional cross sections on a 3D reference curve. When the cross-sectional area is much smaller than the length of the rod (i.e., slender rod), the soft rod can be modeled as the continuous assembly of cross sections moving on a 3D curve according to infinite rigid body transformations that are defined by distribution laws of internal deformations.

The representations for configurations and velocities, as well as the key concepts of rigid body transformation which belongs to the special Euclidean group $SE(3)$, have been provided in Appendix A. Additionally, we have reviewed the standard derivation of the equations of rigid body motion using the Newton-Euler dynamics formulation.

In this chapter, we established the continuous Cosserat models (kinematic, differential kinematic, and static) based on the theoretical modeling basis of a single rigid body. Then, the continuous Cosserat dynamic model is deduced, which paves the way for the subsequent discrete modeling approach.

To make reading equations throughout the thesis easier, all vector and matrix variables will be typed in bold, while scalar quantities will not.

2.2 Kinematic and Differential Kinematic Models

A slender soft robot can be viewed as a Cosserat rod from the perspective of the geometrically exact approach. There have been numerous works on the dynamic theories of Cosserat rods, for example, [125, 126] provided comprehensive derivations of the continuous Cosserat beam theory. In the following, we will illustrate the mathematical models of the continuous Cosserat using geometrical formulation and notations.

The actuation mechanism of any robot plays a crucial role in changing its position and orientation within its workspace to achieve a specific goal, leading to a wide range of applications including simple manipulation such as grasping, or moving along the specified trajectories. For rigid robots, actuation is generally achieved using servo or stepping motors, while the most common type of actuators used for soft robots is the tendon or fluidic actuation. In this thesis, the cable-driven actuation is adopted and studied for the slender soft robot.

In the interest of clarity, we will use $(\cdot)'$ and $\dot{(\cdot)}$ to represent the space and time partial derivatives with respect to X and t, respectively.

2.2.1 Kinematic Model

A Cosserat rod is regarded as a one-dimensional slender continuum deformable body, and each cross section of the rod is considered as an infinitesimal material element whose size and shape cannot change under external forces, as illustrated in the orange part in Fig. 2.1. All variables of the rod can be parameterized by the reference arc length $X \in [0, L] \subset \mathbb{R}$ along the undeformed rod and by the time $t \in \mathbb{R}^+$. For ease of reference, the nomenclature we selected is summarized in Table 1.

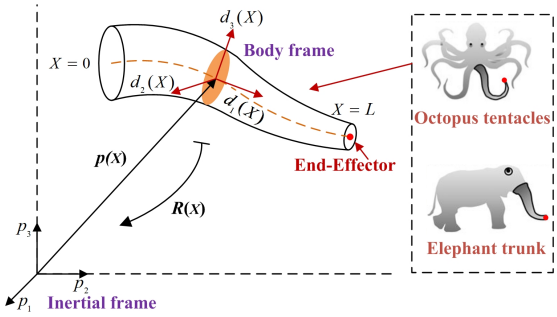


Figure 2.1: Geometric description of a Cosserat rod and its applications.

2.2.1.1 Position field

The position field of any cross section can be represented by a centerline vector $\boldsymbol{p}(X,t) \in \mathbb{R}^3$ and an orthonormal rotation matrix $\boldsymbol{R}(X,t) \in SO(3)$ with respect to the inertial frame at the moment t, as shown in Fig. 2.1. Consequently, the configuration space can be naturally defined as

$$SE(3) = \left\{ \boldsymbol{g}(X,t) = \begin{pmatrix} \boldsymbol{R} & \boldsymbol{p} \\ \boldsymbol{0}^\top & 1 \end{pmatrix} | \boldsymbol{R}(X,t) \in SO(3), \boldsymbol{p}(X,t) \in \mathbb{R}^3 \right\} \tag{2.1}$$

where $\boldsymbol{g}(X,t)$ can be considered as position and orientation of any cross section at X and time t in the body frame with respect to the inertial coordinate frame.

2.2.1.2 Deformation Field

The partial derivative of the configuration curve $\boldsymbol{g}(X)$ with respect to arc length X in the body frame at the moment t is denoted by the strain twist $\widehat{\boldsymbol{\xi}}(X,t)$, that is, $\widehat{\boldsymbol{\xi}}(X,t) := \boldsymbol{g}^{-1}\frac{\partial \boldsymbol{g}}{\partial X} = \boldsymbol{g}^{-1}\boldsymbol{g}'$. As in [127], the symbol $\widehat{\ }$ represents the mapping between the Euclidean space $\mathbb{R}^6$ and tangential space of the Lie group $SE(3)$ (i.e., Lie algebra $se(3)$). The deformation field and its components

are specified as

$$\widehat{\boldsymbol{\xi}}(X,t) = \begin{pmatrix} \widetilde{\boldsymbol{K}} & \boldsymbol{Q} \\ \boldsymbol{0}^\top & 0 \end{pmatrix} \in se(3), \qquad \boldsymbol{\xi}(X,t) = \left(\boldsymbol{K}^\top, \boldsymbol{Q}^\top\right)^\top \in \mathbb{R}^6$$

where $\widehat{\boldsymbol{\xi}}(X,t)$ describes the strain state of the rod, and the symbol $\widetilde{\ }$ is the mapping from $\mathbb{R}^3$ to $so(3)$; $\boldsymbol{K}(X,t) \in \mathbb{R}^3$ represents the bending and torsion states of the rod, and $\boldsymbol{Q}(X,t) \in \mathbb{R}^3$ denotes shear and extension states measuring change rate of position w.r.t. arc length X.

According to the definitions of the strain fields, the differential equation (DE) describing the geometric model $\boldsymbol{g}(X)$ for any cross section along the soft manipulator can be obtained via the Cosserat rod theory, which is given by

$$\boldsymbol{g}' = \boldsymbol{g}\widehat{\boldsymbol{\xi}} \tag{2.2}$$

2.2.2 Differential Kinematic Model

The main purpose of differential kinematics is to derive the velocity and acceleration fields for the robotic mechanisms, and thus the derivation of the mapping between the velocities and the derivatives of the states of the manipulator will be established.

2.2.2.1 Velocity Field

Likewise, another term for time partial derivative of configuration curve $\boldsymbol{g}(X)$ in the body frame at the moment t is represented by the velocity twist $\widehat{\boldsymbol{\eta}}(X,t) \coloneqq \boldsymbol{g}^{-1}\frac{\partial \boldsymbol{g}}{\partial t} = \boldsymbol{g}^{-1}\dot{\boldsymbol{g}}$. As in the above description, the velocity field and its components are expressed as

$$\widehat{\boldsymbol{\eta}}(X,t) = \begin{pmatrix} \widetilde{\boldsymbol{\Omega}} & \boldsymbol{V} \\ \boldsymbol{0}^\top & 0 \end{pmatrix} \in se(3), \qquad \boldsymbol{\eta}(X,t) = \left(\boldsymbol{\Omega}^\top, \boldsymbol{V}^\top\right)^\top \in \mathbb{R}^6$$

where $\widehat{\boldsymbol{\eta}}$ is the velocity state of the rod; $\boldsymbol{\Omega}(X,t) \in \mathbb{R}^3$ and $\boldsymbol{V}(X,t) \in \mathbb{R}^3$ stand for the angular and linear velocity, respectively. Hence, the time derivative of (2.1) yields

$$\dot{\boldsymbol{g}} = \boldsymbol{g}\widehat{\boldsymbol{\eta}} \tag{2.3}$$

2.2.2.2 Compatibility Equations

In continuum mechanics, the compatibility laws of finite strains are formulated for the deformed object without non-physical gaps or overlaps, which translates into the coordination conditions between the strain and velocity of the continuum body.

From (2.2) and (2.3), the space and time derivatives of configuration curve $\boldsymbol{g}(X)$ are respectively used to define the strain and the velocity. Based on the fact that the equality of the mixed partial derivatives $(\dot{\boldsymbol{g}}') = (\dot{\boldsymbol{g}})'$ holds, the compatibility equation between velocity and deformation variables is then formulated as

$$\boldsymbol{\eta}' = \dot{\boldsymbol{\xi}}(X) - \mathrm{ad}_{\boldsymbol{\xi}(X)}\boldsymbol{\eta}(X) \tag{2.4}$$

where ad is the adjoint map defined as

$$\mathrm{ad}_{\boldsymbol{\xi}} = \begin{pmatrix} \widetilde{\boldsymbol{K}} & \boldsymbol{0}_{3\times3} \\ \widetilde{\boldsymbol{Q}} & \widetilde{\boldsymbol{K}} \end{pmatrix} \in \mathbb{R}^{6\times6}, \quad \mathrm{ad}_{\boldsymbol{\eta}} = \begin{pmatrix} \widetilde{\boldsymbol{\Omega}} & \boldsymbol{0}_{3\times3} \\ \widetilde{\boldsymbol{V}} & \widetilde{\boldsymbol{\Omega}} \end{pmatrix} \in \mathbb{R}^{6\times6}$$

2.2.2.3 Acceleration Field

By taking the time derivative of (2.4), the continuous model of acceleration yields

$$\dot{\boldsymbol{\eta}}' = \ddot{\boldsymbol{\xi}}(X) - \mathrm{ad}_{\dot{\boldsymbol{\xi}}(X)}\boldsymbol{\eta}(X) - \mathrm{ad}_{\boldsymbol{\xi}(X)}\dot{\boldsymbol{\eta}}(X) \tag{2.5}$$

2.3 Static Model

The assumption of quasi-static motion is adequate to describe and control soft continuum robots. This section starts by reviewing the development of the static Cosserat rod equations which are a set of nonlinear ordinary differential equations (ODEs) based on idealizing an elastic rod as a one-dimensional object. After that, the formulation that combines Lie group, Lie algebra and screw theory is adopted to express the statics.

2.3.1 Static Cosserat Rod Equations

The static equilibrium equations are obtained by analyzing an infinitesimal slice of the Cosserat rod actuated by cables from X to $X + \mathrm{d}X$ with respect to the inertial frame. The internal force of the rod is denoted by $\boldsymbol{n}(X)$, and the internal torque by $\boldsymbol{m}(X)$. $\boldsymbol{\phi}(X)$ and $\boldsymbol{o}(X)$ represent the actuation force and torque acting on the midline of the Cosserat rod, respectively. The force $\underline{\boldsymbol{n}}(X)$ for unit of X acting on the rod are integrated over the cross section to describe the action on the rod midline, which can be idealized as one-dimensional distributed force $\overline{\boldsymbol{n}}(X)$ and distributed torque $\overline{\boldsymbol{m}}(X)$, as illustrated in Fig. 2.2. By means of the sign convention, the static force balance equation is given by

$$\boldsymbol{n}(X + \mathrm{d}X) + \boldsymbol{\phi}(X + \mathrm{d}X) - \boldsymbol{\phi}(X) - \boldsymbol{n}(X) + \int_X^{X+\mathrm{d}X} \overline{\boldsymbol{n}}(s)\mathrm{d}s = \boldsymbol{0}$$

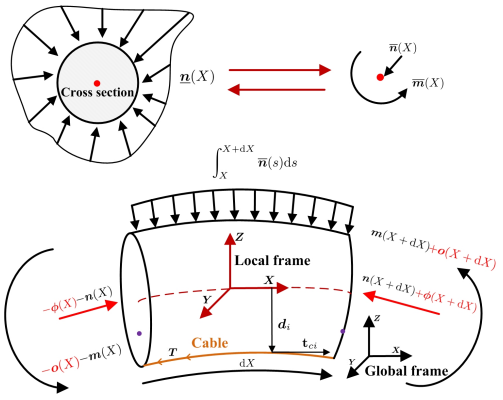


Figure 2.2: The force and torque balance analysis of an infinitesimal material element $\mathrm{d}X$ from the Cosserat rod actuated by cables.

The above equation is differentiated to obtain the following static force balance equation

$$\boldsymbol{n}' + \boldsymbol{\phi}' = -\overline{\boldsymbol{n}}$$

Taking the torque balance of an infinitesimal material element yields

$$\boldsymbol{m}(X+\mathrm{d}X)+\boldsymbol{o}(X+\mathrm{d}X)-\boldsymbol{o}(X)-\boldsymbol{m}(X)+\boldsymbol{p}(X+\mathrm{d}X)\times\Big[\boldsymbol{n}(X+\mathrm{d}X)+\boldsymbol{\phi}(X+\mathrm{d}X)\Big]$$
$$-\boldsymbol{p}(X)\times\Big[\boldsymbol{\phi}(X)+\boldsymbol{n}(X)\Big]+\int_X^{X+\mathrm{d}X}\Big[\overline{\boldsymbol{m}}(s)+\boldsymbol{p}(s)\times\overline{\boldsymbol{n}}(s)\Big]\mathrm{d}s=\boldsymbol{0}$$

This is differentiated to arrive at

$$\boldsymbol{m}'=-\boldsymbol{o}'-\frac{\partial}{\partial X}\Big[\boldsymbol{p}\times(\boldsymbol{n}+\boldsymbol{\phi})\Big]-\overline{\boldsymbol{m}}-\boldsymbol{p}\times\overline{\boldsymbol{n}}=-\boldsymbol{o}'-\boldsymbol{p}'\times(\boldsymbol{n}+\boldsymbol{\phi})-\overline{\boldsymbol{m}}$$

As a result, a set of nonlinear ODEs describing the Cosserat static model in the global frame for any cross section yields

$$\begin{aligned}\boldsymbol{n}'+\boldsymbol{\phi}'+\overline{\boldsymbol{n}}&=\boldsymbol{0}\\ \boldsymbol{m}'+\boldsymbol{o}'+\boldsymbol{p}'\times\boldsymbol{n}+\boldsymbol{p}'\times\boldsymbol{\phi}+\overline{\boldsymbol{m}}&=\boldsymbol{0}\end{aligned}\tag{2.6}$$

Using the transformation relationship including $\boldsymbol{m}=\boldsymbol{R}\boldsymbol{M}$, $\boldsymbol{n}=\boldsymbol{R}\boldsymbol{N}$, $\overline{\boldsymbol{n}}=\boldsymbol{R}\overline{\boldsymbol{N}}$, $\overline{\boldsymbol{m}}=\boldsymbol{R}\overline{\boldsymbol{M}}$, $\boldsymbol{o}=\boldsymbol{R}\boldsymbol{O}$, $\boldsymbol{\phi}=\boldsymbol{R}\boldsymbol{\Phi}$, $\boldsymbol{R}'=\boldsymbol{R}\widetilde{\boldsymbol{K}}$, $\boldsymbol{p}'=\boldsymbol{R}\boldsymbol{Q}$ between the global and body frames, and substituting them into (2.6), the Cosserat statics in the body frame for any cross section can be formulated as

$$\begin{aligned}\boldsymbol{N}'+\boldsymbol{\Phi}'+\boldsymbol{K}\times\boldsymbol{N}+\boldsymbol{K}\times\boldsymbol{\Phi}+\overline{\boldsymbol{N}}&=\boldsymbol{0}\\ \boldsymbol{M}'+\boldsymbol{O}'+\boldsymbol{K}\times\boldsymbol{M}+\boldsymbol{Q}\times\boldsymbol{N}+\boldsymbol{K}\times\boldsymbol{O}+\boldsymbol{Q}\times\boldsymbol{\Phi}+\overline{\boldsymbol{M}}&=\boldsymbol{0}\end{aligned}\tag{2.7}$$

Let us specify the internal elastic wrench, the external wrench including the distributed self-weight and other external wrenches, and also autonomous time-dependent internal forces and torques produced by the actuation. These are given by: $\boldsymbol{\mathcal{F}}_{ie}=\begin{pmatrix}\boldsymbol{M}^\top & \boldsymbol{N}^\top\end{pmatrix}^\top\in\mathbb{R}^6$, $\overline{\boldsymbol{\mathcal{F}}}_e=\begin{pmatrix}\overline{\boldsymbol{M}}^\top & \overline{\boldsymbol{N}}^\top\end{pmatrix}^\top\in\mathbb{R}^6$, $\boldsymbol{\mathcal{F}}_{ia}=\begin{pmatrix}\boldsymbol{O}^\top & \boldsymbol{\Phi}^\top\end{pmatrix}^\top\in\mathbb{R}^6$. Thus, the static model (2.7) can be expressed as a concise and elegant form

$$\begin{aligned}\boldsymbol{\mathcal{F}}_i'-\mathrm{ad}_{\boldsymbol{\xi}}^\top\boldsymbol{\mathcal{F}}_i+\overline{\boldsymbol{\mathcal{F}}}_e&=\boldsymbol{0}\\ \boldsymbol{\mathcal{F}}_i&=\boldsymbol{\mathcal{F}}_{ie}+\boldsymbol{\mathcal{F}}_{ia}\end{aligned}\tag{2.8}$$

with the boundary conditions (BCs) at $X=0$ and $X=L$

$$\begin{aligned}\boldsymbol{\mathcal{F}}_i(0)&=-\boldsymbol{\mathcal{F}}_{e0},\ \text{or}\ \boldsymbol{g}(0)=\boldsymbol{g}_0\\ \boldsymbol{\mathcal{F}}_i(L)&=\boldsymbol{\mathcal{F}}_{eL},\ \text{or}\ \boldsymbol{g}(L)=\boldsymbol{g}_L\end{aligned}\tag{2.9}$$

where $\boldsymbol{\mathcal{F}}_{e0}$ and $\boldsymbol{\mathcal{F}}_{eL}$ are tip external wrenches at $X=0$ and $X=L$, respectively. The ODE with BCs is used to describe the statics of the slender soft robot. The material constitutive law, which relates the internally elastic wrench to strain twists, must be introduced. The linear constitutive relationship used follows the form

$$\boldsymbol{\mathcal{F}}_{ie}(X)=\boldsymbol{\Sigma}(X)(\boldsymbol{\xi}(X)-\boldsymbol{\xi}_0)\tag{2.10}$$

with

$$\boldsymbol{\Sigma}(X)=\begin{bmatrix}\boldsymbol{K}_{tb} & \\ & \boldsymbol{K}_{es}\end{bmatrix}$$

where $\boldsymbol{K}_{tb}=\mathrm{diag}\left(GJ_x(X),EJ_y(X),EJ_z(X)\right)\in\mathbb{R}^{3\times3}$ is stiffness matrix for torsion and bending, $\boldsymbol{K}_{es}=\mathrm{diag}\left(EA(X),GA(X),GA(X)\right)\in\mathbb{R}^{3\times3}$ denotes stiffness matrix for extension and shear,

which are determined by the material properties and cross-sectional geometry; $\boldsymbol{\xi}_0$ represents the strain twist related to undeformed configuration of the manipulator. Note that a rod cross-section does not require to be circular, and it is only needed that the studied manipulator is slender, which implies that its length is much larger than its radius [128].

The external wrench appearing in (2.8) only considers gravity while other external forces are negligible and set to zero, which can be expressed as

$$\overline{\boldsymbol{\mathcal{F}}}_e(X) = \boldsymbol{\mathcal{M}}(X)\mathrm{Ad}^{-1}_{\boldsymbol{g}(X)}\mathrm{Ad}^{-1}_{\boldsymbol{g}_r}\boldsymbol{\mathcal{G}} \tag{2.11}$$

with

$$\boldsymbol{\mathcal{M}}(X) = \begin{pmatrix} \rho\boldsymbol{\mathcal{J}}(X) & \mathbf{0}_3 \\ \mathbf{0}_3 & \rho A(X)\mathbf{I}_3 \end{pmatrix} \in \mathbb{R}^{6\times 6}$$

where $\boldsymbol{\mathcal{M}}(X)$ is the mass matrix for unit of X, $\boldsymbol{\mathcal{J}}(X)$ represents the second moment of area tensor, $\boldsymbol{g}_r$ is the transformation of the base frame w.r.t. the inertial frame, $\boldsymbol{\mathcal{G}} = [0, 0, 0, 0, 0, -9.81]^\top$ is the gravity acceleration twist w.r.t. the global frame; the operator $\mathrm{Ad}^{-1}_{\boldsymbol{g}}$ transforms a twist (velocity or acceleration) from global frame to body frame.

2.3.2 Cable-Driven Actuation Model

The actuation wrench $\boldsymbol{\mathcal{F}}_{ia}$ depends on the type of actuators used, and the most common actuation manners for soft manipulators are tendon, fluidic and pneumatic actuators. The following will concentrate on the modeling of the cable-driven actuator.

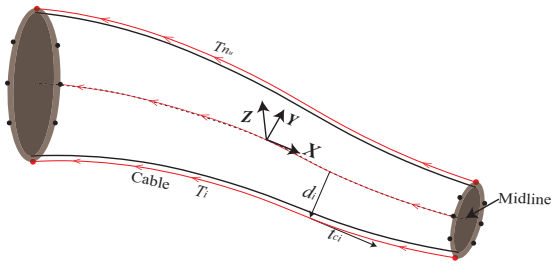


Figure 2.3: Schematics of the Cosserat rod actuated by cables.

Several cables are attached at the free end of the rod, and parallel to the surface of soft rod to produce the maximum torque under the effect of the cables and simultaneously reduce the friction between cables and soft rod, as shown in Fig. 2.3. The friction of the cables is not considered in this thesis. The magnitudes of the cables' tension are recorded as $T_1(t),\ T_2(t),\ \cdots,\ T_{n_u}(t)$, respectively. Additionally, the local distance between the midline of soft rod and the cable $i \in [1, n_u]$ at the cross section X is defined as $\boldsymbol{d}_i(X) \in \mathbb{R}^3$, then we can obtain the cable position vector in the global frame $\boldsymbol{p}_{ci} = \boldsymbol{p} + \boldsymbol{R}\boldsymbol{d}_i$. By taking the derivative of cable position vector w.r.t. X and normalizing, the unit vector $\mathbf{t}_{ci}(X, t) \in \mathbb{R}^3$ tangent to cable path can be expressed as

$$\mathbf{t}_{ci}(X,t) = \frac{\boldsymbol{R}^{-1}\boldsymbol{p}'_{ci}}{\|\boldsymbol{R}^{-1}\boldsymbol{p}'_{ci}\|} = \frac{\boldsymbol{R}^{-1}(\boldsymbol{p}' + \boldsymbol{R}'\boldsymbol{d}_i + \boldsymbol{R}\boldsymbol{d}'_i)}{\|\boldsymbol{R}^{-1}(\boldsymbol{p}' + \boldsymbol{R}'\boldsymbol{d}_i + \boldsymbol{R}\boldsymbol{d}'_i)\|} = \frac{\boldsymbol{Q} + \boldsymbol{K}\times\boldsymbol{d}_i + \boldsymbol{d}'_i}{\|\boldsymbol{Q} + \boldsymbol{K}\times\boldsymbol{d}_i + \boldsymbol{d}'_i\|_2}$$

The distance $\boldsymbol{d}_i(X)$ is fixed once the soft manipulator is designed. Accordingly, the unit tangent vector $\mathbf{t}_{ci}$ hinges on angular strain $\boldsymbol{K}(X)$ and linear strain $\boldsymbol{Q}(X)$ of the soft manipulator. The actuation wrench in the body frame can be obtained by calculating the torque and force

exerted by the cables parallel to the edge of the rod (see Fig. 2.3), and it is given by

$$\boldsymbol{\mathcal{F}}_{ia}(X) = \begin{bmatrix} \boldsymbol{d}_1 \times \mathbf{t}_{c1} & \boldsymbol{d}_2 \times \mathbf{t}_{c2} & \cdots & \boldsymbol{d}_{n_u} \times \mathbf{t}_{cn_u} \\ \mathbf{t}_{c1} & \mathbf{t}_{c2} & \cdots & \mathbf{t}_{cn_u} \end{bmatrix} \boldsymbol{T} = \boldsymbol{\Lambda}(X)\boldsymbol{T} \tag{2.12}$$

where $\boldsymbol{T} = [T_1, T_2, \cdots, T_{n_u}]^\top \in \mathbb{R}^{n_u}$, and $\boldsymbol{\Lambda}(X) \in \mathbb{R}^{6\times n_u}$ is a matrix mapping the cables' tension to the body frame of the arm.

2.4 Dynamic Model

With the aim to make the discussion of the discretization more concrete, we begin with presenting the PDE system to describe the Cosserat dynamics.

Taking infinitesimal material element of the Cosserat rod from X to $X + \mathrm{d}X$ for the force and torque analysis with respect to the inertial frame, we can use the classical Newtonian mechanics to describe translation and rotation of the infinitesimal material element. From the force and torque analysis, the PDE set of the Cosserat dynamics in the inertial frame (in accordance with the inertial frame (p_1, p_2, p_3) given in Fig. 2.1) can be derived

$$\begin{aligned} \rho A \mathbf{I}_3 \ddot{\boldsymbol{p}} &= \boldsymbol{n}' + \boldsymbol{\phi}' + \overline{\boldsymbol{n}} \\ \rho \boldsymbol{I}\dot{\boldsymbol{\omega}} + \boldsymbol{\omega} \times \rho \boldsymbol{I}\boldsymbol{\omega} &= \boldsymbol{m}' + \boldsymbol{o}' + \boldsymbol{p}' \times \boldsymbol{n} + \boldsymbol{p}' \times \boldsymbol{\phi} + \overline{\boldsymbol{m}} \end{aligned} \tag{2.13}$$

where $\boldsymbol{n}'$ and $\boldsymbol{m}'$ represent the change rates of the internal force and torque from the material at $\boldsymbol{p}(X + \mathrm{d}X, t)$ to the material at $\boldsymbol{p}(X, t)$ with respect to the arc length X, respectively.

Remark 1. *Denoting the second moment of area tensor in the body frame as $\boldsymbol{\mathcal{J}}$, and knowing that the kinetic energy of the rotating body is independent of the frame [124], we have*

$$\frac{1}{2}\boldsymbol{\Omega}^\top \boldsymbol{\mathcal{J}}\boldsymbol{\Omega} = \frac{1}{2}\boldsymbol{\omega}^\top \boldsymbol{I}\boldsymbol{\omega} = \frac{1}{2}\boldsymbol{\Omega}^\top (\boldsymbol{R}^\top \boldsymbol{I}\boldsymbol{R})\boldsymbol{\Omega}$$

In other words,

$$\boldsymbol{I} = \boldsymbol{R}\boldsymbol{\mathcal{J}}\boldsymbol{R}^\top$$

The variation of the area moment of inertia $\boldsymbol{I}$ with the arc length X in the inertial frame will increase the computational complexity. To overcome this difficulty, the transformation relationships, including $\boldsymbol{\omega} = \boldsymbol{R}\boldsymbol{\Omega}$, $\dot{\boldsymbol{p}} = \boldsymbol{R}\boldsymbol{V}$, $\boldsymbol{I} = \boldsymbol{R}\boldsymbol{\mathcal{J}}\boldsymbol{R}^\top$, and $\dot{\boldsymbol{R}} = \boldsymbol{R}\widetilde{\boldsymbol{\Omega}}$, are utilized to rewrite (2.13) in the associated body frame, where the area moment of inertia becomes a constant. Substituting these relationships into (2.13), the strong form of the dynamics in the body frame can be then formulated as

$$\begin{aligned} \rho A \mathbf{I}_3 \dot{\boldsymbol{V}} + \boldsymbol{\Omega} \times \rho A \mathbf{I}_3 \boldsymbol{V} &= \boldsymbol{N}' + \boldsymbol{\Phi}' + \boldsymbol{K} \times \boldsymbol{N} + \boldsymbol{K} \times \boldsymbol{\Phi} + \overline{\boldsymbol{N}} \\ \rho \boldsymbol{\mathcal{J}}\dot{\boldsymbol{\Omega}} + \boldsymbol{\Omega} \times \rho \boldsymbol{\mathcal{J}}\boldsymbol{\Omega} &= \boldsymbol{M}' + \boldsymbol{O}' + \boldsymbol{K} \times \boldsymbol{M} + \boldsymbol{Q} \times \boldsymbol{N} + \boldsymbol{K} \times \boldsymbol{O} + \boldsymbol{Q} \times \boldsymbol{\Phi} + \overline{\boldsymbol{M}} \end{aligned} \tag{2.14}$$

Using the notations of twists and wrenches presented in Section 2.2 and Section 2.3, the Cosserat dynamic system (2.14) can be then re-written as the following compact form:

$$\begin{aligned} \boldsymbol{\mathcal{M}}\dot{\boldsymbol{\eta}} - \mathrm{ad}_{\boldsymbol{\eta}}^\top \boldsymbol{\mathcal{M}}\boldsymbol{\eta} &= \boldsymbol{\mathcal{F}}_i' - \mathrm{ad}_{\boldsymbol{\xi}}^\top \boldsymbol{\mathcal{F}}_i + \overline{\boldsymbol{\mathcal{F}}}_e \\ \boldsymbol{\mathcal{F}}_i &= \boldsymbol{\mathcal{F}}_{ie} + \boldsymbol{\mathcal{F}}_{ia} \end{aligned} \tag{2.15}$$

with the same BCs at $X = 0$ and $X = L$ as (2.9). This PDE with the BCs is called strong form, if its solution satisfies those equations at any point in its domain. For the Cosserat case, it means that $\boldsymbol{\xi}(X)$ should satisfy (2.15) and (2.9) for any $X \in [0, L]$.

The internal wrench $\boldsymbol{\mathcal{F}}_i$ and external wrench $\overline{\boldsymbol{\mathcal{F}}}_e$ have been specified in Section 2.3, however, the internal elastic wrench $\boldsymbol{\mathcal{F}}_{ie}$ of any cross section appearing in (2.15) for the Cosserat rod requires to be re-specified due to the existence of viscosity modulus. As far as the internal elastic wrench is concerned, the Kelvin-Voigt model [129] is adopted both for the elastic and viscous members because of the constitutive material behavior of the slender soft robot.

$$\boldsymbol{\mathcal{F}}_{ie}(X) = \boldsymbol{\Sigma}(X)(\boldsymbol{\xi}(X) - \boldsymbol{\xi}_0) + \boldsymbol{\gamma}(X)\dot{\boldsymbol{\xi}}(X) \tag{2.16}$$

with

$$\boldsymbol{\gamma}(X) = \begin{bmatrix} \boldsymbol{D}_{tb} & \\ & \boldsymbol{D}_{es} \end{bmatrix}$$

where $\boldsymbol{D}_{tb} = \text{diag}\left(\mu J_x(X), 3\mu J_y(X), 3\mu J_z(X)\right) \in \mathbb{R}^{3\times 3}$ represents viscosity matrix for torsion and bending, $\boldsymbol{D}_{es} = \text{diag}\left(3\mu A(X), \mu A(X), \mu A(X)\right) \in \mathbb{R}^{3\times 3}$ is viscosity matrix for extension and shear, in which the scalar 3 stands for the Trouton's ratio between shear and extension viscosity for incompressible Newtonian fluids and holds more generally for viscoelastic fluids in the limit of very small strain rates [129], and μ is the viscosity modulus.

2.5 Conclusion

In this chapter, the kinematic, differential kinematic, static and dynamic models of the continuous Cosserat beam internally driven by distributed cables have been presented using the geometric formulations and notations. The continuous models are based on the classical Cosserat beam theory, whose differential equations are formulated on the Lie group and useful for the development of the discrete Cosserat modeling approach proposed in the following chapter.

Chapter 3

Piecewise Linear Strain Cosserat Model for Slender Soft Manipulator

3.1 Introduction

Precise simulation and control of slender soft robots are challenging as a consequence of the absence of dynamic models that are accurate and computationally efficient for large deformations. Therefore, our primary goal is to develop the dynamics formulation capable of addressing geometric nonlinearities for slender soft robots based on the Cosserat rod theory.

In this chapter, we will present a detailed derivation of the piecewise linear strain (PLS) Cosserat modeling method, a fast numerical scheme to improve the computational efficiency of PLS Cosserat dynamics, a model order reduction scheme via PLS Cosserat, and model parameter identification schemes, respectively. Also, a series of simulations and experiments have been provided to prove the accuracy of the proposed Cosserat models, and the related video demonstration is available [1].

3.2 Motivation

The Cosserat rod model is closed to the dynamic deformation behaviors of the slender soft robots, as it can exactly capture nonlinearity in the deformations involving bending, torsion, extension, and shear. Despite its accuracy to the continuum mechanics of the slender soft robots, the resulting highly nonlinear PDE with the boundary conditions, i.e., strong (continuous) form, is difficult (or even impossible) to be analytically solved.

Indeed, due to the challenge posed by the analytical solution of the strong (continuous) form, different approaches have been employed to approximate such a strong form. In [53], by introducing the virtual displacement, the D'Alembert's principle is used to obtain the weak form of the continuous dynamics, which simplifies the complexity of the state space. Then, the length space was discretized into several sections, and the strain was assumed to be constant for each section in order to deduce analytical formula. To reduce the dimension of the deduced system, [54] proposed a global approximation of strain field as $\boldsymbol{\xi}(X,t) = \boldsymbol{\Phi}(X)\boldsymbol{q}(t)$, where

[1] Video demonstration: `https://1drv.ms/v/s!Ak7hK-nVdKMei15WSgvHZ6-fyzQS?e=IaEzS0`

$\boldsymbol{\Phi}(X) = (\Phi_1, \Phi_2, \cdots, \Phi_n)$ defines a generalized matrix of n basis functions to parameterize the strain space with the vector $\boldsymbol{q}(t)$ of the strain-generalized coordinates. In this thesis, we attempt to combine those two ideas: small discretization to keep the local approximation precision (PCS), and interpolation to substantially decrease the dimension (GVS), so as to propose a PLS approach [130], which is capable of obtaining the analytical formula of the model and facilitating the controller design.

For clarity, we would like to give a global picture to illustrate the difference between PCS, GVS and PLS Cosserat modeling methods, and highlight the motivation and advantages of the PLS method.

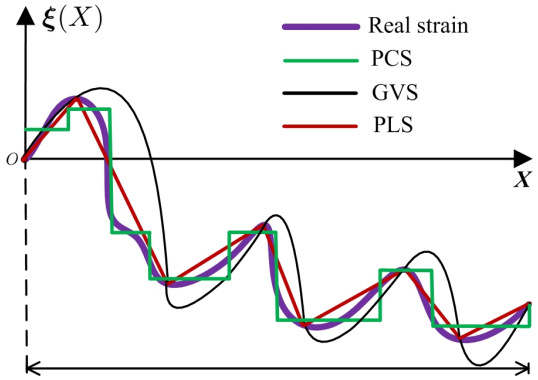


Figure 3.1: Sketch of comparison among PCS, GVS and PLS Cosserat models with real strain.

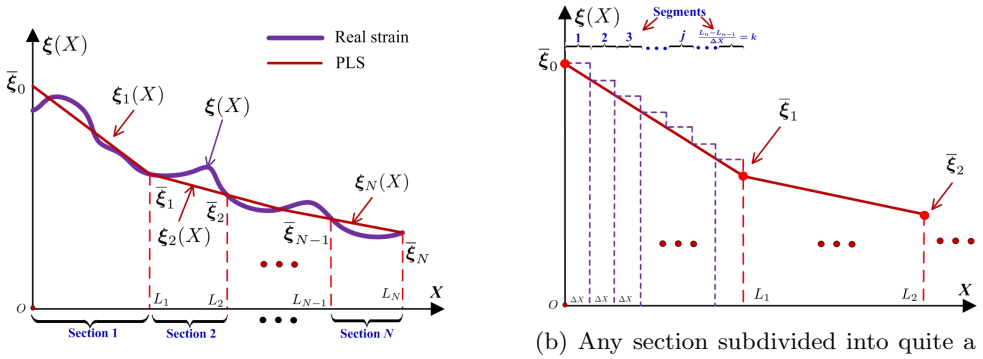
Given an arbitrary strain field of soft body subject to external forces, as described in Fig. 3.1, the PCS, GVS and PLS share the common objective: to approximate the real strain field as precise as possible. PCS adopts a local approximation scheme with local constant strain assumption, while GVS uses a global approximation manner via the chosen basis functions (polynomials for example). Clearly, PCS will result in high-dimensional system in order to reach a certain precision, while GVS will suffer from lower precision near the local mutation region even the number of basis function is increased. PLS combines both advantages of PCS and GVS, and proposes to locally use a linear strain to approximate the real one. Such a scheme is able to locally approximate to the real strain field of soft body compared to GVS, and get much less DoFs than PCS. The sketch described in Fig. 3.1 intuitively demonstrates that PLS has advantages over those proposed approaches in some engineering applications.

3.3 Piecewise Linear Strain Cosserat Model

3.3.1 PLS Cosserat Kinematic Model

Here, we will focus on the PLS Cosserat modeling principle. First of all, it is noted that a "section" is defined as a unit block which is able to produce independent mechanical deformation while "segments" are a subset of a section. To put it differently, one section is made up of quite a few segments, as illustrated in Fig. 3.2. In view of the strain field $\boldsymbol{\xi}(X)$ along the soft manipulator varying with the arc length X at the instant t, we divide the soft manipulator into N variable length continuum sections in the form of $[0, L_1]$, $[L_1, L_2] \cdots [L_{N-1}, L_N]$ (with $L_N = L$). In other words, the continuous strain field $\boldsymbol{\xi}(X)$ is substituted for a finite set of

N continuous strain fields of the form $\{\boldsymbol{\xi}_1(X), \boldsymbol{\xi}_2(X), \cdots, \boldsymbol{\xi}_N(X)\}$. To develop this discrete Cosserat model, for any continuum section n, we make the following two assumptions.



(a) Soft arm divided into several sections.

(b) Any section subdivided into quite a few segments.

Figure 3.2: Schematic illustration of the PLS Cosserat model.

Assumption 1. *The strain twists $\overline{\boldsymbol{\xi}}_{n-1}$ and $\overline{\boldsymbol{\xi}}_n$ respectively corresponding to those at the proximal and distal ends of any section n, are independent of X, yet other strain twists along the section n linearly vary with X, as shown in Fig. 3.2a. Regarding the two constant strain twists as those of linear interpolation nodes, the principle of the PLS can be then formulated as $\boldsymbol{\xi}_n(X) = a_n(X)\overline{\boldsymbol{\xi}}_{n-1} + b_n(X)\overline{\boldsymbol{\xi}}_n$ $(X \in [L_{n-1}, L_n])$, where $a_n(X) = \frac{L_n - X}{L_n - L_{n-1}}$, and $b_n(X) = \frac{X - L_{n-1}}{L_n - L_{n-1}}$.*

Assumption 2. *Since the geometric and differential kinematic models are still linear time-varying systems under the PLS assumption, the section n is subdivided into k ($k \in \mathbb{N}^+$) infinitesimal segments of the form $[L_{n-1}, L_{n-1} + \Delta X], [L_{n-1} + \Delta X, L_{n-1} + 2\Delta X], \cdots, [L_{n-1} + (k-1)\Delta X, L_{n-1} + k\Delta X]$. In this way, the strain twists $\boldsymbol{\xi}_n(X)$ along the segment j remain constant, i.e., $\boldsymbol{\xi}_n(X) \equiv \boldsymbol{\xi}_n(L_{n-1} + (j-1)\Delta X)$, $X \in [L_{n-1} + (j-1)\Delta X, L_{n-1} + j\Delta X]$. ΔX called one segment represents the infinitesimal distance between any adjacent cross sections, as shown in Fig. 3.2b.*

Remark 2. *When developing the proposed discrete Cosserat model, we have introduced the notions of section N and segment k. In general, the number of sections depends on the deformation variation of the soft manipulator under the external forces. If there exists the large deformation variation in the presence of external loads, N should be larger to yield more precise approximation, which in turn increases the computational cost; small value of N can also be used for the scenarios where the strains are not quickly spatial-varying with respect to X. Concerning the number of infinitesimal segments, it is in fact not related to the deformation variation and can take a larger value to improve the integral accuracy which however will increase computational complexity. Thus, a balance between the precision and the complexity needs to be taken into account in practice. In the subsequent derivation of PLS Cosserat models, it is assumed that these two variables are well pre-chosen according to the abovementioned criteria.*

The assumptions have significant implications to the discrete Cosserat modeling. Based on these assumptions, the DEs including (2.2), (2.4) and (2.5) can be seen as linear time-invariant systems and analytically solved for any segment j at the instant t. The initial configuration, velocity or acceleration for DEs of any segment j depend on the rightmost values of the previous

segment $(j-1)$ along the section n. In the remaining part of this subsection, we will separately illustrate how to solve these DEs in detail.

To guarantee the continuity, specifying the rightmost configuration $\boldsymbol{g}(L_{n-1})$ of the section $(n-1)$ as the initial value of the matrix DE (2.2) for section n, the rightmost configuration of any segment j along the section n at the instant t can be recursively expressed as

$$\begin{aligned}
\boldsymbol{g}(L_{n-1}+\Delta X) &= \boldsymbol{g}(L_{n-1})e^{\Delta X\hat{\boldsymbol{\xi}}_n(L_{n-1})}\\
\boldsymbol{g}(L_{n-1}+2\Delta X) &= \boldsymbol{g}(L_{n-1}+\Delta X)e^{\Delta X\hat{\boldsymbol{\xi}}_n(L_{n-1}+\Delta X)}\\
&= \boldsymbol{g}(L_{n-1})e^{\Delta X\hat{\boldsymbol{\xi}}_n(L_{n-1})}e^{\Delta X\hat{\boldsymbol{\xi}}_n(L_{n-1}+\Delta X)}\\
\boldsymbol{g}(L_{n-1}+3\Delta X) &= \boldsymbol{g}(L_{n-1}+2\Delta X)e^{\Delta X\hat{\boldsymbol{\xi}}_n(L_{n-1}+2\Delta X)}\\
&= \boldsymbol{g}(L_{n-1})e^{\Delta X\hat{\boldsymbol{\xi}}_n(L_{n-1})}e^{\Delta X\hat{\boldsymbol{\xi}}_n(L_{n-1}+\Delta X)}e^{\Delta X\hat{\boldsymbol{\xi}}_n(L_{n-1}+2\Delta X)}\\
&\vdots\\
\boldsymbol{g}(L_{n-1}+j\Delta X) &= \boldsymbol{g}(L_{n-1})\prod_{i=0}^{j-1}e^{\Delta X\hat{\boldsymbol{\xi}}_n(L_{n-1}+i\Delta X)}
\end{aligned}$$

where $i\in\mathbb{N}$, and $j\in\mathbb{N}^+$. Taking the rightmost configuration $\boldsymbol{g}(L_{n-1}+j\Delta X)$ of the segment j as the initial value of matrix DE (2.2) for segment $(j+1)$, the configuration of any cross section along the segment $(j+1)$ can be obtained. Assembling these solutions one by one, the integration of DE (2.2) analytically yields

$$\boldsymbol{g}(X)=\boldsymbol{g}(L_{n-1})\Big(\prod_{i=0}^{j-1}e^{\Delta X\hat{\boldsymbol{\xi}}_n(L_{n-1}+i\Delta X)}\Big)e^{(X-L_{n-1}-j\Delta X)\hat{\boldsymbol{\xi}}_n(L_{n-1}+j\Delta X)} \tag{3.2}$$

Inserting the PLS assumption into (3.2), the position and orientation of any cross section along the section n at the instant t can be analytically derived only given the strain twists ($\overline{\boldsymbol{\xi}}_{n-1}$ and $\overline{\boldsymbol{\xi}}_n$).

$$\begin{aligned}
\boldsymbol{g}(X) &= \boldsymbol{g}(L_{n-1})\left[\prod_{i=0}^{j-1}e^{\Delta X\left(\alpha_{ni}\hat{\overline{\boldsymbol{\xi}}}_{n-1}+\beta_{ni}\hat{\overline{\boldsymbol{\xi}}}_n\right)}\right]e^{(X-L_{n-1}-j\Delta X)\left[\alpha_{nj}\hat{\overline{\boldsymbol{\xi}}}_{n-1}+\beta_{nj}\hat{\overline{\boldsymbol{\xi}}}_n\right]}\\
&= \boldsymbol{g}(L_{n-1})\left(\prod_{i=0}^{j-1}e^{\Delta X\Theta_{ni}}\right)e^{(X-L_{n-1}-j\Delta X)\Theta_{nj}} = \boldsymbol{g}(L_{n-1})\boldsymbol{g}_{n(j-1)}\boldsymbol{g}_{nj}(X)\triangleq\boldsymbol{g}(L_{n-1})\boldsymbol{g}_n(X)
\end{aligned} \tag{3.3}$$

with $\alpha_{ni}=1-\frac{i\Delta X}{L_n-L_{n-1}}$, $\beta_{ni}=\frac{i\Delta X}{L_n-L_{n-1}}$, $\alpha_{nj}=1-\frac{j\Delta X}{L_n-L_{n-1}}$, $\beta_{nj}=\frac{j\Delta X}{L_n-L_{n-1}}$, $\Theta_{ni}=\alpha_{ni}\hat{\overline{\boldsymbol{\xi}}}_{n-1}+\beta_{ni}\hat{\overline{\boldsymbol{\xi}}}_n$ and $\Theta_{nj}=\alpha_{nj}\hat{\overline{\boldsymbol{\xi}}}_{n-1}+\beta_{nj}\hat{\overline{\boldsymbol{\xi}}}_n$ for segment i and j, where $\boldsymbol{g}_{n(j-1)}$ represents the rightmost configuration of a segment j along the section n w.r.t. the rightmost counterpart of the section $(n-1)$, and $\boldsymbol{g}_{nj}(X)$ denotes the configuration of any cross section at $X\in[L_{n-1}+j\Delta X, L_{n-1}+(j+1)\Delta X]$ along segment j w.r.t. the rightmost configuration of the segment $(j-1)$. Physically speaking, $\boldsymbol{g}_n(X)$ stands for the position and orientation of any cross section at X along the section n w.r.t. the rightmost counterparts of the section $(n-1)$. To satisfy computational accuracy and speed, Taylor series expansion of $\boldsymbol{g}_n(X)$ in (3.3) is performed, and it yields

$$\begin{aligned}
\boldsymbol{g}_n(X) = \boldsymbol{g}_{n(j-1)}\Big[&\mathbf{I}_4+(X-L_{n-1}-j\Delta X)\Theta_{nj}+\frac{1}{\theta_n^2}\Big(1-\cos\left((X-L_{n-1}-j\Delta X)\theta_n\right)\Big)\Theta_{nj}^2\\
&+\frac{1}{\theta_n^3}\Big((X-L_{n-1}-j\Delta X)\theta_n-\sin\left((X-L_{n-1}-j\Delta X)\theta_n\right)\Big)\Theta_{nj}^3\Big]
\end{aligned} \tag{3.4}$$

with $\theta_n^2 = \boldsymbol{K}_n^\top \boldsymbol{K}_n$, where θ_n in (3.4) relates to the angular strain twist $\boldsymbol{K}_n$ of the section n. For stress-free configuration of the section n, (3.4) becomes

$$\boldsymbol{g}_n(X) = \boldsymbol{g}_{n(j-1)} \left(\mathbf{I}_4 + (X - L_{n-1} - j\Delta X)\Theta_{nj}\right)$$

Intuitively, compared to the PCS modeling method where $\boldsymbol{g}(X)$ is only a function of one strain twist, the model deduced from PLS in (3.3) depends on both $\overline{\boldsymbol{\xi}}_{n-1}$ and $\overline{\boldsymbol{\xi}}_n$ of a certain section n.

3.3.2 PLS Cosserat Differential Kinematic Model

The differential kinematics aims to find the mapping between the velocity twist along the robot and the time derivatives of the system state representing deformation twist of the soft arm. In the same way, the rightmost velocity of any segment j along the section n at the instant t can be calculated by means of the integral of DE (2.4).

$$\boldsymbol{\eta}(L_{n-1} + \Delta X) = e^{-\Delta X \mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1})}} \boldsymbol{\eta}(L_{n-1}) + \int_{L_{n-1}}^{L_{n-1}+\Delta X} e^{(s-L_{n-1}-\Delta X)\mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1})}} \mathrm{d}s \dot{\boldsymbol{\xi}}_n(L_{n-1})$$

$$\begin{aligned}
\boldsymbol{\eta}(L_{n-1} + 2\Delta X) &= e^{-\Delta X \mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1}+\Delta X)}} \boldsymbol{\eta}(L_{n-1} + \Delta X) \\
&\quad + \int_{L_{n-1}+\Delta X}^{L_{n-1}+2\Delta X} e^{(s-L_{n-1}-2\Delta X)\mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1}+\Delta X)}} \mathrm{d}s \dot{\boldsymbol{\xi}}_n(L_{n-1} + \Delta X) \\
&= e^{-\Delta X \mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1}+\Delta X)}} e^{-\Delta X \mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1})}} \boldsymbol{\eta}(L_{n-1}) \\
&\quad + e^{-\Delta X \mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1}+\Delta X)}} \int_{L_{n-1}}^{L_{n-1}+\Delta X} e^{(s-L_{n-1}-\Delta X)\mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1})}} \mathrm{d}s \dot{\boldsymbol{\xi}}_n(L_{n-1}) \\
&\quad + \int_{L_{n-1}+\Delta X}^{L_{n-1}+2\Delta X} e^{(s-L_{n-1}-2\Delta X)\mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1}+\Delta X)}} \mathrm{d}s \dot{\boldsymbol{\xi}}_n(L_{n-1} + \Delta X)
\end{aligned}$$

$$\begin{aligned}
\boldsymbol{\eta}(L_{n-1} + 3\Delta X) &= e^{-\Delta X \mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1}+2\Delta X)}} \boldsymbol{\eta}(L_{n-1} + 2\Delta X) \\
&\quad + \int_{L_{n-1}+2\Delta X}^{L_{n-1}+3\Delta X} e^{(s-L_{n-1}-3\Delta X)\mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1}+2\Delta X)}} \mathrm{d}s \dot{\boldsymbol{\xi}}_n(L_{n-1} + 2\Delta X) \\
&= e^{-\Delta X \mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1}+2\Delta X)}} e^{-\Delta X \mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1}+\Delta X)}} e^{-\Delta X \mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1})}} \boldsymbol{\eta}(L_{n-1}) \\
&\quad + e^{-\Delta X \mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1}+2\Delta X)}} e^{-\Delta X \mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1}+\Delta X)}} \int_{L_{n-1}}^{L_{n-1}+\Delta X} e^{(s-L_{n-1}-\Delta X)\mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1})}} \mathrm{d}s \dot{\boldsymbol{\xi}}_n(L_{n-1}) \\
&\quad + e^{-\Delta X \mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1}+2\Delta X)}} \int_{L_{n-1}+\Delta X}^{L_{n-1}+2\Delta X} e^{(s-L_{n-1}-2\Delta X)\mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1}+\Delta X)}} \mathrm{d}s \dot{\boldsymbol{\xi}}_n(L_{n-1} + \Delta X) \\
&\quad + \int_{L_{n-1}+2\Delta X}^{L_{n-1}+3\Delta X} e^{(s-L_{n-1}-3\Delta X)\mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1}+2\Delta X)}} \mathrm{d}s \dot{\boldsymbol{\xi}}_n(L_{n-1} + 2\Delta X)
\end{aligned}$$

$$\boldsymbol{\eta}(L_{n-1} + j\Delta X) = \left[\prod_{i=0}^{j-1} e^{-\Delta X \mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1}+i\Delta X)}}\right] \boldsymbol{\eta}(L_{n-1}) + \sum_{i=1}^{j} \left\{ \left[\prod_{\tau=i}^{j-1} e^{-\Delta X \mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1}+\tau\Delta X)}}\right] \right.$$

$$\int_{L_{n-1}+(i-1)\Delta X}^{L_{n-1}+i\Delta X} e^{(s-L_{n-1}-i\Delta X)\mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1}+(i-1)\Delta X)}}\dot{\boldsymbol{\xi}}_n\Big(L_{n-1}+(i-1)\Delta X\Big)\mathrm{d}s\Bigg\}$$

After that, the velocity of each cross section along the section n at the instant t can be analytically obtained

$$\begin{aligned}\boldsymbol{\eta}(X) = e^{-(X-L_{n-1}-j\Delta X)\mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1}+j\Delta X)}}\Bigg\{&\Big[\prod_{i=0}^{j-1}e^{-\Delta X\mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1}+i\Delta X)}}\Big]\boldsymbol{\eta}(L_{n-1})\\ &+\sum_{i=1}^{j}\Big[\Big(\prod_{\tau=i}^{j-1}e^{-\Delta X\mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1}+\tau\Delta X)}}\Big)\int_{L_{n-1}+(i-1)\Delta X}^{L_{n-1}+i\Delta X}\\ &e^{(s-L_{n-1}-i\Delta X)\mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1}+(i-1)\Delta X)}}\dot{\boldsymbol{\xi}}_n\Big(L_{n-1}+(i-1)\Delta X\Big)\mathrm{d}s\Big]\Bigg\}\\ &+\int_{L_{n-1}+j\Delta X}^{X}e^{-(X-s)\mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1}+j\Delta X)}}\dot{\boldsymbol{\xi}}_n\Big(L_{n-1}+j\Delta X\Big)\mathrm{d}s\end{aligned}\tag{3.5}$$

For any cross section X at the instant t, (3.5) under the PLS assumption can be formulated as follows

$$\begin{aligned}\boldsymbol{\eta}(X) = e^{-(X-L_{n-1}-j\Delta X)\mathrm{ad}_{\Theta_{nj}^{\vee}}}\Bigg\{&\Big(\prod_{i=0}^{j-1}e^{-\mathrm{ad}_{\Delta X\Theta_{ni}^{\vee}}}\Big)\boldsymbol{\eta}(L_{n-1})\\ &+\sum_{i=1}^{j}\Big[\Big(\prod_{\tau=i}^{j-1}e^{-\mathrm{ad}_{\Delta X\Theta_{n\tau}^{\vee}}}\Big)\int_{L_{n-1}+(i-1)\Delta X}^{L_{n-1}+i\Delta X}e^{(s-L_{n-1}-i\Delta X)\mathrm{ad}_{\Theta_{n(i-1)}^{\vee}}}\alpha_{n(i-1)}\mathrm{d}s\Big]\dot{\bar{\boldsymbol{\xi}}}_{n-1}\\ &+\sum_{i=1}^{j}\Big[\Big(\prod_{\tau=i}^{j-1}e^{-\mathrm{ad}_{\Delta X\Theta_{n\tau}^{\vee}}}\Big)\int_{L_{n-1}+(i-1)\Delta X}^{L_{n-1}+i\Delta X}e^{(s-L_{n-1}-i\Delta X)\mathrm{ad}_{\Theta_{n(i-1)}^{\vee}}}\beta_{n(i-1)}\mathrm{d}s\Big]\dot{\bar{\boldsymbol{\xi}}}_{n}\Bigg\}\\ &+\int_{L_{n-1}+j\Delta X}^{X}e^{-(X-s)\mathrm{ad}_{\Theta_{nj}^{\vee}}}\alpha_{nj}\mathrm{d}s\dot{\bar{\boldsymbol{\xi}}}_{n-1}+\int_{L_{n-1}+j\Delta X}^{X}e^{-(X-s)\mathrm{ad}_{\Theta_{nj}^{\vee}}}\beta_{nj}\mathrm{d}s\dot{\bar{\boldsymbol{\xi}}}_{n}\end{aligned}\tag{3.6}$$

where

$$\begin{aligned}&e^{-(X-L_{n-1}-j\Delta X)\mathrm{ad}_{\Theta_{nj}^{\vee}}}\sum_{i=1}^{j}\Big[\Big(\prod_{\tau=i}^{j-1}e^{-\mathrm{ad}_{\Delta X\Theta_{n\tau}^{\vee}}}\Big)\int_{L_{n-1}+(i-1)\Delta X}^{L_{n-1}+i\Delta X}\\ &\quad e^{(s-L_{n-1}-i\Delta X)\mathrm{ad}_{\Theta_{n(i-1)}^{\vee}}}\alpha_{n(i-1)}\mathrm{d}s\Big]+\int_{L_{n-1}+j\Delta X}^{X}e^{-(X-s)\mathrm{ad}_{\Theta_{nj}^{\vee}}}\alpha_{nj}\mathrm{d}s\triangleq\mathrm{T}_{\boldsymbol{g}_{n1}(X)},\end{aligned}$$

$$\begin{aligned}&e^{-(X-L_{n-1}-j\Delta X)\mathrm{ad}_{\Theta_{nj}^{\vee}}}\sum_{i=1}^{j}\Big[\Big(\prod_{\tau=i}^{j-1}e^{-\mathrm{ad}_{\Delta X\Theta_{n\tau}^{\vee}}}\Big)\int_{L_{n-1}+(i-1)\Delta X}^{L_{n-1}+i\Delta X}\\ &\quad e^{(s-L_{n-1}-i\Delta X)\mathrm{ad}_{\Theta_{n(i-1)}^{\vee}}}\beta_{n(i-1)}\mathrm{d}s\Big]+\int_{L_{n-1}+j\Delta X}^{X}e^{-(X-s)\mathrm{ad}_{\Theta_{nj}^{\vee}}}\beta_{nj}\mathrm{d}s\triangleq\mathrm{T}_{\boldsymbol{g}_{n2}(X)},\end{aligned}$$

and the symbol $\vee$ is an operator about mapping a matrix into a vector.

According to the Proposition 2.25 in [131] that $e^{\mathrm{ad}_{\boldsymbol{\Theta}^{\vee}}}$ is equivalent to $\mathrm{Ad}_{e^{\boldsymbol{\Theta}}}$, we know that the coefficient matrix of the velocity $\boldsymbol{\eta}(L_{n-1})$ of tip at $X = L_{n-1}$ of the section n in (3.6)

is the exponential in the adjoint representation of the Lie group transformation $\boldsymbol{g}_n(X)$, i.e., $e^{-(X-L_{n-1}-j\Delta X)\mathrm{ad}_{\Theta_{nj}^\vee}}(\prod_{i=0}^{j-1} e^{-\mathrm{ad}_{\Delta X\Theta_{ni}^\vee}}) = \mathrm{Ad}^{-1}_{\boldsymbol{g}_n(X)}$. We define the coefficient matrices of $\dot{\overline{\boldsymbol{\xi}}}_{n-1}$ and $\dot{\overline{\boldsymbol{\xi}}}_n$ as $\mathrm{T}_{\boldsymbol{g}_{n1}(X)}$ and $\mathrm{T}_{\boldsymbol{g}_{n2}(X)}$, respectively, (3.6) can be then written as the following concise form

$$\boldsymbol{\eta}(X) = \mathrm{Ad}^{-1}_{\boldsymbol{g}_n(X)}\boldsymbol{\eta}(L_{n-1}) + \mathrm{T}_{\boldsymbol{g}_{n1}(X)}\dot{\overline{\boldsymbol{\xi}}}_{n-1} + \mathrm{T}_{\boldsymbol{g}_{n2}(X)}\dot{\overline{\boldsymbol{\xi}}}_n \tag{3.7}$$

From (3.7), if we know the strain twists ($\overline{\boldsymbol{\xi}}_{n-1}$ and $\overline{\boldsymbol{\xi}}_n$) and time derivative of strain twists ($\dot{\overline{\boldsymbol{\xi}}}_{n-1}$ and $\dot{\overline{\boldsymbol{\xi}}}_n$) of tips at $X = L_{n-1}$ and $X = L_n$, the velocity of any cross section X at the instant t along the section n can be recursively derived.

Due to the same reason, the rightmost acceleration of any segment j of the section n along the soft manipulator can be computed at the instant t.

$$\begin{aligned}
\dot{\boldsymbol{\eta}}(L_{n-1}+j\Delta X) = &\left[\prod_{i=0}^{j-1} e^{-\Delta X\mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1}+i\Delta X)}}\right]\dot{\boldsymbol{\eta}}(L_{n-1}) + \sum_{i=1}^{j}\left\{\left[\prod_{\tau=i}^{j-1} e^{-\Delta X\mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1}+\tau\Delta X)}}\right]\right.\\
&\left.\int_{L_{n-1}+(i-1)\Delta X}^{L_{n-1}+i\Delta X} e^{(s-L_{n-1}-i\Delta X)\mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1}+(i-1)\Delta X)}}\mathrm{d}s\ddot{\boldsymbol{\xi}}_n\Big(L_{n-1}+(i-1)\Delta X\Big)\right\}\\
&+\sum_{i=1}^{j}\left\{\left[\prod_{\tau=i}^{j-1} e^{-\Delta X\mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1}+\tau\Delta X)}}\right]\int_{L_{n-1}+(i-1)\Delta X}^{L_{n-1}+i\Delta X} e^{(s-L_{n-1}-i\Delta X)\mathrm{ad}_{\boldsymbol{\xi}_n(L_{n-1}+(i-1)\Delta X)}}\right.\\
&\left.\mathrm{ad}_{\boldsymbol{\eta}(\mathrm{s})}\mathrm{d}s\dot{\boldsymbol{\xi}}_n\Big(L_{n-1}+(i-1)\Delta X\Big)\right\}
\end{aligned}$$

Considering linearly variable strain twists along a certain section n, and using the property of Lie algebra that $\mathrm{ad}_m n = -\mathrm{ad}_n m$ holds for any m and n, the integration of DE (2.5) can be analytically solved with the appropriate initial value.

$$\begin{aligned}
\dot{\boldsymbol{\eta}}(X) = &\, e^{-(X-L_{n-1}-j\Delta X)\mathrm{ad}_{\Theta_{nj}^\vee}}\left\{\Big(\prod_{i=0}^{j-1} e^{-\mathrm{ad}_{\Delta X\Theta_{ni}^\vee}}\Big)\dot{\boldsymbol{\eta}}(L_{n-1})\right.\\
&+\sum_{i=1}^{j}\Big[\Big(\prod_{\tau=i}^{j-1} e^{-\mathrm{ad}_{\Delta X\Theta_{n\tau}^\vee}}\Big)\int_{L_{n-1}+(i-1)\Delta X}^{L_{n-1}+i\Delta X} e^{(s-L_{n-1}-i\Delta X)\mathrm{ad}_{\Theta_{n(i-1)}^\vee}}\alpha_{n(i-1)}\mathrm{d}s\Big]\ddot{\overline{\boldsymbol{\xi}}}_{n-1}\\
&+\sum_{i=1}^{j}\Big[\Big(\prod_{\tau=i}^{j-1} e^{-\mathrm{ad}_{\Delta X\Theta_{n\tau}^\vee}}\Big)\int_{L_{n-1}+(i-1)\Delta X}^{L_{n-1}+i\Delta X} e^{(s-L_{n-1}-i\Delta X)\mathrm{ad}_{\Theta_{n(i-1)}^\vee}}\beta_{n(i-1)}\mathrm{d}s\Big]\ddot{\overline{\boldsymbol{\xi}}}_{n}\\
&+\sum_{i=1}^{j}\Big[\Big(\prod_{\tau=i}^{j-1} e^{-\mathrm{ad}_{\Delta X\Theta_{n\tau}^\vee}}\Big)\int_{L_{n-1}+(i-1)\Delta X}^{L_{n-1}+i\Delta X} e^{(s-L_{n-1}-i\Delta X)\mathrm{ad}_{\Theta_{n(i-1)}^\vee}}\mathrm{ad}_{\boldsymbol{\eta}(\mathrm{s})}\alpha_{n(i-1)}\mathrm{d}s\Big]\dot{\overline{\boldsymbol{\xi}}}_{n-1}\\
&\left.+\sum_{i=1}^{j}\Big[\Big(\prod_{\tau=i}^{j-1} e^{-\mathrm{ad}_{\Delta X\Theta_{n\tau}^\vee}}\Big)\int_{L_{n-1}+(i-1)\Delta X}^{L_{n-1}+i\Delta X} e^{(s-L_{n-1}-i\Delta X)\mathrm{ad}_{\Theta_{n(i-1)}^\vee}}\mathrm{ad}_{\boldsymbol{\eta}(\mathrm{s})}\beta_{n(i-1)}\mathrm{d}s\Big]\dot{\overline{\boldsymbol{\xi}}}_{n}\right\}\\
&+\int_{L_{n-1}+j\Delta X}^{X} e^{-(X-s)\mathrm{ad}_{\Theta_{nj}^\vee}}\alpha_{nj}\mathrm{d}s\ddot{\overline{\boldsymbol{\xi}}}_{n-1} + \int_{L_{n-1}+j\Delta X}^{X} e^{-(X-s)\mathrm{ad}_{\Theta_{nj}^\vee}}\beta_{nj}\mathrm{d}s\ddot{\overline{\boldsymbol{\xi}}}_{n}\\
&+\int_{L_{n-1}+j\Delta X}^{X} e^{-(X-s)\mathrm{ad}_{\Theta_{nj}^\vee}}\mathrm{ad}_{\boldsymbol{\eta}(s)}\alpha_{nj}\mathrm{d}s\dot{\overline{\boldsymbol{\xi}}}_{n-1} + \int_{L_{n-1}+j\Delta X}^{X} e^{-(X-s)\mathrm{ad}_{\Theta_{nj}^\vee}}\mathrm{ad}_{\boldsymbol{\eta}(s)}\beta_{nj}\mathrm{d}s\dot{\overline{\boldsymbol{\xi}}}_{n}
\end{aligned} \tag{3.8}$$

where

$$e^{-(X-L_{n-1}-j\Delta X)\mathrm{ad}_{\Theta^\vee_{nj}}}\sum_{i=1}^{j}\Big[\Big(\prod_{\tau=i}^{j-1}e^{-\mathrm{ad}_{\Delta X\Theta^\vee_{n\tau}}}\Big)\int_{L_{n-1}+(i-1)\Delta X}^{L_{n-1}+i\Delta X}$$
$$e^{(s-L_{n-1}-i\Delta X)\mathrm{ad}_{\Theta^\vee_{n(i-1)}}}\mathrm{ad}_{\boldsymbol{\eta}(s)}\alpha_{n(i-1)}\mathrm{d}s\Big]+\int_{L_{n-1}+j\Delta X}^{X}e^{-(X-s)\mathrm{ad}_{\Theta^\vee_{nj}}}\mathrm{ad}_{\boldsymbol{\eta}(s)}\alpha_{nj}\mathrm{d}s\triangleq\mathrm{AD}_{\boldsymbol{g}_{n1}(X)}$$

and

$$e^{-(X-L_{n-1}-j\Delta X)\mathrm{ad}_{\Theta^\vee_{nj}}}\sum_{i=1}^{j}\Big[\Big(\prod_{\tau=i}^{j-1}e^{-\mathrm{ad}_{\Delta X\Theta^\vee_{n\tau}}}\Big)\int_{L_{n-1}+(i-1)\Delta X}^{L_{n-1}+i\Delta X}$$
$$e^{(s-L_{n-1}-i\Delta X)\mathrm{ad}_{\Theta^\vee_{n(i-1)}}}\mathrm{ad}_{\boldsymbol{\eta}(s)}\beta_{n(i-1)}\mathrm{d}s\Big]+\int_{L_{n-1}+j\Delta X}^{X}e^{-(X-s)\mathrm{ad}_{\Theta^\vee_{nj}}}\mathrm{ad}_{\boldsymbol{\eta}(s)}\beta_{nj}\mathrm{d}s\triangleq\mathrm{AD}_{\boldsymbol{g}_{n2}(X)}$$

For conciseness, the equivalence of the acceleration of each cross section along the section n in (3.8) conveniently yields

$$\dot{\boldsymbol{\eta}}(X)=\mathrm{Ad}^{-1}_{\boldsymbol{g}_n(X)}\dot{\boldsymbol{\eta}}(L_{n-1})+\mathrm{AD}_{\boldsymbol{g}_{n1}(X)}\dot{\overline{\boldsymbol{\xi}}}_{n-1}+\mathrm{AD}_{\boldsymbol{g}_{n2}(X)}\dot{\overline{\boldsymbol{\xi}}}_{n}+\mathrm{T}_{\boldsymbol{g}_{n1}(X)}\ddot{\overline{\boldsymbol{\xi}}}_{n-1}+\mathrm{T}_{\boldsymbol{g}_{n2}(X)}\ddot{\overline{\boldsymbol{\xi}}}_{n}\quad(3.9)$$

Thus, we can use (3.9) to calculate the acceleration of all cross sections along the section n if we know the strain twists ($\overline{\boldsymbol{\xi}}_{n-1}$ and $\overline{\boldsymbol{\xi}}_{n}$), the time derivative of strain twists ($\dot{\overline{\boldsymbol{\xi}}}_{n-1}$ and $\dot{\overline{\boldsymbol{\xi}}}_{n}$) and the second-order time derivative of strain twists ($\ddot{\overline{\boldsymbol{\xi}}}_{n-1}$ and $\ddot{\overline{\boldsymbol{\xi}}}_{n}$) of tips at $X=L_{n-1}$ and $X=L_n$.

The relation between the velocity twist $\boldsymbol{\eta}(X)$ along the robot and the strain twists ($\overline{\boldsymbol{\xi}}_{n-1}$ and $\overline{\boldsymbol{\xi}}_{n}$), and another relation between the acceleration twist $\dot{\boldsymbol{\eta}}(X)$ and the strain twists ($\overline{\boldsymbol{\xi}}_{n-1}$ and $\overline{\boldsymbol{\xi}}_{n}$) as well as time derivative of strain twists ($\dot{\overline{\boldsymbol{\xi}}}_{n-1}$ and $\dot{\overline{\boldsymbol{\xi}}}_{n}$) of the tips for the section n ought to be illustrated in order to derive the subsequent PLS Cosserat dynamic model. Applying (3.7) from base to tip for all cross sections along the soft manipulator in the chain, we can obtain the mapping as the geometric Jacobian which is an essential tool to describe the differential kinematics and dynamics of the PLS Cosserat model. Defining $\boldsymbol{S}_{(\cdot)}\in\mathbb{R}^{6\times6}$ and $\dot{\boldsymbol{S}}_{(\cdot)}\in\mathbb{R}^{6\times6}$ as the components of the Jacobian matrix and its time partial derivative, the structural form of Jacobian and its derivative over the length of the soft manipulator can be expressed as $\boldsymbol{J}(\overline{\boldsymbol{\xi}}_0,\overline{\boldsymbol{\xi}}_1,\cdots,\overline{\boldsymbol{\xi}}_{N-1},\overline{\boldsymbol{\xi}}_N,X)=[\boldsymbol{S}_0\quad\boldsymbol{S}_1\quad\boldsymbol{S}_2\quad\boldsymbol{S}_3\cdots\boldsymbol{S}_N]\in\mathbb{R}^{6\times6(N+1)}$, and $\dot{\boldsymbol{J}}(\overline{\boldsymbol{\xi}}_0,\overline{\boldsymbol{\xi}}_1,\cdots,\overline{\boldsymbol{\xi}}_N,\dot{\overline{\boldsymbol{\xi}}}_0,\dot{\overline{\boldsymbol{\xi}}}_1,\cdots,\dot{\overline{\boldsymbol{\xi}}}_{N-1},\dot{\overline{\boldsymbol{\xi}}}_N,X)=[\dot{\boldsymbol{S}}_0\quad\dot{\boldsymbol{S}}_1\quad\dot{\boldsymbol{S}}_2\quad\dot{\boldsymbol{S}}_3\cdots\dot{\boldsymbol{S}}_N]\in\mathbb{R}^{6\times6(N+1)}$, respectively.

The geometric Jacobian represents the relationship between the velocity twists of the soft robot and the time derivative of the deformations. The generalized strain vector $\boldsymbol{q}=[\overline{\boldsymbol{\xi}}_0^\top,\overline{\boldsymbol{\xi}}_1^\top,\overline{\boldsymbol{\xi}}_2^\top,\cdots,\overline{\boldsymbol{\xi}}_N^\top]^\top\in\mathbb{R}^{6(N+1)}$ composed of strain twists of all linear interpolation nodes is introduced, and the slender soft robot like a cantilever rod selected as an objective is fixed to a mobile base (the velocity twist of the fixed end $\boldsymbol{\eta}(0)=\boldsymbol{0}$). As a result, the discrete model of velocity (3.7) is equivalent to

$$\boldsymbol{\eta}(X)=\boldsymbol{J}(\boldsymbol{q},X)\dot{\boldsymbol{q}}\quad(3.10)$$

Finally, by taking the derivative of (3.10) w.r.t. time t, the acceleration twist $\dot{\boldsymbol{\eta}}(X)$ arrives at

$$\dot{\boldsymbol{\eta}}(X)=\boldsymbol{J}(\boldsymbol{q},X)\ddot{\boldsymbol{q}}+\dot{\boldsymbol{J}}(\boldsymbol{q},\dot{\boldsymbol{q}},X)\dot{\boldsymbol{q}}\quad(3.11)$$

From recursive use of (3.7) and (3.9), the thorough calculations of the Jacobian matrix $\boldsymbol{J}(\boldsymbol{q},X)$ and its time derivative $\dot{\boldsymbol{J}}(\boldsymbol{q},\dot{\boldsymbol{q}},X)$ can be obtained, as respectively represented in

$$
\boldsymbol{J}(\boldsymbol{q},X)=\begin{cases}
\overset{0}{\bullet}\xrightarrow{\qquad n\qquad}\overset{N}{\bullet}\\
\underbrace{{}^{\mathrm{T}}\boldsymbol{g}_{11}(X)\quad {}^{\mathrm{T}}\boldsymbol{g}_{12}(X)}_{\left[\boldsymbol{s}_0\quad \boldsymbol{s}_1\right]}\quad \mathbf{0}_6\quad\cdots\quad\mathbf{0}_6 & X\in(0,L_1]\\
\underbrace{{}^{\mathrm{Ad}^{-1}_{\boldsymbol{g}_2(X)}}{}^{\mathrm{T}}\boldsymbol{g}_{11}(L_1)\quad {}^{\mathrm{Ad}^{-1}_{\boldsymbol{g}_2(X)}}{}^{\mathrm{T}}\boldsymbol{g}_{12}(L_1)+{}^{\mathrm{T}}\boldsymbol{g}_{21}(X)\quad {}^{\mathrm{T}}\boldsymbol{g}_{22}(X)}_{\left[\boldsymbol{s}_0\quad \boldsymbol{s}_1\quad \boldsymbol{s}_2\right]}\quad \mathbf{0}_6\ \cdots\ \mathbf{0}_6 & X\in(L_1,L_2]\\
\underbrace{{}^{\mathrm{Ad}^{-1}_{\boldsymbol{g}_3(X)}}{}^{\mathrm{Ad}^{-1}_{\boldsymbol{g}_2(L_2)}}{}^{\mathrm{T}}\boldsymbol{g}_{11}(L_1)\quad {}^{\mathrm{Ad}^{-1}_{\boldsymbol{g}_3(X)}}\left[{}^{\mathrm{Ad}^{-1}_{\boldsymbol{g}_2(L_2)}}{}^{\mathrm{T}}\boldsymbol{g}_{12}(L_1)+{}^{\mathrm{T}}\boldsymbol{g}_{21}(L_2)\right]\ \cdots\ {}^{\mathrm{T}}\boldsymbol{g}_{32}(X)}_{\left[\boldsymbol{s}_0\quad \boldsymbol{s}_1\quad \boldsymbol{s}_2\quad \boldsymbol{s}_3\right]}\ \mathbf{0}_6\ \cdots\ \mathbf{0}_6 & X\in(L_2,L_3]\\
\vdots\qquad\qquad\vdots\qquad\qquad\qquad\qquad\vdots\\
\underbrace{\prod_{i=2}^{N}{}^{\mathrm{Ad}^{-1}_{\boldsymbol{g}_i\min(L_i,X)}}{}^{\mathrm{T}}\boldsymbol{g}_{11}(L_1)\qquad\cdots\qquad {}^{\mathrm{Ad}^{-1}_{\boldsymbol{g}_N(X)}}{}^{\mathrm{T}}\boldsymbol{g}_{(N-1)2}(X)+{}^{\mathrm{T}}\boldsymbol{g}_{N1}(X)\quad {}^{\mathrm{T}}\boldsymbol{g}_{N2}(X)}_{\left[\boldsymbol{s}_0\quad \boldsymbol{s}_1\quad \boldsymbol{s}_2\quad \boldsymbol{s}_3\quad\cdots\quad \boldsymbol{s}_N\right]} & X\in(L_{N-1},L_N]
\end{cases}
\tag{3.12}
$$

$$
\dot{\boldsymbol{J}}(\boldsymbol{q},\dot{\boldsymbol{q}},X)=\begin{cases}
{}^{\mathrm{AD}}\boldsymbol{g}_{11}(X)\quad {}^{\mathrm{AD}}\boldsymbol{g}_{12}(X)\quad \mathbf{0}_6\quad\cdots\quad\mathbf{0}_6\\
{}^{\mathrm{Ad}^{-1}_{\boldsymbol{g}_2(X)}}{}^{\mathrm{AD}}\boldsymbol{g}_{11}(L_1)\quad {}^{\mathrm{Ad}^{-1}_{\boldsymbol{g}_2(X)}}{}^{\mathrm{AD}}\boldsymbol{g}_{12}(L_1)+{}^{\mathrm{AD}}\boldsymbol{g}_{21}(X)\quad {}^{\mathrm{AD}}\boldsymbol{g}_{22}(X)\quad \mathbf{0}_6\quad\cdots\quad\mathbf{0}_6\\
{}^{\mathrm{Ad}^{-1}_{\boldsymbol{g}_3(X)}}{}^{\mathrm{Ad}^{-1}_{\boldsymbol{g}_2(L_2)}}{}^{\mathrm{AD}}\boldsymbol{g}_{11}(L_1)\quad {}^{\mathrm{Ad}^{-1}_{\boldsymbol{g}_3(X)}}\left[{}^{\mathrm{Ad}^{-1}_{\boldsymbol{g}_2(L_2)}}{}^{\mathrm{AD}}\boldsymbol{g}_{12}(L_1)+{}^{\mathrm{AD}}\boldsymbol{g}_{21}(L_2)\right]\ \cdots\ {}^{\mathrm{AD}}\boldsymbol{g}_{32}(X)\quad \mathbf{0}_6\ \cdots\ \mathbf{0}_6\\
\vdots\qquad\qquad\vdots\qquad\qquad\qquad\vdots\\
\Pi_{i=2}^{N}{}^{\mathrm{Ad}^{-1}_{\boldsymbol{g}_i\min(L_i,X)}}{}^{\mathrm{AD}}\boldsymbol{g}_{11}(L_1)\qquad\cdots\qquad\cdots\qquad {}^{\mathrm{Ad}^{-1}_{\boldsymbol{g}_N(X)}}{}^{\mathrm{AD}}\boldsymbol{g}_{(N-1)2}(X)+{}^{\mathrm{AD}}\boldsymbol{g}_{N1}(X)\quad {}^{\mathrm{AD}}\boldsymbol{g}_{N2}(X)
\end{cases}
\tag{3.13}
$$

(3.12) and (3.13). Note that the components in (3.13) can be denoted by the derivative of the corresponding counterparts of (3.12) w.r.t. time t. We would like to emphasize that, compared to PCS method, the deduced Jacobian matrix and its time derivative are significantly different.

3.3.3 PLS Cosserat Dynamic Model

In an effort to deduce Cosserat rod dynamic model corresponding to PDE (2.15), we introduce the following relation between virtual displacement and the state vector

$$
\delta\boldsymbol{\phi}(X)=\boldsymbol{J}(\boldsymbol{q},X)\boldsymbol{\mathcal{P}}\delta\boldsymbol{q}_a
$$

with $\boldsymbol{\mathcal{P}}=\begin{bmatrix}\mathbf{I}_{6N\times 6N}\\ \mathbf{0}_{6\times 6N}\end{bmatrix}\in\mathbb{R}^{6(N+1)\times 6N}$, and $\delta\boldsymbol{q}_a=\begin{bmatrix}\delta\overline{\boldsymbol{\xi}}_0^{\top} & \delta\overline{\boldsymbol{\xi}}_1^{\top} & \cdots & \delta\overline{\boldsymbol{\xi}}_{N-1}^{\top}\end{bmatrix}^{\top}\in\mathbb{R}^{6N}$, where $\boldsymbol{\mathcal{P}}$ is the selection matrix of the nodal strain twists for the purpose of making the increment of tip strain twist $\delta\overline{\boldsymbol{\xi}}_N$ equal to $\mathbf{0}$ (i.e., the tip strain twist increment $\delta\overline{\boldsymbol{\xi}}_N$ is only constrained by the boundary condition). The D'Alembert's principle is then used to calculate the total virtual works of the Cosserat beam, which allows us to obtain the weak form of (2.15). Substituting differential kinematic models (3.10) and (3.11) into the weak form, a nonlinear ODE can be obtained. Remarkably, the resulting ODE holds for $\forall\delta\boldsymbol{q}_a^{\top}\neq\mathbf{0}$, and thus the PLS Cosserat dynamic model yields

$$
\begin{aligned}
&\left(\boldsymbol{\mathcal{P}}^{\top}\int_0^{L_N}\boldsymbol{J}^{\top}\boldsymbol{\mathcal{M}}\boldsymbol{J}\mathrm{d}X\right)\ddot{\boldsymbol{q}}-\left[\boldsymbol{\mathcal{P}}^{\top}\int_0^{L_N}\boldsymbol{J}^{\top}\left(\mathrm{ad}^{\top}_{\boldsymbol{J}\dot{\boldsymbol{q}}}\boldsymbol{\mathcal{M}}\boldsymbol{J}-\boldsymbol{\mathcal{M}}\dot{\boldsymbol{J}}\right)\mathrm{d}X\right]\dot{\boldsymbol{q}}\\
&=\boldsymbol{\mathcal{P}}^{\top}\int_0^{L_N}\boldsymbol{J}^{\top}\left(\boldsymbol{\mathcal{F}}'_{ie}-\mathrm{ad}^{\top}_{\boldsymbol{\xi}}\boldsymbol{\mathcal{F}}_{ie}\right)\mathrm{d}X+\boldsymbol{\mathcal{P}}^{\top}\int_0^{L_N}\boldsymbol{J}^{\top}\left(\boldsymbol{\mathcal{F}}'_{ia}-\mathrm{ad}^{\top}_{\boldsymbol{\xi}}\boldsymbol{\mathcal{F}}_{ia}\right)\mathrm{d}X\\
&\quad+\left(\boldsymbol{\mathcal{P}}^{\top}\int_0^{L_N}\boldsymbol{J}^{\top}\boldsymbol{\mathcal{M}}\mathrm{Ad}^{-1}_{\boldsymbol{g}(X)}\mathrm{d}X\right)\mathrm{Ad}^{-1}_{\boldsymbol{g}_r}\boldsymbol{\mathcal{G}}
\end{aligned}
\tag{3.14}
$$

Let us define the coefficient matrices and wrenches in (3.14) as follows:

- $\boldsymbol{\mathcal{P}}^\top \int_0^{L_N} \boldsymbol{J}^\top \boldsymbol{\mathcal{M}} \boldsymbol{J} \mathrm{d}X = \boldsymbol{M}(\boldsymbol{q})$, the $6N \times 6(N+1)$ mass matrix.
- $-\boldsymbol{\mathcal{P}}^\top \int_0^{L_N} \boldsymbol{J}^\top \left(\mathrm{ad}^\top_{\boldsymbol{J}\dot{\boldsymbol{q}}} \boldsymbol{\mathcal{M}} \boldsymbol{J} - \boldsymbol{\mathcal{M}} \dot{\boldsymbol{J}} \right) \mathrm{d}X = \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$, the $6N \times 6(N+1)$ Coriolis matrix.
- $\boldsymbol{\mathcal{P}}^\top \int_0^{L_N} \boldsymbol{J}^\top \left(\boldsymbol{\mathcal{F}}'_{ie} - \mathrm{ad}^\top_{\boldsymbol{\xi}} \boldsymbol{\mathcal{F}}_{ie} \right) \mathrm{d}X = \boldsymbol{F}_{\mathrm{i}}(\boldsymbol{q}, \dot{\boldsymbol{q}})$, the $6N \times 1$ internal wrench.
- $\boldsymbol{\mathcal{P}}^\top \int_0^{L_N} \boldsymbol{J}^\top \left(\boldsymbol{\mathcal{F}}'_{ia} - \mathrm{ad}^\top_{\boldsymbol{\xi}} \boldsymbol{\mathcal{F}}_{ia} \right) \mathrm{d}X = \boldsymbol{F}_a(\boldsymbol{q})$, the $6N \times 1$ actuation wrench.
- $\boldsymbol{\mathcal{P}}^\top \int_0^{L_N} \boldsymbol{J}^\top \boldsymbol{\mathcal{M}} \mathrm{Ad}^{-1}_{\boldsymbol{g}(X)} \mathrm{d}X = \boldsymbol{G}(\boldsymbol{q})$, the $6N \times 6$ gravitational matrix.

Thanks to the definition of the internal wrench and PLS assumption, the stiffness and viscosity matrices can be decoupled from $\boldsymbol{F}_{\mathrm{i}}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ (see simplification details in Appendix B). The concise formulation can be then written as

$$\boldsymbol{F}_{\mathrm{i}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{K}(\boldsymbol{q})(\boldsymbol{q} - \boldsymbol{q}_0) + \boldsymbol{D}(\boldsymbol{q})\dot{\boldsymbol{q}} \tag{3.15}$$

where $\boldsymbol{K}(\boldsymbol{q}) \in \mathbb{R}^{6N \times 6(N+1)}$ stands for the stiffness matrix, $\boldsymbol{D}(\boldsymbol{q}) \in \mathbb{R}^{6N \times 6(N+1)}$ represents the viscosity matrix, and $\boldsymbol{q}_0 = [\boldsymbol{\xi}_{00}^\top, \boldsymbol{\xi}_{10}^\top, \boldsymbol{\xi}_{20}^\top, \cdots, \boldsymbol{\xi}_{N0}^\top]^\top \in \mathbb{R}^{6(N+1)}$ is the undeformed reference configuration of the slender soft manipulator.

Aside from (3.14), the BCs in (2.9) should be considered. Thus, a new PLS Cosserat dynamic system is given by

$$\begin{aligned} \boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \boldsymbol{G}(\boldsymbol{q})\mathrm{Ad}^{-1}_{\boldsymbol{g}_r}\boldsymbol{\mathcal{G}} - \boldsymbol{K}(\boldsymbol{q})(\boldsymbol{q} - \boldsymbol{q}_0) - \boldsymbol{D}(\boldsymbol{q})\dot{\boldsymbol{q}} &= \boldsymbol{F}_a(\boldsymbol{q}) \\ \boldsymbol{\Gamma}\dot{\boldsymbol{q}} + \boldsymbol{\sigma}(\boldsymbol{q} - \boldsymbol{q}_0) - \boldsymbol{\mathcal{F}}_e(L_N) &= -\boldsymbol{\mathcal{F}}_{ia}(L_N) \end{aligned} \tag{3.16}$$

with $\boldsymbol{\Gamma} = [\boldsymbol{0}_{6\times 6N}, \boldsymbol{\gamma}(L_N)]$, and $\boldsymbol{\sigma} = [\boldsymbol{0}_{6\times 6N}, \boldsymbol{\Sigma}(L_N)]$.

We would like to emphasize that the actuation wrench depends on the type of actuators used and the distribution of the chosen actuators on the soft manipulator. Taking the soft arm driven by cables for example, the non-parallel (spiral) cables distribution manner can create torsion and rotation motion. In this thesis, we generally choose to mount the cables in a special parallel manner.

Compared to PCS dynamic model where the mass matrix $\boldsymbol{M}(\boldsymbol{q})$ is square, the deduced PLS dynamic model has a non-square $\boldsymbol{M}(\boldsymbol{q})$ which is not positive definite. However, by complementing with the BCs (2.9) and the constitutive law (2.16), a similar Lagrangian model can be obtained. If the viscosity is not considered in (2.16), i.e., $\boldsymbol{\gamma} = \boldsymbol{0}$, then it leads to an algebraic equation in (3.16), which in fact is a differential-algebraic system. This characteristic, distinguished with the PCS and GVS methods, is exactly due to the PLS assumption.

The states of the PLS Cosserat dynamic system include the generalized strain $\boldsymbol{q}(t)$ and its derivative $\dot{\boldsymbol{q}}(t)$. The following part will introduce how to calculate the coefficient matrices and wrenches in (3.16). According to the structure of Jacobian matrix (5.4.2) and the property of piecewise integral, the exhaustive calculation of the mass matrix can be formulated as (3.17) shown at the bottom of next page, which is applicable to solve the Coriolis matrix $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ due to the same structure between them. In terms of the computation for 6×6 element component of row n and column m in $\boldsymbol{M}(\boldsymbol{q})$, the general element formulation is summarized as

$$\boldsymbol{M}_{(n,m)} = \sum_{j=\max(n-1,m-1)}^{N} \int_{L_{j-1}}^{L_j} \boldsymbol{S}_{n-1}^\top \boldsymbol{\mathcal{M}} \boldsymbol{S}_{m-1} \mathrm{d}X.$$

However, it should be noted that if $m = n = 1$ in the above equation, the value of j starts from 1.

The Coriolis matrix is split into two terms including $\boldsymbol{C}_1(\boldsymbol{q}, \dot{\boldsymbol{q}})$ and $\boldsymbol{C}_2(\boldsymbol{q}, \dot{\boldsymbol{q}})$ for calculation. With the same computation as $\boldsymbol{M}(\boldsymbol{q})$, the general expressions of each 6×6 element component of row n and column m in matrices $\boldsymbol{C}_1$ and $\boldsymbol{C}_2$ separately yield

$$\boldsymbol{C}_{1(n,m)} = -\sum_{j=\max(n-1,m-1)}^{N} \int_{L_{j-1}}^{L_j} \boldsymbol{S}_{n-1}^{\top} \mathrm{ad}_{\boldsymbol{J}\dot{\boldsymbol{q}}}^{\top} \boldsymbol{\mathcal{M}} \boldsymbol{S}_{m-1} \mathrm{d}X$$

$$\boldsymbol{C}_{2(n,m)} = \sum_{j=\max(n-1,m-1)}^{N} \int_{L_{j-1}}^{L_j} \boldsymbol{S}_{n-1}^{\top} \boldsymbol{\mathcal{M}} \dot{\boldsymbol{S}}_{m-1} \mathrm{d}X.$$

Likewise, the formulation of the gravitational matrix $\boldsymbol{G}(\boldsymbol{q})$ can be derived by implementing the following computation

$$\boldsymbol{G}(\boldsymbol{q}) = \begin{bmatrix} \sum_{j=1}^{N} \int_{L_{j-1}}^{L_j} \boldsymbol{S}_0^{\top} \boldsymbol{\mathcal{M}} \mathrm{Ad}_{\boldsymbol{g}(X)}^{-1} \mathrm{d}X \\ \sum_{j=1}^{N} \int_{L_{j-1}}^{L_j} \boldsymbol{S}_1^{\top} \boldsymbol{\mathcal{M}} \mathrm{Ad}_{\boldsymbol{g}(X)}^{-1} \mathrm{d}X \\ \sum_{j=2}^{N} \int_{L_{j-1}}^{L_j} \boldsymbol{S}_2^{\top} \boldsymbol{\mathcal{M}} \mathrm{Ad}_{\boldsymbol{g}(X)}^{-1} \mathrm{d}X \\ \vdots \\ \sum_{j=i}^{N} \int_{L_{j-1}}^{L_j} \boldsymbol{S}_i^{\top} \boldsymbol{\mathcal{M}} \mathrm{Ad}_{\boldsymbol{g}(X)}^{-1} \mathrm{d}X \\ \vdots \\ \sum_{j=N-1}^{N} \int_{L_{j-1}}^{L_j} \boldsymbol{S}_{N-1}^{\top} \boldsymbol{\mathcal{M}} \mathrm{Ad}_{\boldsymbol{g}(X)}^{-1} \mathrm{d}X \end{bmatrix}$$

Thus, the 6×6 element component of the row n in the matrix $\boldsymbol{G}(\boldsymbol{q})$ is given by

$$\boldsymbol{G}_{(n)} = \sum_{j=n-1}^{N} \int_{L_{j-1}}^{L_j} \boldsymbol{S}_{n-1}^{\top} \boldsymbol{\mathcal{M}} \mathrm{Ad}_{\boldsymbol{g}(X)}^{-1} \mathrm{d}X$$

Here we need to emphasize that when calculating the first 6×6 row element component, the value of j starts from 1.

$$\boldsymbol{M}(\boldsymbol{q}) = \begin{bmatrix} \sum_{j=1}^{N} \int_{L_{j-1}}^{L_j} \boldsymbol{S}_0^{\top} \boldsymbol{\mathcal{M}} \boldsymbol{S}_0 \mathrm{d}X & \sum_{j=1}^{N} \int_{L_{j-1}}^{L_j} \boldsymbol{S}_0^{\top} \boldsymbol{\mathcal{M}} \boldsymbol{S}_1 \mathrm{d}X & \cdots & \int_{L_{N-1}}^{L_N} \boldsymbol{S}_0^{\top} \boldsymbol{\mathcal{M}} \boldsymbol{S}_N \mathrm{d}X \\ \sum_{j=1}^{N} \int_{L_{j-1}}^{L_j} \boldsymbol{S}_1^{\top} \boldsymbol{\mathcal{M}} \boldsymbol{S}_0 \mathrm{d}X & \sum_{j=1}^{N} \int_{L_{j-1}}^{L_j} \boldsymbol{S}_1^{\top} \boldsymbol{\mathcal{M}} \boldsymbol{S}_1 \mathrm{d}X & \cdots & \int_{L_{N-1}}^{L_N} \boldsymbol{S}_1^{\top} \boldsymbol{\mathcal{M}} \boldsymbol{S}_N \mathrm{d}X \\ \sum_{j=2}^{N} \int_{L_{j-1}}^{L_j} \boldsymbol{S}_2^{\top} \boldsymbol{\mathcal{M}} \boldsymbol{S}_0 \mathrm{d}X & \sum_{j=2}^{N} \int_{L_{j-1}}^{L_j} \boldsymbol{S}_2^{\top} \boldsymbol{\mathcal{M}} \boldsymbol{S}_1 \mathrm{d}X & \cdots & \int_{L_{N-1}}^{L_N} \boldsymbol{S}_2^{\top} \boldsymbol{\mathcal{M}} \boldsymbol{S}_N \mathrm{d}X \\ \vdots & \vdots & \vdots & \vdots \\ \sum_{j=i}^{N} \int_{L_{j-1}}^{L_j} \boldsymbol{S}_i^{\top} \boldsymbol{\mathcal{M}} \boldsymbol{S}_0 \mathrm{d}X & \sum_{j=i}^{N} \int_{L_{j-1}}^{L_j} \boldsymbol{S}_i^{\top} \boldsymbol{\mathcal{M}} \boldsymbol{S}_1 \mathrm{d}X & \cdots & \int_{L_{N-1}}^{L_N} \boldsymbol{S}_i^{\top} \boldsymbol{\mathcal{M}} \boldsymbol{S}_N \mathrm{d}X \\ \vdots & \vdots & \vdots & \vdots \\ \sum_{j=N-1}^{N} \int_{L_{j-1}}^{L_j} \boldsymbol{S}_{N-1}^{\top} \boldsymbol{\mathcal{M}} \boldsymbol{S}_0 \mathrm{d}X & \sum_{j=N-1}^{N} \int_{L_{j-1}}^{L_j} \boldsymbol{S}_{N-1}^{\top} \boldsymbol{\mathcal{M}} \boldsymbol{S}_1 \mathrm{d}X & \cdots & \sum_{j=N-1}^{N} \int_{L_{j-1}}^{L_j} \boldsymbol{S}_{N-1}^{\top} \boldsymbol{\mathcal{M}} \boldsymbol{S}_N \mathrm{d}X \end{bmatrix} \tag{3.17}$$

Considering the structure of Jacobian matrix and substituting (2.12) into the actuation wrench lead to

$$\begin{aligned}\boldsymbol{F}_a(\boldsymbol{q}) &= \boldsymbol{\mathcal{P}}^\top \int_0^{L_N} \boldsymbol{J}^\top \left(\boldsymbol{\mathcal{F}}'_{ia} - \mathrm{ad}^\top_{\boldsymbol{\xi}(X)}\boldsymbol{\mathcal{F}}_{ia}\right) \mathrm{d}X \\ &= \begin{bmatrix} \sum_{j=1}^{N}\int_{L_{j-1}}^{L_j} \boldsymbol{S}_0^\top \left(\boldsymbol{\Lambda}' - \mathrm{ad}^\top_{\boldsymbol{\xi}_j(X)}\boldsymbol{\Lambda}\right)\mathrm{d}X \\ \sum_{j=1}^{N}\int_{L_{j-1}}^{L_j} \boldsymbol{S}_1^\top \left(\boldsymbol{\Lambda}' - \mathrm{ad}^\top_{\boldsymbol{\xi}_j(X)}\boldsymbol{\Lambda}\right)\mathrm{d}X \\ \sum_{j=2}^{N}\int_{L_{j-1}}^{L_j} \boldsymbol{S}_2^\top \left(\boldsymbol{\Lambda}' - \mathrm{ad}^\top_{\boldsymbol{\xi}_j(X)}\boldsymbol{\Lambda}\right)\mathrm{d}X \\ \vdots \\ \sum_{j=N-1}^{N}\int_{L_{j-1}}^{L_j} \boldsymbol{S}_{N-1}^\top \left(\boldsymbol{\Lambda}' - \mathrm{ad}^\top_{\boldsymbol{\xi}_j(X)}\boldsymbol{\Lambda}\right)\mathrm{d}X \end{bmatrix} \boldsymbol{T} \\ &= \boldsymbol{H}(\boldsymbol{q})\boldsymbol{T}\end{aligned} \tag{3.18}$$

with $\boldsymbol{\xi}_j(X) = \overline{\boldsymbol{\xi}}_{j-1}\frac{L_j - X}{L_j - L_{j-1}} + \overline{\boldsymbol{\xi}}_j \frac{X - L_{j-1}}{L_j - L_{j-1}}$, where $\boldsymbol{H}(\boldsymbol{q}) \in \mathbb{R}^{6N\times n_u}$ is the actuation matrix.

Substituting (3.18) into (3.16), and taking the time derivative of the second equation in (3.16), the PLS Cosserat dynamic system driven by cables is then given by

$$\begin{aligned}&\begin{bmatrix}\boldsymbol{M}(\boldsymbol{q}) \\ \boldsymbol{\Gamma}\end{bmatrix}\ddot{\boldsymbol{q}} + \begin{bmatrix}\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}}) \\ \boldsymbol{\sigma}\end{bmatrix}\dot{\boldsymbol{q}} - \begin{bmatrix}\boldsymbol{G}(\boldsymbol{q})\mathrm{Ad}^{-1}_{\boldsymbol{g}_r}\boldsymbol{\mathcal{G}} + \boldsymbol{K}(\boldsymbol{q})(\boldsymbol{q}-\boldsymbol{q}_0) + \boldsymbol{D}(\boldsymbol{q})\dot{\boldsymbol{q}} \\ \dot{\boldsymbol{\mathcal{F}}}_e(L_N)\end{bmatrix} \\ &\qquad = \begin{bmatrix}\boldsymbol{H}(\boldsymbol{q}) \\ \boldsymbol{0}_{6\times s}\end{bmatrix}\boldsymbol{T} + \begin{bmatrix}\boldsymbol{0}_{6N\times s} \\ -\boldsymbol{\Lambda}(L_N)\end{bmatrix}\dot{\boldsymbol{T}}\end{aligned} \tag{3.19}$$

Redefining each term and naming the coefficient matrices of (3.19), the generalized PLS Cosserat dynamic model can be then formulated as the following general Lagrangian structural form

$$\underline{\boldsymbol{M}}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \underline{\boldsymbol{C}}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \overline{\boldsymbol{K}}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \overline{\boldsymbol{H}}(\boldsymbol{q})\boldsymbol{T} + \overline{\boldsymbol{\mathcal{H}}}\dot{\boldsymbol{T}} \tag{3.20}$$

with the initial conditions at $t = t_0$

$$\boldsymbol{q}(t_0) = \begin{bmatrix}\overline{\boldsymbol{\xi}}_0^\top & \overline{\boldsymbol{\xi}}_1^\top & \cdots & \overline{\boldsymbol{\xi}}_N^\top\end{bmatrix}^\top$$

and

$$\dot{\boldsymbol{q}}(t_0) = \begin{bmatrix}\dot{\overline{\boldsymbol{\xi}}}_0^\top & \cdots & \dot{\overline{\boldsymbol{\xi}}}_{(N-1)}^\top & -(\boldsymbol{\Sigma}(\overline{\boldsymbol{\xi}}_N - \boldsymbol{\xi}_{N0}) - \boldsymbol{\mathcal{F}}_e + \boldsymbol{\Lambda}\boldsymbol{T})^\top\boldsymbol{\gamma}^{-\top}\end{bmatrix}^\top$$

where $\underline{\boldsymbol{M}}(\boldsymbol{q}) \in \mathbb{R}^{6(N+1)\times 6(N+1)}$ is a positive-definite mass matrix. It is worth noting that continuous Cosserat rod dynamic formulation of slender soft robots leads to a PDE in the form of a boundary value problem, however, the PLS dynamic model actuated by cables takes the form of a nonlinear ODE, which paves the way to the development of the identical model-based control methods used in the classical rigid robots.

3.3.4 Implicit Implementation

Choosing a proper numerical scheme is important when analyzing and implementing simulations. The Cosserat models are involved with high-frequency modes (extension and shear) and slow-frequency modes (bending and torsion). If using explicit time integration methods, the effects of high-frequency modes of the PLS Cosserat dynamic model may be lost when time step is larger, while the high modes will pose difficulties for efficient simulation of the slow modes due to the

small time step [52], which makes the dynamic system fail to achieve accurate simulation in real time. As a result, an implicit calculation framework for the PLS Cosserat dynamic system is established to address this issue.

First of all, the continuous time space is divided into discrete intervals: $[t_0, t_1, \cdots, t_{kt}]$. Then, let us consider a small time interval $[t_k, t_{k+1}]$, and $\Delta t = t_{k+1} - t_k$. Supposing the matrix $\underline{\boldsymbol{M}}(\boldsymbol{q})$ in (3.20) over this interval is constant, and adopting the semi-implicit Euler method [62], we can obtain:

$$\begin{aligned}\underline{\boldsymbol{M}}(\boldsymbol{q}_{t_k})\ddot{\boldsymbol{q}}_{t_{k+1}} = -\underline{\boldsymbol{C}}(\boldsymbol{q}_{t_k}, \dot{\boldsymbol{q}}_{t_k})\dot{\boldsymbol{q}}_{t_{k+1}} + \begin{bmatrix} \boldsymbol{G}(\boldsymbol{q}_{t_k})\mathrm{Ad}_{\boldsymbol{g}_r}^{-1}\boldsymbol{\mathcal{G}} + \boldsymbol{K}(\boldsymbol{q}_{t_k})(\boldsymbol{q}_{t_{k+1}} - \boldsymbol{q}_0) + \boldsymbol{D}(\boldsymbol{q}_{t_k})\dot{\boldsymbol{q}}_{t_{k+1}} \\ \dot{\boldsymbol{\mathcal{F}}}_e(L_N) \end{bmatrix} \\ + \overline{\boldsymbol{H}}(\boldsymbol{q}_{t_k})\boldsymbol{T} + \overline{\boldsymbol{\mathcal{H}}}\dot{\boldsymbol{T}}\end{aligned} \tag{3.21}$$

A less expensive method, though less accurate when the timestep is large is used to solve the system. Consequently, the formulation (3.21) can be re-written as follows:

$$\underbrace{\left(\underline{\boldsymbol{M}} + \underline{\boldsymbol{C}}\Delta t - \begin{bmatrix} \boldsymbol{K}\Delta^2 t + \boldsymbol{D}\Delta t \\ \boldsymbol{0}_{6\times 6(N+1)} \end{bmatrix}\right)}_{\boldsymbol{A}(\boldsymbol{q}_{t_k}, \dot{\boldsymbol{q}}_{t_k})} \ddot{\boldsymbol{q}}_{t_{k+1}} = \underbrace{\left(-\underline{\boldsymbol{C}} + \begin{bmatrix} \boldsymbol{K}\Delta t \\ \boldsymbol{0}_{6\times 6(N+1)} \end{bmatrix}\right)}_{\boldsymbol{B}(\boldsymbol{q}_{t_k}, \dot{\boldsymbol{q}}_{t_k})} \dot{\boldsymbol{q}}_{t_k} + \overline{\boldsymbol{K}} + \overline{\boldsymbol{H}}(\boldsymbol{q}_{t_k})\boldsymbol{T} + \overline{\boldsymbol{\mathcal{H}}}\dot{\boldsymbol{T}}$$

$$\dot{\boldsymbol{q}}_{t_{k+1}} = \dot{\boldsymbol{q}}_{t_k} + \ddot{\boldsymbol{q}}_{t_{k+1}}\Delta t$$

The simulation process starts with the cables' tension $\boldsymbol{T}(t)$ and its time derivative $\dot{\boldsymbol{T}}(t)$. Computational efficiency of the dynamic model is significantly improved due to the unconditional stability of implicit Euler scheme at large time steps. Besides, the PLS Cosserat dynamic system is numerically solved in MATLAB, and the simulation implementation turns out to be relatively simple.

3.3.5 Strain Mode Reduction

According to the Cosserat rod theory, the strain twist of any interpolation node $\overline{\boldsymbol{\xi}}_i$ could take any value in the six-dimensional components. However, the internal rod kinematics describing the motions between the cross sections can be constrained by some restrictions in the usual application of soft robots. For instance, if we neglect the transverse shear and the extensibility, and if we use the arc length as label X, one has $\boldsymbol{Q} = (1, 0, 0)^\top$, and the beam kinematics can be reduced from the six internal DoFs of a Cosserat beam to the three internal DoFs of those of a Kirchhoff beam [54]. To tackle these restrictions, we decompose the strain twist of any interpolation node i, by preserving the same variable order, as

$$\overline{\boldsymbol{\xi}}_i = \mathbf{B}_a\boldsymbol{\xi}_{ia}^* + \mathbf{B}_c\boldsymbol{\xi}_{ic}^* \tag{3.22}$$

where $\boldsymbol{\xi}_{ia}^* \in \mathbb{R}^{n_i}$ determines the vector field of the free strains (i.e., the number of DoFs n_i) of the interpolation nodes allowed by the rod kinematics, $\boldsymbol{\xi}_{ic}^* \in \mathbb{R}^{6-n_i}$ represents the vector field of constrained strains, $\mathbf{B}_a$ and $\mathbf{B}_c$ stand for the complementary selection matrix of 1 and 0 such that $\mathbf{B}_a^\top\mathbf{B}_a = \mathbf{I}_{(n_i)\times(n_i)}$, $\mathbf{B}_c^\top\mathbf{B}_c = \mathbf{I}_{(6-n_i)\times(6-n_i)}$, and $\mathbf{B}_a^\top\mathbf{B}_c = \mathbf{0}$. Since the variable order in $\overline{\boldsymbol{\xi}}_i$ is preserved during the decomposition, thus the matrices $\mathbf{B}_a$ and $\mathbf{B}_c$ are uniquely determined.

Substituting (3.22) into (3.3), the reduced geometric model can be obtained. For the constrained soft manipulator, it is necessary to consider the strain twists of all interpolation nodes. Hence, the generalized strain vector can be expressed as

$$\boldsymbol{q} = \overline{\mathbf{B}}_a\overline{\boldsymbol{q}} + \overline{\mathbf{B}}_c\underline{\boldsymbol{q}} \tag{3.23}$$

with $\overline{\mathbf{B}}_a = \mathbf{I}_{(N+1)\times(N+1)} \otimes \mathbf{B}_a$, $\overline{\mathbf{B}}_c = \mathbf{I}_{(N+1)\times(N+1)} \otimes \mathbf{B}_c$, where $\otimes$ represents the Kronecker tensor product. In such a way, $\overline{\mathbf{B}}_a \in \mathbb{R}^{6(N+1)\times[n_i(N+1)]}$ is the generalized selection matrix for the allowed states, $\overline{\boldsymbol{q}} = \begin{bmatrix}\boldsymbol{\xi}_{0a}^{*\top} & \boldsymbol{\xi}_{1a}^{*\top} & \boldsymbol{\xi}_{2a}^{*\top} & \cdots & \boldsymbol{\xi}_{Na}^{*\top}\end{bmatrix}^\top \in \mathbb{R}^{n_i(N+1)}$ includes the allowed DoFs of the soft manipulator.

For the PLS Cosserat model reduction, the internal elastic wrench should be divided into two parts: one is constrained wrench in charge of imposing the internal constraints for prohibited strains, another is the elastic wrench related to the allowed DoFs. Thus, the reduced internal wrench is given by

$$\boldsymbol{\mathcal{F}}_{ie} = \underbrace{\boldsymbol{\Sigma}(X)\mathbf{B}_a(\boldsymbol{\xi}_a^*(X) - \boldsymbol{\xi}_{i0}^*) + \boldsymbol{\gamma}(X)\mathbf{B}_a\dot{\boldsymbol{\xi}}_a^*(X)}_{\boldsymbol{\mathcal{F}}_{ie}^*} + \mathbf{B}_c\boldsymbol{\lambda}(X) \tag{3.24}$$

where $\boldsymbol{\xi}_a^*(X)$ for $X \in [L_{n-1}, L_n]$ can be obtained by the linear interpolation of allowed strains of the adjacent nodes, $\boldsymbol{\xi}_{i0}^*$ is the initial states of allowed strains, $\boldsymbol{\lambda}(X) \in \mathbb{R}^{(6-n_i)}$ is the constrained wrench.

Inserting (3.23) into (3.10) and (3.11), the reduced PLS Cosserat differential kinematics models yields

$$\begin{aligned}\boldsymbol{\eta}(X) &= \overline{\boldsymbol{J}}(\overline{\boldsymbol{q}}, X)\dot{\overline{\boldsymbol{q}}}\\ \dot{\boldsymbol{\eta}}(X) &= \overline{\boldsymbol{J}}(\overline{\boldsymbol{q}}, X)\ddot{\overline{\boldsymbol{q}}} + \dot{\overline{\boldsymbol{J}}}(\overline{\boldsymbol{q}}, \dot{\overline{\boldsymbol{q}}}, X)\dot{\overline{\boldsymbol{q}}}\end{aligned}$$

where $\overline{\boldsymbol{J}}(\overline{\boldsymbol{q}}, X) = \boldsymbol{J}(\boldsymbol{q}, X)\overline{\mathbf{B}}_a \in \mathbb{R}^{6\times[n_i(N+1)]}$ is the reduced Jacobian matrix. Using the relation $\delta\boldsymbol{\phi}(X) = \overline{\boldsymbol{J}}(\overline{\boldsymbol{q}}, X)\overline{\boldsymbol{\mathcal{P}}}\delta\overline{\boldsymbol{q}}_a$, and substituting the reduced kinematics relations as well as (3.24) into the weak form (2.15) lead to

$$\begin{aligned}&\underbrace{\left(\overline{\boldsymbol{\mathcal{P}}}^{\mathrm{T}}\int_0^{L_N}\overline{\boldsymbol{J}}^\top\boldsymbol{\mathcal{M}}\overline{\boldsymbol{J}}\mathrm{d}X\right)}_{\overline{\boldsymbol{M}}(\overline{\boldsymbol{q}})}\ddot{\overline{\boldsymbol{q}}} - \underbrace{\left[\overline{\boldsymbol{\mathcal{P}}}^\top\int_0^{L_N}\overline{\boldsymbol{J}}^\top\left(\mathrm{ad}^\top_{\overline{\boldsymbol{J}}\dot{\overline{\boldsymbol{q}}}}\boldsymbol{\mathcal{M}}\overline{\boldsymbol{J}} - \boldsymbol{\mathcal{M}}\dot{\overline{\boldsymbol{J}}}\right)\mathrm{d}X\right]}_{\overline{\boldsymbol{C}}(\overline{\boldsymbol{q}},\dot{\overline{\boldsymbol{q}}})}\dot{\overline{\boldsymbol{q}}}\\
&= \underbrace{\overline{\boldsymbol{\mathcal{P}}}^\top\int_0^{L_N}\overline{\boldsymbol{J}}^\top\left(\boldsymbol{\mathcal{F}}_{ie}^{*\prime} - \mathrm{ad}^\top_{\boldsymbol{\xi}}\boldsymbol{\mathcal{F}}_{ie}^*\right)\mathrm{d}X}_{\overline{\boldsymbol{F}}_{\mathrm{i}}(\overline{\boldsymbol{q}},\dot{\overline{\boldsymbol{q}}})} + \underbrace{\overline{\boldsymbol{\mathcal{P}}}^\top\int_0^{L_N}\overline{\boldsymbol{J}}^\top\overline{\boldsymbol{\mathcal{F}}}_e\mathrm{d}X}_{\overline{\boldsymbol{F}}_e(\overline{\boldsymbol{q}})} + \cdots\\
&\cdots + \underbrace{\overline{\boldsymbol{\mathcal{P}}}^\top\int_0^{L_N}\overline{\boldsymbol{J}}^\top\left(\boldsymbol{\mathcal{F}}_{ia}^{\prime} - \mathrm{ad}^\top_{\boldsymbol{\xi}}\boldsymbol{\mathcal{F}}_{ia}\right)\mathrm{d}X}_{\overline{\boldsymbol{F}}_a(\overline{\boldsymbol{q}})} + \underbrace{\overline{\boldsymbol{\mathcal{P}}}^\top\int_0^{L_N}\overline{\boldsymbol{J}}^\top\left(\mathbf{B}_c\boldsymbol{\lambda}' - \mathrm{ad}^\top_{\boldsymbol{\xi}}\mathbf{B}_c\boldsymbol{\lambda}\right)\mathrm{d}X}_{\overline{\boldsymbol{F}}_\lambda(\overline{\boldsymbol{q}})}\end{aligned} \tag{3.25}$$

where $\overline{\boldsymbol{\mathcal{P}}} \in \mathbb{R}^{n_i(N+1)\times n_i N}$ represents a selection matrix, $\overline{\boldsymbol{q}}_a = \begin{bmatrix}\boldsymbol{\xi}_{0a}^{*\top} & \boldsymbol{\xi}_{1a}^{*\top} & \boldsymbol{\xi}_{2a}^{*\top} & \cdots & \boldsymbol{\xi}_{(N-1)a}^{*\top}\end{bmatrix}^\top \in \mathbb{R}^{n_i N}$ is composed of allowed DoFs of all strain nodes except those of the free end. In addition, it is essential to emphasize that the reduced boundary condition can be formulated as

$$\mathbf{B}_a^{\mathrm{T}}\boldsymbol{\Sigma}(L_N)\mathbf{B}_a(\boldsymbol{\xi}_{Na}^* - \boldsymbol{\xi}_{N0}^*) + \mathbf{B}_a^\top\boldsymbol{\gamma}(L_N)\mathbf{B}_a\dot{\boldsymbol{\xi}}_{Na}^* = \mathbf{B}_a^\top\big(-\boldsymbol{\mathcal{F}}_{ia}(L_N) + \boldsymbol{\mathcal{F}}_e(L_N)\big) \tag{3.26a}$$

$$\boldsymbol{\lambda}(L_N) = \mathbf{B}_c^\top\big(-\boldsymbol{\mathcal{F}}_{ia}(L_N) + \boldsymbol{\mathcal{F}}_e(L_N)\big) \tag{3.26b}$$

All the items except the last one in (3.25) can be obtained by replacing $\boldsymbol{J}$ in (3.14) with $\boldsymbol{J}\overline{\mathbf{B}}_a$. Next, we focus on how to calculate the following integration item with unknown constrained wrench $\boldsymbol{\lambda}(X)$.

$$\overline{\boldsymbol{F}}_\lambda = \overline{\boldsymbol{\mathcal{P}}}^\top\int_0^{L_N}\overline{\boldsymbol{J}}^\top\left(\mathbf{B}_c\boldsymbol{\lambda}' - \mathrm{ad}^\top_{\boldsymbol{\xi}}\mathbf{B}_c\boldsymbol{\lambda}\right)\mathrm{d}X \tag{3.27}$$

Theorem 1. *For the PLS Cosserat model with full modes, if the strain field is re-formulated as $\boldsymbol{\xi}(X) = \boldsymbol{\Phi}(X)\boldsymbol{q}(t)$, where $\boldsymbol{\Phi}(X) \in \mathbb{R}^{6\times 6(N+1)}$ is a matrix comprised of coefficient of strain interpolation nodes via the PLS assumption, then there exists a relationship among three quantities (i.e., $\overline{\mathbf{B}}_a$, $\boldsymbol{\Phi}$, and $\mathbf{B}_c$) satisfying the following equality*

$$\overline{\mathbf{B}}_a^{\top}\boldsymbol{\Phi}^{\top}\mathbf{B}_c = \begin{bmatrix} a_1(X)\mathbf{B}_a^{\top} \\ \big(b_1(X)+a_2(X)\big)\mathbf{B}_a^{\top} \\ \big(b_{n-1}(X)+a_n(X)\big)\mathbf{B}_a^{\top} \\ \vdots \\ b_N(X)\mathbf{B}_a^{\top} \end{bmatrix}\mathbf{B}_c = \mathbf{0} \tag{3.28}$$

with $a_n(X) = \frac{L_n - X}{L_n - L_{n-1}}$, and $b_n(X) = \frac{X - L_{n-1}}{L_n - L_{n-1}}$, then the generalized constrained wrench $\overline{\boldsymbol{F}}_\lambda$ for the reduced PLS Cosserat relates to the constrained wrench of end cross section.

Proof. Note that we want to prove that $\overline{\boldsymbol{F}}_\lambda$ is only dependent on the constrained wrench at $X = L_N$. At this aim, let us insert (3.10) into (2.4), and it yields

$$\boldsymbol{J}'\dot{\boldsymbol{q}} = -\operatorname{ad}_{\boldsymbol{\xi}}\boldsymbol{J}\dot{\boldsymbol{q}} + \dot{\boldsymbol{\xi}}(X) = -\operatorname{ad}_{\boldsymbol{\xi}}\boldsymbol{J}\dot{\boldsymbol{q}} + \boldsymbol{\Phi}(X)\dot{\boldsymbol{q}}$$

which holds for $\forall \dot{\boldsymbol{q}} \neq \mathbf{0}$, and thus

$$\overline{\boldsymbol{J}}' = -\operatorname{ad}_{\boldsymbol{\xi}}\overline{\boldsymbol{J}} + \boldsymbol{\Phi}(X)\overline{\mathbf{B}}_a \tag{3.29}$$

with

$$\begin{aligned}\boldsymbol{\Phi}(X) &= \begin{bmatrix} a_1(X)\mathbf{I}_6 & b_1(X)\mathbf{I}_6 & a_2(X)\mathbf{I}_6 & b_2(X)\mathbf{I}_6 & \cdots & a_N(X)\mathbf{I}_6 & b_N(X)\mathbf{I}_6 \end{bmatrix}\boldsymbol{\mathcal{I}} \\ &= \begin{bmatrix} a_1(X)\mathbf{I}_6 & (b_1(X)+a_2(X))\mathbf{I}_6 & b_2(X)\mathbf{I}_6 & \cdots & (b_{N-1}(X)+a_N(X))\mathbf{I}_6 & b_N(X)\mathbf{I}_6 \end{bmatrix}\end{aligned}$$

where $\boldsymbol{\mathcal{I}} \in \mathbb{R}^{12N\times 6(N+1)}$ is the selection matrix (see Appendix B). Substituting (3.29) into (3.27) then arrives at

$$\begin{aligned}\overline{\boldsymbol{F}}_\lambda &= \overline{\boldsymbol{\mathcal{P}}}^{\top}\int_0^L \Big[(\overline{\boldsymbol{J}}^{\top}\mathbf{B}_c\boldsymbol{\lambda})' - \overline{\boldsymbol{J}}'^{\top}\mathbf{B}_c\boldsymbol{\lambda} - \overline{\boldsymbol{J}}^{\top}\operatorname{ad}_{\boldsymbol{\xi}}^{\top}\mathbf{B}_c\boldsymbol{\lambda}\Big]\mathrm{d}X \\ &= \overline{\boldsymbol{\mathcal{P}}}^{\top}\int_0^L \Big[(\overline{\boldsymbol{J}}^{\top}\mathbf{B}_c\boldsymbol{\lambda})' + \overline{\boldsymbol{J}}^{\top}\operatorname{ad}_{\boldsymbol{\xi}}^{\top}\mathbf{B}_c\boldsymbol{\lambda} - \overline{\mathbf{B}}_a^{\top}\boldsymbol{\Phi}^{\top}\mathbf{B}_c\boldsymbol{\lambda} - \overline{\boldsymbol{J}}^{\top}\operatorname{ad}_{\boldsymbol{\xi}}^{\top}\mathbf{B}_c\boldsymbol{\lambda}\Big]\mathrm{d}X \\ &= \overline{\boldsymbol{\mathcal{P}}}^{\top}\int_0^L (\overline{\boldsymbol{J}}^{\top}\mathbf{B}_c\boldsymbol{\lambda})'\mathrm{d}X - \overline{\boldsymbol{\mathcal{P}}}^{\top}\int_0^L \overline{\mathbf{B}}_a^{\top}\boldsymbol{\Phi}^{\top}\mathbf{B}_c\boldsymbol{\lambda}\mathrm{d}X\end{aligned}$$

Clearly, the second item in above equation can be removed in accordance with (3.28). Consequently, by using (3.26b), we obtain

$$\overline{\boldsymbol{F}}_\lambda = \overline{\boldsymbol{\mathcal{P}}}^{\top}(\overline{\boldsymbol{J}}^{\top}\mathbf{B}_c\boldsymbol{\lambda})|_0^{L_N} = \overline{\boldsymbol{\mathcal{P}}}^{\top}\overline{\boldsymbol{J}}^{\top}(L_N)\mathbf{B}_c\mathbf{B}_c^{\top}\big(-\boldsymbol{\mathcal{F}}_{ia}(L_N) + \boldsymbol{\mathcal{F}}_e(L_N)\big)$$

□

By combining (3.25) and (3.26), one can easily obtain the reduced model via PLS Cosserat to model several simplified systems ($n_i \leqslant 6$). An exhaustive reference of the reduced systems with the complementary selection matrices $\mathbf{B}_a$ and $\mathbf{B}_c$ for describing different systems is shown as follows:

- All strain modes but two curvatures on the Y-axis and Z-axis are neglected for Euler-Bernoulli (E-B) beam in 3D space.

$$\mathbf{B}_a = \begin{bmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}, \ \mathbf{B}_c = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}.$$

- For extensible Kirchhoff (E-K) rod in 3D space, bending, twist and extension modes are considered.

$$\mathbf{B}_a = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \ \mathbf{B}_c = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}$$

- For inextensible Kirchhoff (E-K) rod, only the bending and twist modes are considered.

$$\mathbf{B}_a = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \ \mathbf{B}_c = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

- As for Timoshenko beam in 3D space, all modes except twist and extension about X-axis are included.

$$\mathbf{B}_a = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \ \mathbf{B}_c = \begin{bmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

3.4 Simulation Comparison of Discrete Cosserat Models

This section is devoted to validating the precision of the proposed PLS Cosserat models, by comparing it with the result obtained via FEM. Moreover, since PLS Cosserat shares the same local approximation scheme with PCS Cosserat, we will compare as well the precision of PLS model and PCS model. In addition, a contrast of precision of the reduced models via PLS Cosserat static model will be implemented. Besides, the computational efficiency of the discrete Cosserat dynamic model using different numerical schemes (i.e., explicit and semi-implicit Euler algorithms) will be compared.

3.4.1 Simulation Setup

The comparison is effectuated by simulating a cantilever rod under external forces (for example, under gravity). The simulated rod is of conical shape and actuated by four cables (see Fig. 3.3), with total length $L = 0.20$ m, base radius $R_{\max} = 1 \times 10^{-2}$ m, tip radius $R_{\min} = 5 \times 10^{-3}$ m,

Young's modulus $E = 1.1 \times 10^5$ Pa, shear modulus $G = 3.793 \times 10^4$ Pa, and density of material $\rho = 2000$ kg/m^3. $\boldsymbol{\xi}_0 = [0, 0, 0, 1, 0, 0]^\top$ represents the undeformed straight configuration when the rod is stress-free. Besides, the rod shares the X-axis, Y-axis and Z-axis with the inertial frame, and thus the map $\boldsymbol{g}_r$ between the base frame of the rod and the inertial frame is a 4×4 identity matrix.

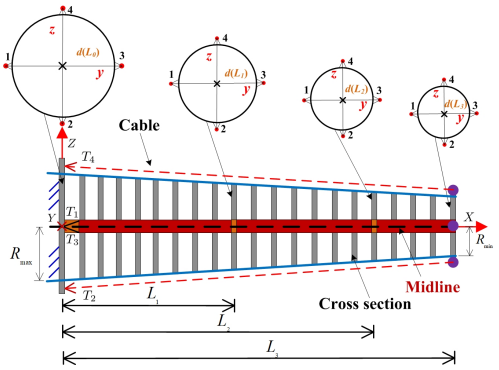


Figure 3.3: The side view of the PLS Cosserat model actuated by four cables.

3.4.2 Comparison of Accuracy for Static Models

FEM can be regarded as a reliable tool (and reference) to verify the modeling result by selecting finer mesh, since it discretizes the space in a very generic manner without introducing restrictive assumptions. That is why we used the FEM approach as an alternative of the real soft arm. In short, the main idea of FEM is to spatially discretize the geometric shape of the rod by using finite number of finer elements. The geometric model of the cantilever rod is established in the SolidWorks, and then we use the FEM in COMSOL which is a general FEM software to obtain its equilibrium position. Specifically, in terms of spatial discretization of the studied rod, quadrilateral mesh elements are used, and the rod is discretized into 650 elements along the X-axis regarding the trade-off between accuracy and computation expense. The mesh average element quality, equals to 0.8033, indicates the high discretization accuracy of the model.

The slender rod modeled by PCS and PLS is divided into three sections, and the length of each section is separately 9×10^{-2} m, 7×10^{-2} m and 4×10^{-2} m from base to tip. For a more specific comparison and evaluation, the material and geometric parameters for the discrete Cosserat models under the same constraints in MATLAB are in accordance with those of the FEM. From the simulation results, the equilibrium position of the end-effector of the cantilever rod via FEM under gravity is $\boldsymbol{p}_e = [5.8479, 0, -17.8395]^\top$, as shown in Fig. 3.4a. It took around 14 seconds to complete one simulation because the FEM for large deformation always requires unnecessary computation. Likewise, we can derive the positions of the end-effector of the discrete Cosserat static models by using the Newton method. From the perspective of computation time of discrete Cosserat static models, we observe that the systems can converge in less than 2 seconds (i.e., 1.3 s for PCS, and 1.2 s for PLS) mainly due to the use of the basic idea of order-model reduction. To intuitively demonstrate the whole shape accuracy of the soft rod modeled by different modeling methods, the spatial configurations of the models under gravity have been depicted in Fig. 3.4c which highlights the high deformation similarity of the PLS and FEM. Comprehensively considering the comparison results among them illustrated in Table 3.1 and Fig. 3.4, we come to a conclusion that the model via the PLS Cosserat modeling approach fits much better with the FEM compared to the PCS, with the relative error of the end-effector

position less than 5%. In other words, the PLS Cosserat static model is essentially comparable to the FEM in terms of accuracy, which can be further verified by the purple PLS configuration tendency plotted in Fig. 3.4b-c, almost same as the FEM in Fig. 3.4a. Logically, this can be explained by the fact that the piecewise linear interpolation technique applied to all cross sections of the model allows the system locally approximate to the deformation behaviors of the slender soft manipulator in the real scenario.

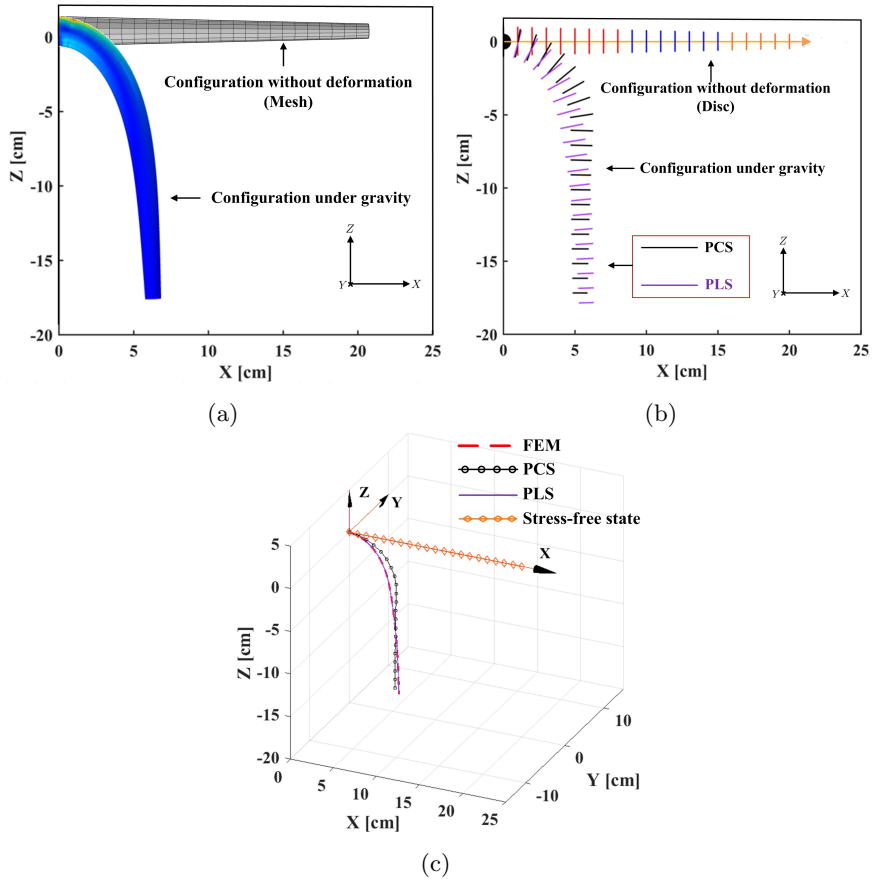


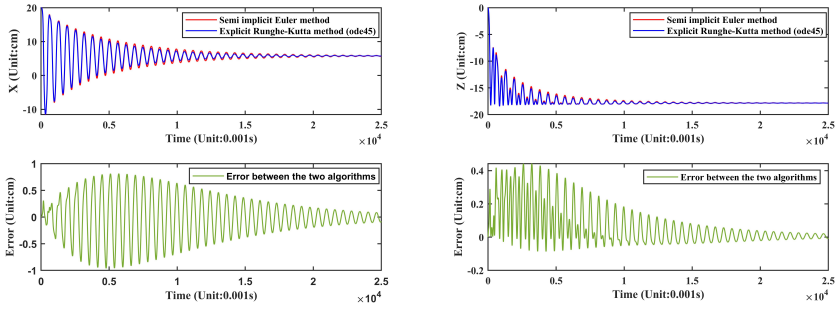
Figure 3.4: Simulation comparison of three different modeling methods for the cantilever rod before and after deformation. (a) FEM in COMSOL. (b) PCS and PLS Cosserat models in MATLAB. (c) Configuration of the soft manipulator under gravity in 3D space.

Table 3.1: Comparison results of FEM, PCS and PLS static Cosserat models under gravity in terms of the end-effector position coordinate relative error and computation time.

Modeling method	The position of the end-effector (Unit:cm)			Position coordinate error w.r.t. the FEM			Computation time
	p_x	p_y	p_z	e_x	e_y	e_z	
FEM	5.8479	0.0000	-17.8395	×	×	×	14 s
PCS [53]	5.3450	0.0000	-17.1693	-8.60%	0.00	-16.88%	1.3 s
PLS	5.7787	0.0000	-17.8394	-1.18%	0.00	-0.01%	1.2 s

3.4.3 Comparison of Computational Efficiency for PLS Cosserat Dynamic Model using Different Numerical Algorithms

In this part, the numerical analysis of continuous Cosserat dynamics will not be discussed in detail, as it has been thoroughly described in [127]. As for the PCS Cosserat dynamic model, the numerical method used was explicit Runghe-Kutta method (using the ode45 function in MATLAB) [53]. To solve the PLS Cosserat dynamic model, a faster computational scheme known as semi-implicit Euler method has been employed, which allows for the selection of larger fixed time step and results in increased computational efficiency during the simulation process.



(a) X-axis. (b) Z-axis.

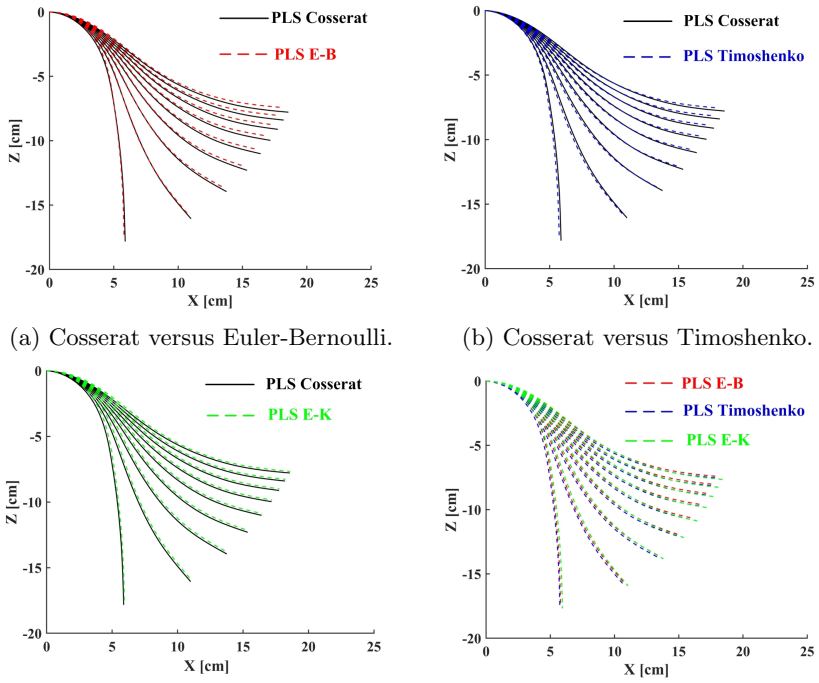
Figure 3.5: Comparison of end-effector position coordinate and its corresponding absolute error using explicit Runghe-Kutta (ode45) and semi-implicit Euler methods.

As far as the computational time is concerned, we make the simulation of the PLS dynamic model only under gravity. The model is numerically solved with four-order Runghe-Kutta scheme requiring around 116 minutes of calculation, and about 1.4×10^6 iterations in order to calculate 25 s of simulation, which is virtually similar compared to the PCS dynamic model solved by using the same numerical algorithm. However, it only takes 21 minutes of calculation, and 25000 time steps of 0.001s each to complete the same time simulation for the implicit Euler method. Fig. 3.5 shows that the variation with the time of the end-effector position along X-axis and Z-axis obtained by using the two numerical algorithms are almost the same, and the absolute error of the end-effector position gradually decreases and fluctuates around $\mathbf{0}$. Besides, the end-effector position asymptotically reaches the equilibrium point $\boldsymbol{p}_e$ as the simulation time approaches 25 s because of the existence of the viscosity modulus. The results mentioned above further elucidate that for the PLS Cosserat dynamics simulation, using the implicit Euler with much less computation time can achieve almost the same accuracy as the explicit Runghe-Kutta method. It is worth noting that computational efficiency is significantly improved owing to the stability of implicit Euler method at a large time step, and this real-time simulation strategy can be effectively applied for the model-based controller design.

3.4.4 Comparison of the Systems with Different Modes

Based on the aforementioned manipulator parameters setting, different models via the PLS Cosserat involving Euler-Bernoulli (E-B), extensible Kirchhoff (E-K) and Timoshenko beams are established by strain mode selection. These beams are fixed at $X = 0$, and subject to gravity as well as an external imposed concentrated load with $\boldsymbol{F}_{\text{tip}} = [0, 0, 0, F, 0, 0]^\top$ (i.e., tension along X-axis) at $X = L_N$ in the inertial frame, we can then convert the concentrated load to the

body frame with $\boldsymbol{\mathcal{F}}_e(L_N) = \mathrm{Ad}^{-1}_{\boldsymbol{g}(L_N)} \boldsymbol{F}_{\mathrm{tip}}$. The static simulation is implemented by increasing load with increment of 0.05 N at a time. Fig. 3.6a-d displays a contrast of the evolution of the equilibrium configurations among these beams for several sets of the tip load, and Table 3.2 shows different beams' end-effector positions versus that of FEM. The results indicate that it is feasible to remove negligible modes in some particular cases with low-precision requirement, which contributes to the real-time simulation and control.



(a) Cosserat versus Euler-Bernoulli.

(b) Cosserat versus Timoshenko.

(c) Cosserat versus Euler-Bernoulli.

(d) E-B, E-K versus Timoshenko.

Figure 3.6: Configuration comparison among different beam models via PLS Cosserat under external tip loads.

Table 3.2: Comparison results of different beam models via PLS Cosserat w.r.t. FEM under gravity in terms of end-effector position and its relative error.

Modeling method	Position of end-effector (Unit:cm)			Relative error of end-effector position w.r.t. FEM (Unit: %)		
	p_x	p_y	p_z	e_x	e_y	e_z
FEM	5.8479	0.0000	-17.8395	×	×	×
Cosserat	5.7787	0.0000	-17.8394	-1.18	0.00	-0.01
E-B	5.4940	0.0000	-17.7527	-6.05	0.00	-0.49
E-K	5.6925	0.0000	-17.8523	-2.66	0.00	0.07
Timoshenko	5.3596	0.0000	-17.9002	-8.35	0.00	0.34

3.5 Model Parameter Identification

The model parameter identification scheme aims to arrive at an appropriate input-output relationship of the model by combining information derived from the experiment with that

obtained from the mechanical behavior of the model. In practice, the exact values of the physical parameters are typically unknown or difficult to derive even for the soft robot manufacturers. Besides, even though there is full knowledge of the model and sufficient data available, an exact description is most often not desirable. Therefore, it is necessary and important to identify the physical parameters for a soft robot conveniently and accurately.

To reach this goal, we present three identification frameworks for the physical parameters of the PLS Cosserat model. The first two methods are for the model with uniform or nonuniform shape under the assumption that the generalized strain vector is measurable. From these two schemes, we can obtain the analytical solutions of the parameters to be identified. The last identification method aims to use the optimization-based approach to identify the parameters of the studied soft manipulator.

3.5.1 Uniform Cosserat Rod

In this part, we will perform the derivation of the material and geometric parameters of the PLS Cosserat model with the columnar shape (i.e., the manipulator is uniform in shape). Subsequently, the identified parameters are validated by the numerical simulation.

3.5.1.1 Algorithm Framework

By setting the generalized joint velocity and acceleration in the dynamic model (3.20) to zero, we can obtain the following model which can be used to achieve model physical parameters identification. It is given by

$$\boldsymbol{\mathcal{P}}^{\top}\int_0^{L_N}\boldsymbol{J}^{\top}\left(\boldsymbol{\mathcal{F}}_{ie}'-\mathrm{ad}_{\boldsymbol{\xi}}^{\top}\boldsymbol{\mathcal{F}}_{ie}\right)\mathrm{d}X+\boldsymbol{\mathcal{P}}^{\top}\int_0^{L_N}\boldsymbol{J}^{\top}\boldsymbol{\mathcal{M}}\mathrm{Ad}_{\boldsymbol{g}(X)}^{-1}\mathrm{Ad}_{\boldsymbol{g}_r}^{-1}\boldsymbol{\mathcal{G}}\mathrm{d}X=-\boldsymbol{H}(\boldsymbol{q})\boldsymbol{T} \tag{3.30a}$$

$$\boldsymbol{\Sigma}(L_N)(\overline{\boldsymbol{\xi}}_N-\boldsymbol{\xi}_{N0})=-\boldsymbol{\Lambda}(L_N)\boldsymbol{T} \tag{3.30b}$$

Substituting (2.16) into (3.30a) yields

$$\boldsymbol{\mathcal{P}}^{\top}\int_0^{L_N}\boldsymbol{J}^{\top}\left[\boldsymbol{\Sigma}(\boldsymbol{\xi}(X)-\boldsymbol{\xi}_0)'-\mathrm{ad}_{\boldsymbol{\xi}}^{\top}\boldsymbol{\Sigma}(\boldsymbol{\xi}(X)-\boldsymbol{\xi}_0)\right]\mathrm{d}X+\boldsymbol{\mathcal{P}}^{\top}\int_0^{L_N}\boldsymbol{J}^{\top}\boldsymbol{\mathcal{M}}\boldsymbol{\mathcal{U}}(\boldsymbol{q})\mathrm{d}X=-\boldsymbol{H}(\boldsymbol{q})\boldsymbol{T} \tag{3.31}$$

with $\boldsymbol{\mathcal{U}}(\boldsymbol{q})=\mathrm{Ad}_{\boldsymbol{g}(X)}^{-1}\mathrm{Ad}_{\boldsymbol{g}_r}^{-1}\boldsymbol{\mathcal{G}}\in\mathbb{R}^6$, where $\boldsymbol{\Sigma}$ and $\boldsymbol{\mathcal{M}}$ are constant diagonal matrices. Considering the correlation of elements of the stiffness matrix (i.e., $J_x=J_y+J_z$), the number of parameters to be identified can be then reduced. Therefore, the vector $\overline{\boldsymbol{\Sigma}}=(GJ_x,EJ_y,EJ_z,EA,GA)^{\top}$ composed of elements from the screw stiffness matrix is defined as the identified quantity. Substituting the PLS assumption into the generalized elastic wrench in (3.31) arrives at

$$\begin{aligned}
&\boldsymbol{\mathcal{P}}^{\top}\int_0^{L_N}\boldsymbol{J}^{\top}\left[\mathrm{diag}(\boldsymbol{\xi}(X)-\boldsymbol{\xi}_0)'-\mathrm{ad}_{\boldsymbol{\xi}}^{\top}\left(\mathrm{diag}(\boldsymbol{\xi}(X)-\boldsymbol{\xi}_0)\right)\right]\mathrm{d}X\boldsymbol{\mathcal{I}}_1\overline{\boldsymbol{\Sigma}}\\
&=\boldsymbol{\mathcal{P}}^{\top}\sum_{n=1}^{N}\int_{L_{n-1}}^{L_n}\boldsymbol{J}^{\top}\mathrm{diag}\left[(\overline{\boldsymbol{\xi}}_{n-1}-\boldsymbol{\xi}_{(n-1)0})a(X)'+(\overline{\boldsymbol{\xi}}_n-\boldsymbol{\xi}_{n0})b(X)'\right]\mathrm{d}X\boldsymbol{\mathcal{I}}_1\overline{\boldsymbol{\Sigma}}\\
&\quad-\boldsymbol{\mathcal{P}}^{\top}\sum_{n=1}^{N}\int_{L_{n-1}}^{L_n}\boldsymbol{J}^{\top}\mathrm{ad}_{\boldsymbol{\xi}_n}^{\top}\mathrm{diag}\left[(\overline{\boldsymbol{\xi}}_{n-1}-\boldsymbol{\xi}_{(n-1)0})a(X)+(\overline{\boldsymbol{\xi}}_n-\boldsymbol{\xi}_{n0})b(X)\right]\mathrm{d}X\boldsymbol{\mathcal{I}}_1\overline{\boldsymbol{\Sigma}}\\
&=\boldsymbol{\mathcal{P}}^{\top}\sum_{n=1}^{N}\int_{L_{n-1}}^{L_n}\boldsymbol{J}^{\top}\mathrm{diag}\left[\begin{pmatrix}a(X)'\mathbf{I}_6 & b(X)'\mathbf{I}_6\end{pmatrix}\begin{pmatrix}\overline{\boldsymbol{\xi}}_{n-1}-\boldsymbol{\xi}_{(n-1)0}\\ \overline{\boldsymbol{\xi}}_n-\boldsymbol{\xi}_{n0}\end{pmatrix}\right]\mathrm{d}X\boldsymbol{\mathcal{I}}_1\overline{\boldsymbol{\Sigma}}\\
&\quad-\boldsymbol{\mathcal{P}}^{\top}\sum_{n=1}^{N}\int_{L_{n-1}}^{L_n}\boldsymbol{J}^{\top}\mathrm{ad}_{\boldsymbol{\xi}_n}^{\top}\mathrm{diag}\left[\begin{pmatrix}a(X)\mathbf{I}_6 & b(X)\mathbf{I}_6\end{pmatrix}\begin{pmatrix}\overline{\boldsymbol{\xi}}_{n-1}-\boldsymbol{\xi}_{(n-1)0}\\ \overline{\boldsymbol{\xi}}_n-\boldsymbol{\xi}_{n0}\end{pmatrix}\right]\mathrm{d}X\boldsymbol{\mathcal{I}}_1\overline{\boldsymbol{\Sigma}}\\
&=\boldsymbol{A}_1(\boldsymbol{q})\boldsymbol{\mathcal{I}}_1\overline{\boldsymbol{\Sigma}}
\end{aligned}$$

with $a_n(X) = \frac{L_n - X}{L_n - L_{n-1}}$, $b_n(X) = \frac{X - L_{n-1}}{L_n - L_{n-1}}$, and

$$\boldsymbol{\mathcal{I}}_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{6\times 5}$$

where diag$(\cdot)$ returns a square diagonal matrix with the elements of the vector on the main diagonal. For the same reason as the screw stiffness matrix, the vector $\overline{\boldsymbol{\mathcal{M}}} = (\rho J_y, \rho J_z, \rho A)^\top$ consisted of elements from the mass matrix is defined as the identified quantity. Therefore, the equation (3.30) is then equivalent to

$$\underbrace{\begin{bmatrix} \boldsymbol{A}_1(\boldsymbol{q}) \\ \mathrm{diag}(\overline{\boldsymbol{\xi}}_N - \boldsymbol{\xi}_{N0}) \end{bmatrix} \boldsymbol{\mathcal{I}}_1}_{\boldsymbol{A}(\boldsymbol{q})} \overline{\boldsymbol{\Sigma}} + \underbrace{\begin{bmatrix} \boldsymbol{B}_1(\boldsymbol{q}) \\ \boldsymbol{0}_{6\times 6} \end{bmatrix} \boldsymbol{\mathcal{I}}_2}_{\boldsymbol{B}(\boldsymbol{q})} \overline{\boldsymbol{\mathcal{M}}} = -\underbrace{\begin{bmatrix} \boldsymbol{H}(\boldsymbol{q}) \\ \boldsymbol{\Lambda}(L_N) \end{bmatrix}}_{\boldsymbol{\mathcal{H}}(\boldsymbol{q})} \boldsymbol{T}$$

with $\boldsymbol{B}_1(\boldsymbol{q}) = \boldsymbol{\mathcal{P}}^\top \int_0^{L_N} \boldsymbol{J}^\top \mathrm{diag}(\boldsymbol{\mathcal{U}}(\boldsymbol{q}))\mathrm{d}X$, and

$$\boldsymbol{\mathcal{I}}_2 = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix} \in \mathbb{R}^{6\times 3}$$

For each test, we assume that the strain vector $\boldsymbol{q}$ is measurable. By implementing multiple tests, the physical parameters of the model can be then obtained. In order to determine the identified parameters, $\overline{N}$ sets of tests are performed, and this amounts to the following expression:

$$\begin{bmatrix} \boldsymbol{A}(\boldsymbol{q}_1) & \boldsymbol{B}(\boldsymbol{q}_1) \\ \boldsymbol{A}(\boldsymbol{q}_2) & \boldsymbol{B}(\boldsymbol{q}_2) \\ \vdots & \vdots \\ \boldsymbol{A}(\boldsymbol{q}_{\overline{N}}) & \boldsymbol{B}(\boldsymbol{q}_{\overline{N}}) \end{bmatrix} \boldsymbol{\theta} = - \begin{bmatrix} \boldsymbol{\mathcal{H}}(\boldsymbol{q}_1)\boldsymbol{T}_1 \\ \boldsymbol{\mathcal{H}}(\boldsymbol{q}_2)\boldsymbol{T}_2 \\ \vdots \\ \boldsymbol{\mathcal{H}}(\boldsymbol{q}_{\overline{N}})\boldsymbol{T}_{\overline{N}} \end{bmatrix} \tag{3.32}$$

with $\boldsymbol{\theta} = \begin{bmatrix} \overline{\boldsymbol{\Sigma}}^\top & \overline{\boldsymbol{\mathcal{M}}}^\top \end{bmatrix}^\top \in \mathbb{R}^8$. According to the least square method (LSM), the physical parameters estimation $\widehat{\boldsymbol{\theta}}$ of the PLS Cosserat model can be analytically derived.

$$\widehat{\boldsymbol{\theta}} = (\boldsymbol{\mathcal{K}}^\top \boldsymbol{\mathcal{K}})^{-1} \boldsymbol{\mathcal{K}}^\top \boldsymbol{\mathcal{C}} \tag{3.33}$$

with

$$\boldsymbol{\mathcal{K}} = \begin{bmatrix} \boldsymbol{A}(\boldsymbol{q}_1) & \boldsymbol{B}(\boldsymbol{q}_1) \\ \boldsymbol{A}(\boldsymbol{q}_2) & \boldsymbol{B}(\boldsymbol{q}_2) \\ \vdots & \vdots \\ \boldsymbol{A}(\boldsymbol{q}_{\overline{N}}) & \boldsymbol{B}(\boldsymbol{q}_{\overline{N}}) \end{bmatrix}, \ \boldsymbol{\mathcal{C}} = - \begin{bmatrix} \boldsymbol{\mathcal{H}}(\boldsymbol{q}_1)\boldsymbol{T}_1 \\ \boldsymbol{\mathcal{H}}(\boldsymbol{q}_2)\boldsymbol{T}_2 \\ \vdots \\ \boldsymbol{\mathcal{H}}(\boldsymbol{q}_{\overline{N}})\boldsymbol{T}_{\overline{N}} \end{bmatrix}$$

Finally, the geometric and material parameters of the PLS Cosserat model can be formulated as

$$\widehat{\boldsymbol{\Sigma}} = \mathrm{diag}(\boldsymbol{\mathcal{I}}_1 \widehat{\overline{\boldsymbol{\Sigma}}})$$
$$\widehat{\boldsymbol{\mathcal{M}}} = \mathrm{diag}(\boldsymbol{\mathcal{I}}_2 \widehat{\overline{\boldsymbol{M}}})$$

3.5.1.2 Simulation Validation

The slender soft manipulator modeled by PLS Cosserat is divided into three sections, and actuated by four cables. The geometric and material parameters of the PLS Cosserat model in MATLAB is as follows: total length $L = 0.20$ m, cross-sectional radius $R = 1.5 \times 10^{-2}$ m, Young's modulus $E = 1.1 \times 10^5$ Pa, shear modulus $G = 3.793 \times 10^4$ Pa, and material density $\rho = 2000$ kg/m^3. By implementing 5 sets of different tests, we can determine the stiffness and mass matrices of the model. As illustrated from Table 3.3, the relative error of identification results of the model parameters is within $\pm 0.3\%$.

Table 3.3: The parameter identification results of the model with columnar shape.

Parameters to be identified	Real value	Estimation	Unit	Relative error(%)
GJ_x	3016.3352	3015.8936	$10^{-6}\text{Pa} \cdot \text{m}^4$	-0.03
EJ_y	4373.6860	4372.3236	$10^{-6}\text{Pa} \cdot \text{m}^4$	-0.02
EJ_z	4373.6860	4373.3741	$10^{-6}\text{Pa} \cdot \text{m}^4$	-0.01
EA	7775.4418	7774.5101	$10^{-2}\text{Pa} \cdot \text{m}^2$	-0.01
GA	2681.1868	2680.9676	$10^{-2}\text{Pa} \cdot \text{m}^2$	-0.01
GA	2681.1868	2680.9676	$10^{-2}\text{Pa} \cdot \text{m}^2$	-0.01
ρJ_x	0.0160	$\times$	$10^{-2}\text{kg} \cdot \text{m}$	$\times$
ρJ_y	0.0080	$\times$	$10^{-2}\text{kg} \cdot \text{m}$	$\times$
ρJ_z	0.0080	$\times$	$10^{-2}\text{kg} \cdot \text{m}$	$\times$
ρA	1.4137	1.4134	kg/m	-0.21

From Table 3.3, we can observe that the estimation algorithm cannot identify the parameters including ρJ_x, ρJ_y and ρJ_z. By analysis and validation, these parameters are unnecessary for parameter estimation of the PLS Cosserat static model, which is mainly due to the model only under gravity not presenting torque.

3.5.2 Nonuniform Cosserat Rod

In engineering, the shape of the most soft robots is not always uniform, and the PLS Cosserat approach has taken fully into account the variation with X of the stiffness matrix $\mathbf{\Sigma}(X)$ and mass matrix $\boldsymbol{\mathcal{M}}(X)$. The approximate functions of geometric parameters on the arc length X can be obtained by employing several methods such as the N-order polynomial, radial basis function (RBF), etc.

In an effort to determine the geometric and material parameters of the PLS Cosserat model with conical shape, supposing that Young's modulus E, density of material ρ and shear modulus G of the rod are always constant along the soft manipulator, we can then apply the polynomials to approximate the parameters (i.e., the second moment of area tensor $\boldsymbol{\mathcal{J}}(X)$ and cross-sectional area $A(X)$).

3.5.2.1 Algorithm Framework

In the light of the relation between the area moment of inertia and radius as well as that between cross-sectional area and radius, the different order of polynomials are used to formulate the second moment of area tensor and the cross-sectional area, and they become

$$
\begin{aligned}
J_y(X) &= a_0 + a_1 X + a_2 X^2 + a_3 X^3 + a_4 X^4 \\
J_z(X) &= b_0 + b_1 X + b_2 X^2 + b_3 X^3 + b_4 X^4 \\
J_x(X) &= (a_0 + b_0) + (a_1 + b_1) X + (a_2 + b_2) X^2 + (a_3 + b_3) X^3 + (a_4 + b_4) X^4 \\
A(X) &= d_0 + d_1 X + d_2 X^2
\end{aligned}
\tag{3.34}
$$

where $J_y(X)$ and $J_z(X)$ refer to the second moments of the area of the cross section with respect to Y-axis and Z-axis, and $J_x(X)$ is the polar moment of the area around the X-axis, equal to $J_x(X) = J_y(X) + J_z(X)$. Thus, the stiffness matrix $\boldsymbol{\Sigma}(X)$ and mass matrix $\boldsymbol{\mathcal{M}}(X)$ can be then formulated as

$$
\boldsymbol{\Sigma}(X) = \overline{\boldsymbol{K}} \begin{bmatrix} G(\boldsymbol{\lambda}_1 + \boldsymbol{\lambda}_2) & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & E\boldsymbol{\lambda}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & E\boldsymbol{\lambda}_2 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & E\boldsymbol{\lambda}_3 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & G\boldsymbol{\lambda}_3 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & G\boldsymbol{\lambda}_3 \end{bmatrix} = \overline{\boldsymbol{K}}(X)\boldsymbol{P}
$$

$$
\boldsymbol{\mathcal{M}}(X) = \overline{\boldsymbol{K}} \begin{bmatrix} \rho(\boldsymbol{\lambda}_1 + \boldsymbol{\lambda}_2) & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \rho\boldsymbol{\lambda}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \rho\boldsymbol{\lambda}_2 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \rho\boldsymbol{\lambda}_3 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \rho\boldsymbol{\lambda}_3 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \rho\boldsymbol{\lambda}_3 \end{bmatrix} = \overline{\boldsymbol{K}}(X)\boldsymbol{Q}
$$

with $\overline{\boldsymbol{K}}(X) = \mathbf{I}_6 \otimes \boldsymbol{k} \in \mathbb{R}^{6\times 30}$, $\boldsymbol{\lambda}_1 = \begin{bmatrix} a_0 & a_1 & a_2 & a_3 & a_4 \end{bmatrix}^\top \in \mathbb{R}^5$, $\boldsymbol{\lambda}_2 = \begin{bmatrix} b_0 & b_1 & b_2 & b_3 & b_4 \end{bmatrix}^\top \in \mathbb{R}^5$. $\boldsymbol{\lambda}_3 = \begin{bmatrix} d_0 & d_1 & d_2 & 0 & 0 \end{bmatrix}^\top \in \mathbb{R}^5$, and $\boldsymbol{\lambda}_3 = \boldsymbol{\Gamma}\overline{\boldsymbol{\lambda}}_3$ with

$$
\boldsymbol{\Gamma} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \in \mathbb{R}^{5\times 3}, \quad \overline{\boldsymbol{\lambda}}_3 = \begin{bmatrix} d_0 \\ d_1 \\ d_2 \end{bmatrix}
$$

where $\boldsymbol{k}$ is a row vector, equal to $\boldsymbol{k} = \begin{bmatrix} 1 & X & X^2 & X^3 & X^4 \end{bmatrix} \in \mathbb{R}^{1\times 5}$, and the symbol $\otimes$ represents the Kronecker tensor product.

Substituting (2.16) into (3.30a) yields

$$
\begin{aligned}
&\boldsymbol{\mathcal{P}}^\top \int_0^{L_N} \boldsymbol{J}^\top \Big[\boldsymbol{\Sigma}(X)(\boldsymbol{\xi}(X) - \boldsymbol{\xi}_0)' + \boldsymbol{\Sigma}'(X)(\boldsymbol{\xi}(X) - \boldsymbol{\xi}_0) - \mathrm{ad}_{\boldsymbol{\xi}}^\top \boldsymbol{\Sigma}(X)(\boldsymbol{\xi}(X) - \boldsymbol{\xi}_0) \Big] \mathrm{d}X + \cdots \\
&\cdots + \boldsymbol{\mathcal{P}}^\top \int_0^{L_N} \boldsymbol{J}^\top \boldsymbol{\mathcal{M}}(X) \boldsymbol{\mathcal{U}}(\boldsymbol{q}) \mathrm{d}X = -\boldsymbol{H}(\boldsymbol{q})\boldsymbol{T}
\end{aligned}
\tag{3.35}
$$

with

$$
\boldsymbol{\Sigma}'(X) = \overline{\boldsymbol{K}}_d \begin{bmatrix} G(\boldsymbol{\lambda}_1 + \boldsymbol{\lambda}_2) & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & E\boldsymbol{\lambda}_1 & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & E\boldsymbol{\lambda}_2 & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & E\boldsymbol{\lambda}_3 & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & G\boldsymbol{\lambda}_3 & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & G\boldsymbol{\lambda}_3 \end{bmatrix} = \overline{\boldsymbol{K}}_d(X)\boldsymbol{P}
$$

where $\overline{\boldsymbol{K}}_d(X) = \mathbf{I}_6 \otimes \boldsymbol{k}_d \in \mathbb{R}^{6\times 30}$ with the row vector $\boldsymbol{k}_d = \begin{bmatrix}0 & 1 & 2X & 3X^2 & 4X^3\end{bmatrix}$.

It can be clearly seen that the screw stiffness matrix $\boldsymbol{\Sigma}(X)$ and its space derivative can be decomposed into the product of the matrix function of X and a constant matrix to be identified. Due to the correlation of elements of the stiffness matrix, the number of parameters to be identified can be reduced. Therefore, we can define the vector $\overline{\boldsymbol{P}} = (G(\boldsymbol{\lambda}_1+\boldsymbol{\lambda}_2)^\top, E\boldsymbol{\lambda}_1^\top, E\boldsymbol{\lambda}_2^\top, E\overline{\boldsymbol{\lambda}}_3^\top, G\overline{\boldsymbol{\lambda}}_3^\top)^\top$ composed of elements from the constant matrix $\boldsymbol{P}$ as the quantity to be identified. Then, plugging the stiffness matrix and its space derivative into the internal wrench in (3.35), we have

$$
\begin{aligned}
&\boldsymbol{\mathcal{P}}^\top \int_0^{L_N} \boldsymbol{J}^\top \Big[\overline{\boldsymbol{K}}\Big(\operatorname{diag}(\boldsymbol{\xi}(X)-\boldsymbol{\xi}_0)'\otimes \mathbf{I}_5\Big) + \overline{\boldsymbol{K}}_d\Big(\operatorname{diag}(\boldsymbol{\xi}(X)-\boldsymbol{\xi}_0)\otimes \mathbf{I}_5\Big) + \cdots \\
&\quad \cdots - \operatorname{ad}_{\boldsymbol{\xi}}^\top\Big(\overline{\boldsymbol{K}}\Big(\operatorname{diag}(\boldsymbol{\xi}(X)-\boldsymbol{\xi}_0)\otimes \mathbf{I}_5\Big)\Big)\Big]\mathrm{d}X\boldsymbol{\mathcal{I}}_1\boldsymbol{\Lambda}_1\overline{\boldsymbol{P}} \\
&= \boldsymbol{\mathcal{P}}^\top \sum_{n=1}^{N}\int_{L_{n-1}}^{L_n} \boldsymbol{J}^{\mathrm{T}}\overline{\boldsymbol{K}}\Big\{\operatorname{diag}\Big[\begin{pmatrix}a(X)'\mathbf{I}_6 & b(X)'\mathbf{I}_6\end{pmatrix}\begin{pmatrix}\overline{\boldsymbol{\xi}}_{n-1}-\boldsymbol{\xi}_{(n-1)0}\\ \overline{\boldsymbol{\xi}}_n-\boldsymbol{\xi}_{n0}\end{pmatrix}\Big]\otimes\mathbf{I}_5\Big\}\mathrm{d}X\boldsymbol{\mathcal{I}}_1\boldsymbol{\Lambda}_1\overline{\boldsymbol{P}} \\
&\quad + \boldsymbol{\mathcal{P}}^\top \sum_{n=1}^{N}\int_{L_{n-1}}^{L_n} \boldsymbol{J}^\top\Big(\overline{\boldsymbol{K}}_d - \operatorname{ad}_{\boldsymbol{\xi}_n}^\top\overline{\boldsymbol{K}}\Big)\Big\{\operatorname{diag}\Big[\begin{pmatrix}a(X)\mathbf{I}_6 & b(X)\mathbf{I}_6\end{pmatrix}\begin{pmatrix}\overline{\boldsymbol{\xi}}_{n-1}-\boldsymbol{\xi}_{(n-1)0}\\ \overline{\boldsymbol{\xi}}_n-\boldsymbol{\xi}_{n0}\end{pmatrix}\Big]\otimes\mathbf{I}_5\Big\}\mathrm{d}X\boldsymbol{\mathcal{I}}_1\boldsymbol{\Lambda}_1\overline{\boldsymbol{P}} \\
&= \boldsymbol{A}_1(\boldsymbol{q})\boldsymbol{\mathcal{I}}_1\boldsymbol{\Lambda}_1\overline{\boldsymbol{P}}
\end{aligned}
$$

with

$$
\boldsymbol{\mathcal{I}}_1 = \begin{bmatrix}
\mathbf{I}_5 & \mathbf{0}_5 & \mathbf{0}_5 & \mathbf{0}_5 & \mathbf{0}_5\\
\mathbf{0}_5 & \mathbf{I}_5 & \mathbf{0}_5 & \mathbf{0}_5 & \mathbf{0}_5\\
\mathbf{0}_5 & \mathbf{0}_5 & \mathbf{I}_5 & \mathbf{0}_5 & \mathbf{0}_5\\
\mathbf{0}_5 & \mathbf{0}_5 & \mathbf{0}_5 & \mathbf{I}_5 & \mathbf{0}_5\\
\mathbf{0}_5 & \mathbf{0}_5 & \mathbf{0}_5 & \mathbf{0}_5 & \mathbf{I}_5\\
\mathbf{0}_5 & \mathbf{0}_5 & \mathbf{0}_5 & \mathbf{0}_5 & \mathbf{I}_5
\end{bmatrix} \in \mathbb{R}^{30\times 25},\ \boldsymbol{\Lambda}_1 = \begin{bmatrix}
\mathbf{I}_5 & \mathbf{0}_5 & \mathbf{0}_5 & \mathbf{0}_{5\times 3} & \mathbf{0}_{5\times 3}\\
\mathbf{0}_5 & \mathbf{I}_5 & \mathbf{0}_5 & \mathbf{0}_{5\times 3} & \mathbf{0}_{5\times 3}\\
\mathbf{0}_5 & \mathbf{0}_5 & \mathbf{I}_5 & \mathbf{0}_{5\times 3} & \mathbf{0}_{5\times 3}\\
\mathbf{0}_5 & \mathbf{0}_5 & \mathbf{0}_5 & \boldsymbol{\Gamma}_{5\times 3} & \mathbf{0}_{5\times 3}\\
\mathbf{0}_5 & \mathbf{0}_5 & \mathbf{0}_5 & \mathbf{0}_{5\times 3} & \boldsymbol{\Gamma}_{5\times 3}
\end{bmatrix} \in \mathbb{R}^{25\times 21}
$$

For the same reason as the stiffness matrix, the vector $\overline{\boldsymbol{Q}} = (\rho\boldsymbol{\lambda}_1^\top, \rho\boldsymbol{\lambda}_2^\top, \rho\overline{\boldsymbol{\lambda}}_3^\top)^\top$ composed of elements from the constant matrix $\boldsymbol{Q}$ is defined as the identified quantity. In this end, the equation (3.30) arrives at

$$
\underbrace{\begin{bmatrix}\boldsymbol{A}_1(\boldsymbol{q})\\ \overline{\boldsymbol{K}}(L_N)\operatorname{diag}(\overline{\boldsymbol{\xi}}_N-\boldsymbol{\xi}_{N0})\otimes\mathbf{I}_5\end{bmatrix}\boldsymbol{\mathcal{I}}_1\boldsymbol{\Lambda}_1}_{\boldsymbol{A}(\boldsymbol{q})}\overline{\boldsymbol{P}} + \underbrace{\begin{bmatrix}\boldsymbol{B}_1(\boldsymbol{q})\\ \mathbf{0}_{6\times 30}\end{bmatrix}\boldsymbol{\mathcal{I}}_2\boldsymbol{\Lambda}_2}_{\boldsymbol{B}(\boldsymbol{q})}\overline{\boldsymbol{Q}} = -\underbrace{\begin{bmatrix}\boldsymbol{H}(\boldsymbol{q})\\ \boldsymbol{\Lambda}(L_N)\end{bmatrix}}_{\boldsymbol{\mathcal{H}}(\boldsymbol{q})}\boldsymbol{T}
$$

with $\boldsymbol{B}_1(\boldsymbol{q}) = \boldsymbol{\mathcal{P}}^\top\int_0^{L_N}\boldsymbol{J}^\top\overline{\boldsymbol{K}}\left[\operatorname{diag}(\boldsymbol{\mathcal{U}}(\boldsymbol{q}))\otimes\mathbf{I}_5\right]\mathrm{d}X$, and

$$
\boldsymbol{\mathcal{I}}_2 = \begin{bmatrix}
\mathbf{I}_5 & \mathbf{I}_5 & \mathbf{0}_5\\
\mathbf{I}_5 & \mathbf{0}_5 & \mathbf{0}_5\\
\mathbf{0}_5 & \mathbf{I}_5 & \mathbf{0}_5\\
\mathbf{0}_5 & \mathbf{0}_5 & \mathbf{I}_5\\
\mathbf{0}_5 & \mathbf{0}_5 & \mathbf{I}_5\\
\mathbf{0}_5 & \mathbf{0}_5 & \mathbf{I}_5
\end{bmatrix} \in \mathbb{R}^{30\times 15},\ \boldsymbol{\Lambda}_2 = \begin{bmatrix}
\mathbf{I}_5 & \mathbf{0}_5 & \mathbf{0}_{5\times 3}\\
\mathbf{0}_5 & \mathbf{I}_5 & \mathbf{0}_{5\times 3}\\
\mathbf{0}_5 & \mathbf{0}_5 & \boldsymbol{\Gamma}_{5\times 3}
\end{bmatrix} \in \mathbb{R}^{15\times 13}
$$

Here, we suppose that the generalized strain vector $\boldsymbol{q}_i$ of the soft manipulator driven by cables for the ith test is measurable. Selecting multiple testing samples, the geometric and material parameters can be then determined. In order to obtain the identified parameters, a set of $\overline{N}$ pairs of observations is used to find a function relating the value of the dependent variable to the value of independent variable. Based on the selected training samples, the regression

equation can be formulated as

$$\begin{bmatrix} \boldsymbol{A}(\boldsymbol{q}_1) & \boldsymbol{B}(\boldsymbol{q}_1) \\ \boldsymbol{A}(\boldsymbol{q}_2) & \boldsymbol{B}(\boldsymbol{q}_2) \\ \vdots & \vdots \\ \boldsymbol{A}(\boldsymbol{q}_{\overline{N}}) & \boldsymbol{B}(\boldsymbol{q}_{\overline{N}}) \end{bmatrix} \boldsymbol{\theta} = - \begin{bmatrix} \boldsymbol{\mathcal{H}}(\boldsymbol{q}_1)\boldsymbol{T}_1 \\ \boldsymbol{\mathcal{H}}(\boldsymbol{q}_2)\boldsymbol{T}_2 \\ \vdots \\ \boldsymbol{\mathcal{H}}(\boldsymbol{q}_{\overline{N}})\boldsymbol{T}_{\overline{N}} \end{bmatrix} \tag{3.36}$$

with $\boldsymbol{\theta} = \begin{bmatrix} \overline{\boldsymbol{P}}^\top & \overline{\boldsymbol{Q}}^\top \end{bmatrix}^\top \in \mathbb{R}^{34}$. The least square method (LSM) defines the parameters identification as the value that minimizes the sum of squared errors between the observations and their counterparts computed by the parameterized model for all tests, which is equivalent to

$$\begin{aligned} \mathrm{J}(\widehat{\boldsymbol{\theta}}) &= \frac{1}{2}\sum_{i=1}^{\overline{N}} (\boldsymbol{\Psi}_i - \boldsymbol{\mathcal{K}}_i\widehat{\boldsymbol{\theta}})^2 = \frac{1}{2}(\boldsymbol{\Psi} - \boldsymbol{\mathcal{K}}\widehat{\boldsymbol{\theta}})^\top(\boldsymbol{\Psi} - \boldsymbol{\mathcal{K}}\widehat{\boldsymbol{\theta}}) \\ &= \frac{1}{2}(\boldsymbol{\Psi}^\top\boldsymbol{\Psi} - \widehat{\boldsymbol{\theta}}^\top\boldsymbol{\mathcal{K}}^\top\boldsymbol{\Psi} - \boldsymbol{\Psi}^\top\boldsymbol{\mathcal{K}}\widehat{\boldsymbol{\theta}} + \widehat{\boldsymbol{\theta}}^\top\boldsymbol{\mathcal{K}}^\top\boldsymbol{\mathcal{K}}\widehat{\boldsymbol{\theta}}) \end{aligned} \tag{3.37}$$

with

$$\boldsymbol{\mathcal{K}} = \begin{bmatrix} \boldsymbol{\mathcal{K}}_1 \\ \boldsymbol{\mathcal{K}}_2 \\ \vdots \\ \boldsymbol{\mathcal{K}}_{\overline{N}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{A}(\boldsymbol{q}_1) & \boldsymbol{B}(\boldsymbol{q}_1) \\ \boldsymbol{A}(\boldsymbol{q}_2) & \boldsymbol{B}(\boldsymbol{q}_2) \\ \vdots & \vdots \\ \boldsymbol{A}(\boldsymbol{q}_{\overline{N}}) & \boldsymbol{B}(\boldsymbol{q}_{\overline{N}}) \end{bmatrix}, \ \boldsymbol{\Psi} = \begin{bmatrix} \boldsymbol{\Psi}_1 \\ \boldsymbol{\Psi}_2 \\ \vdots \\ \boldsymbol{\Psi}_{\overline{N}} \end{bmatrix} = - \begin{bmatrix} \boldsymbol{\mathcal{H}}(\boldsymbol{q}_1)\boldsymbol{T}_1 \\ \boldsymbol{\mathcal{H}}(\boldsymbol{q}_2)\boldsymbol{T}_2 \\ \vdots \\ \boldsymbol{\mathcal{H}}(\boldsymbol{q}_{\overline{N}})\boldsymbol{T}_{\overline{N}} \end{bmatrix}$$

where $\mathrm{J}(\widehat{\boldsymbol{\theta}})$ represents the sum of squared errors which is the quantity to be minimized. Taking the partial derivative of $\mathrm{J}(\widehat{\boldsymbol{\theta}})$ with respect to $\widehat{\boldsymbol{\theta}}$, we have

$$\frac{\partial \mathrm{J}(\widehat{\boldsymbol{\theta}})}{\partial \widehat{\boldsymbol{\theta}}} = \frac{1}{2}(-2\boldsymbol{\mathcal{K}}^\top\boldsymbol{\Psi} + 2\boldsymbol{\mathcal{K}}^\top\boldsymbol{\mathcal{K}}\widehat{\boldsymbol{\theta}})$$

Therefore, the geometric and material parameters estimation $\widehat{\boldsymbol{\theta}}$ of the PLS Cosserat model with the nonuniform shape can be analytically derived when $\boldsymbol{\mathcal{K}}^\top\boldsymbol{\mathcal{K}}$ is non-singular.

$$\widehat{\boldsymbol{\theta}} = \begin{bmatrix} \widehat{\overline{\boldsymbol{P}}}^\top & \widehat{\overline{\boldsymbol{Q}}}^\top \end{bmatrix}^\top = (\boldsymbol{\mathcal{K}}^\top\boldsymbol{\mathcal{K}})^{-1}\boldsymbol{\mathcal{K}}^\top\boldsymbol{\Psi} \tag{3.38}$$

From (3.38), we have

$$\widehat{\boldsymbol{P}} = \mathrm{diag}(\boldsymbol{\mathcal{I}}_1\boldsymbol{\Lambda}_1\widehat{\overline{\boldsymbol{P}}})(\mathbf{I}_6 \otimes \mathbf{E}), \quad \widehat{\boldsymbol{Q}} = \mathrm{diag}(\boldsymbol{\mathcal{I}}_2\boldsymbol{\Lambda}_2\widehat{\overline{\boldsymbol{Q}}})(\mathbf{I}_6 \otimes \mathbf{E})$$

with the column vector

$$\mathbf{E} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 \end{bmatrix}^\top \in \mathbb{R}^5$$

Finally, we can derive the estimation of the stiffness and mass matrices as follows

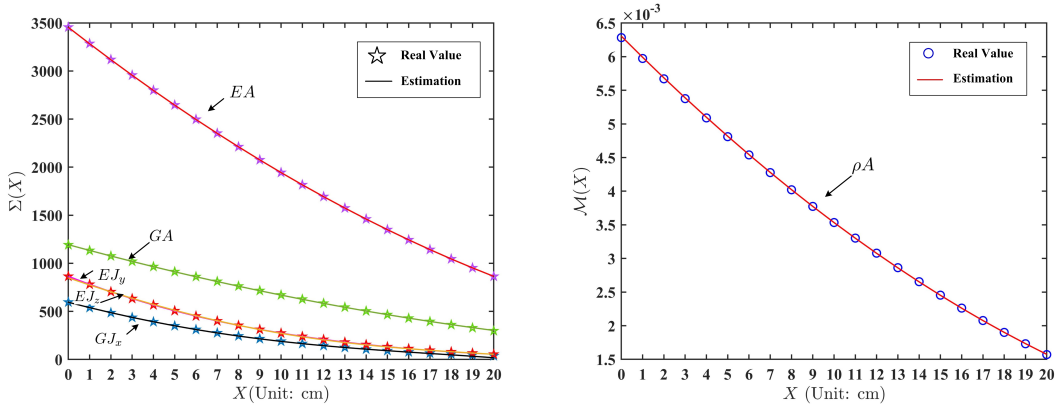
$$\widehat{\boldsymbol{\Sigma}}(X) = \overline{\boldsymbol{K}}(X)\widehat{\boldsymbol{P}}$$

$$\widehat{\boldsymbol{\mathcal{M}}}(X) = \overline{\boldsymbol{K}}(X)\widehat{\boldsymbol{Q}}$$

3.5.2.2 Simulation and Verification

The PLS Cosserat model with conical shape is divided into three sections, resulting in 24 linear equations if implementing one test. However, there are 34 parameters to be identified for the system. Thus, at least 2 sets of tests need to be performed in order to achieve the parameter identification algorithm. The simulated rod has the same parameter settings as those in Section 3.4.1, which are total length $L = 0.20$ m, base radius $R_{\max} = 1 \times 10^{-2}$ m, tip radius $R_{\min} = 5 \times 10^{-3}$ m, Young's modulus $E = 1.1 \times 10^5$ Pa, shear modulus $G = 3.793 \times 10^4$ Pa, and density of material $\rho = 2000$ kg/m^3.

Selecting 5 sets of tests involving $\boldsymbol{\mathcal{T}}_1 = [1, 0, 0, 0]^\top$, $\boldsymbol{\mathcal{T}}_2 = [0, 0.5, 0, 0]^\top$, $\boldsymbol{\mathcal{T}}_3 = [0, 0, 0.5, 0]^\top$, $\boldsymbol{\mathcal{T}}_4 = [0, 0, 0, 0.5]^\top$ and $\boldsymbol{\mathcal{T}}_5 = [0, 0, 0, 0]^\top$, calculating their associated strain vector $\boldsymbol{q}_i$ by means of the Newton-Raphson method, and substituting these testing samples into (3.38), we can then determine the identified parameters of the PLS Cosserat, as illustrated in Fig. 3.7.



(a) The stiffness matrix.

(b) The inertia matrix.

Figure 3.7: Comparison results between the real value and estimation of material and geometric parameters for the soft manipulator with the nonuniform shape.

The simulation results demonstrate that the proposed parameter identification algorithm is not only feasible but also highly precise. In regard to the estimation of the stiffness matrix, all the components can be accurately identified, as shown in Fig. 3.7a. However, as for the estimation of the mass matrix $\widehat{\boldsymbol{\mathcal{M}}}(X)$ plotted in Fig. 3.7b, only the component ρA is identifiable. This is mainly due to the gravity acceleration located in the negative Z-axis, which results in the vector composed of the matrix $\boldsymbol{B}(\boldsymbol{q})$ post-multiplying by the components $\rho\boldsymbol{\lambda}_1$ and $\rho\boldsymbol{\lambda}_2$ from $\overline{\boldsymbol{Q}}$ always equal to zero. It should be emphasized that these parameters are not necessary for this case. To put it another way, the parameters (i.e., J_x, J_y and J_z) included in the matrix $\boldsymbol{\mathcal{M}}(X)$ can be negligible for the PLS Cosserat static model, which has also been verified by the simulation.

The model parameter identification schemes mentioned above are dependent on the generalized strain vector $\boldsymbol{q}$. It is essential to emphasize that measuring $\boldsymbol{q}$ in practical applications is challenging due to the limitations of strain sensors. To overcome this difficulty and achieve the model parameter identification in practice, we can add more sensors along the soft manipulator to calculate $\boldsymbol{q}$, and then use (3.33) or (3.38) to determine the identified parameters. Alternatively, another method is to solve a nonlinear programming (NLP) problem based on the end-effector position obtained through the magnetic sensor, which we will focus on in the following.

3.5.3 Optimization-Based Approach for Model Parameter Identification

Unlike the two previously proposed parameter identification methods for uniform and nonuniform rods, this scheme is based on the measurement of the end-effector position since the position vector can be easily acquired by using the magnetic position sensor.

To achieve the model parameter identification, it is assumed that $\overline{N}$ sets of different experiments are effectuated and the objective is to seek optimal parameters to minimize the difference between the real measured end-effector position and that obtained from simulation, by satisfying of course the PLS Cosserat static model. Consequently, the proposed identification framework can be formulated by the following NLP problem:

$$\begin{aligned} &\underset{\boldsymbol{\delta}=(\boldsymbol{\theta},\boldsymbol{q}_1,\boldsymbol{q}_2,\cdots,\boldsymbol{q}_{\overline{N}})}{\arg\min} \quad f(\boldsymbol{\delta}) = \sum_{i=1}^{\overline{N}} \|\boldsymbol{p}_i - \boldsymbol{p}_{ei}\|_2^2 \\ &s.t. \quad \begin{cases} \mathbb{K}(\boldsymbol{\theta},\boldsymbol{q}_1) - \mathbb{F}_a(\boldsymbol{q}_1) = \boldsymbol{0} \\ \mathbb{K}(\boldsymbol{\theta},\boldsymbol{q}_2) - \mathbb{F}_a(\boldsymbol{q}_2) = \boldsymbol{0} \\ \qquad\qquad \vdots \\ \mathbb{K}(\boldsymbol{\theta},\boldsymbol{q}_{\overline{N}}) - \mathbb{F}_a(\boldsymbol{q}_{\overline{N}}) = \boldsymbol{0} \end{cases} \end{aligned} \tag{3.39}$$

with

$$\mathbb{K}(\boldsymbol{\theta},\boldsymbol{q}_i) = \begin{bmatrix} -\boldsymbol{K}(\boldsymbol{q}_i) \\ \boldsymbol{\sigma} \end{bmatrix} (\boldsymbol{q}_i - \boldsymbol{q}_{0i}) - \begin{bmatrix} \boldsymbol{G}(\boldsymbol{q}_i)\mathrm{Ad}_{\boldsymbol{g}_r}^{-1}\boldsymbol{\mathcal{G}} \\ \boldsymbol{\mathcal{F}}_e(L_N) \end{bmatrix}, \ \mathbb{F}_a(\boldsymbol{q}_i) = \begin{bmatrix} \boldsymbol{H}(\boldsymbol{q}_i) \\ -\boldsymbol{\Lambda}(L_N) \end{bmatrix} \boldsymbol{T},$$

where $\boldsymbol{p}_i = \boldsymbol{W}\boldsymbol{g}(\boldsymbol{q}_i, L)\boldsymbol{\Upsilon}$ implies the end-effector position in the i^{th} experiment with $\boldsymbol{W} = \begin{bmatrix}\mathbf{I}_3 & \mathbf{0}\end{bmatrix}$ and $\boldsymbol{\Upsilon} = \begin{bmatrix}\mathbf{0}_3 & 1\end{bmatrix}^\top$, $\boldsymbol{g}(\boldsymbol{q}_i, L)$ stands for the position and orientation of the end-effector in the i^{th} experiment, with $\boldsymbol{q}_i$ being the strain vector, $\boldsymbol{\theta}$ represents those parameters to be identified, $\boldsymbol{p}_{ei}$ is the i^{th} experimental measurement of the end-effector position.

Remark 3. *It is worth noting that the number of sections N is pre-chosen in the proposed optimization algorithm since we are interested in identifying the material-related parameter, such as Young's modulus. However, the value of N might be also treated as model parameter to be optimized for the proposed PLS Cosserat model. This will in fact lead to a nonlinear parameter identification problem, which can be handled via a two-stage method. The basic idea of such an approach is to fix firstly the value of N and to solve the optimization problem (3.39) in the first stage. Once the objective function can not be minimized within the prescribed tolerance, then in the second stage we can set N to $N+1$ and repeat the first stage until the optimal number of the sections N is found to minimize the cost function by well satisfying the tolerance.*

The parameter vector $\boldsymbol{\theta}$ consists of the Young's modulus E, shear modulus G and density of material ρ to be identified, noted as $\boldsymbol{\theta} = \begin{bmatrix}E & G & \rho\end{bmatrix}$. To solve the above NLP problem, the Newton-type method is used by attempting to find the optimal solution $\boldsymbol{\delta}^*$ which can generally satisfy the Karush-Kuhn-Tucker (KKT) conditions that there exist multiplier vectors $\overline{\boldsymbol{\lambda}}^* \in \mathbb{R}^{6\overline{N}(N+1)}$ such that the following equations hold:

$$\begin{aligned} \bigtriangledown_{\boldsymbol{\delta}} \mathcal{L}(\boldsymbol{\delta}^*, \overline{\boldsymbol{\lambda}}^*) &= 0 \\ \overline{\boldsymbol{\lambda}}^* &\neq \boldsymbol{0} \\ \underline{\boldsymbol{\mathcal{H}}}(\boldsymbol{\delta}^*) &= \boldsymbol{0} \end{aligned}$$

with

$$\mathcal{L} = f(\boldsymbol{\delta}) + \underline{\boldsymbol{\mathcal{H}}}(\boldsymbol{\delta})^{\top}\overline{\boldsymbol{\lambda}}$$

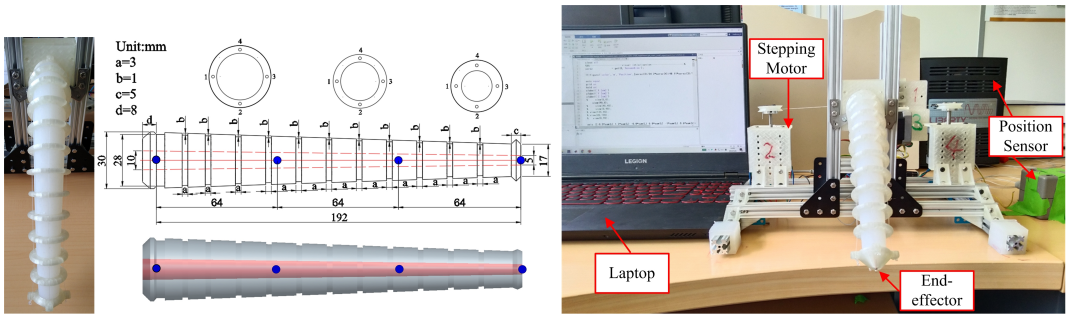
$$\underline{\boldsymbol{\mathcal{H}}}(\boldsymbol{\delta}) = \left[\underline{\boldsymbol{\mathcal{H}}}_1(\boldsymbol{\theta}, \boldsymbol{q}_1)^{\top} \quad \underline{\boldsymbol{\mathcal{H}}}_2(\boldsymbol{\theta}, \boldsymbol{q}_2)^{\top} \quad \cdots \quad \underline{\boldsymbol{\mathcal{H}}}_{\overline{N}}(\boldsymbol{\theta}, \boldsymbol{q}_{\overline{N}})^{\top}\right]^{\top}$$

where $\underline{\boldsymbol{\mathcal{H}}}_i(\boldsymbol{\theta}, \boldsymbol{q}_i) = \mathbb{K}(\boldsymbol{\theta}, \boldsymbol{q}_i) - \mathbb{F}_a(\boldsymbol{q}_i)$ is the static model in the i^{th} experiment, $\mathcal{L}$ represents the Lagrange function, and $\overline{\boldsymbol{\lambda}}$ is the Lagrangian multiplier vector.

Remark 4. *The Newton-type algorithm is sensitive to the choice of initial guess. Note that if a minimum exists, it is not necessarily unique. In other words, there may be an infinite number of feasible points that meet the KKT conditions and are thus minima. However, regardless of the number of local minima, there is always a unique optimal solution (if it exists). To obtain the material parameters of soft manipulator accurately and efficiently, the determination of the initial variable $\boldsymbol{\delta}_0$ composed of material parameters and configuration of the manipulator should be mentioned. In general, it is recommended to start the iteration with estimates that are close to the true parameter values. Following this criterion, we refer to the initial guesses of material parameters (i.e., Young's modulus E, shear modulus G and density ρ) of the soft manipulator provided by manufacturers, and choose the undeformed reference straight shape of the arm as initial configuration. When the material parameters are not available from robot manufacturers, we can guess an initial value according to the properties of the material, and then allow the algorithm to run multiple times in order to determine the optimality of the solution for this NLP problem.*

3.6 Experimental Platform and Model Validation

3.6.1 Illustration of Experimental Platform



(a) Soft manipulator prototype and position sensors distribution.

(b) Experimental setup for the model-based control.

Figure 3.8: Experimental setup for the model parameter identification and control of the slender soft manipulator.

A soft manipulator prototype, similar to that used in the simulation, was designed to carry out the material parameter identification of the PLS Cosserat model by the real input-output relationship obtained from the experimental setup. The exact geometric parameters of the studied manipulator are illustrated in Fig. 3.8a. And several 3D-printed rigid rings are mounted along the soft manipulator to minimize friction between the cables and a single conical piece of silicone. The casting material of the manipulator we used is an isotropic silicone rubber

Table 3.4: Experimental samples for identification algorithm.

Order of experiment	Cables' tension (Unit: N)	End-effector position measured by the sensor (Unit: cm)
1	$[0, 0, 0, 0]^\top$	$[15.77, 0.03, -10.10]^\top$
2	$[0, 0, 0, 0.98]^\top$	$[17.73, 0.02, -6.44]^\top$
3	$[0.98, 0, 0, 4.90]^\top$	$[6.85, -4.77, 7.20]^\top$
4	$[0, 0, 0, 1.96]^\top$	$[18.31, 0.01, -1.90]^\top$
5	$[1.96, 0, 0, 1.96]^\top$	$[13.09, -9.24, -2.46]^\top$
6	$[0.98, 0, 0, 0]^\top$	$[14.72, -4.20, -9.78]^\top$

with unknown material parameters (i.e., Young's modulus, and density of material) which are needed to be identified by using the proposed parameter identification scheme. In addition, the experimental platform includes the MATLAB environment on the laptop, the Polhemus magnetic position sensor attached to the robot's tip, the micro controller, and also the actuation system, as shown in Fig. 3.8b.

The investigated soft manipulator is controlled by 4 cables mounted through it from base to tip, and the cables are respectively driven by different weights, as shown in Fig. 3.9. To obtain the position of any waypoint along the soft manipulator, the magnetic sensors are placed on the corresponding position depicted in blue point in Fig. 3.8a, and for this purpose, a long and conical hole was made along the whole length of the arm.

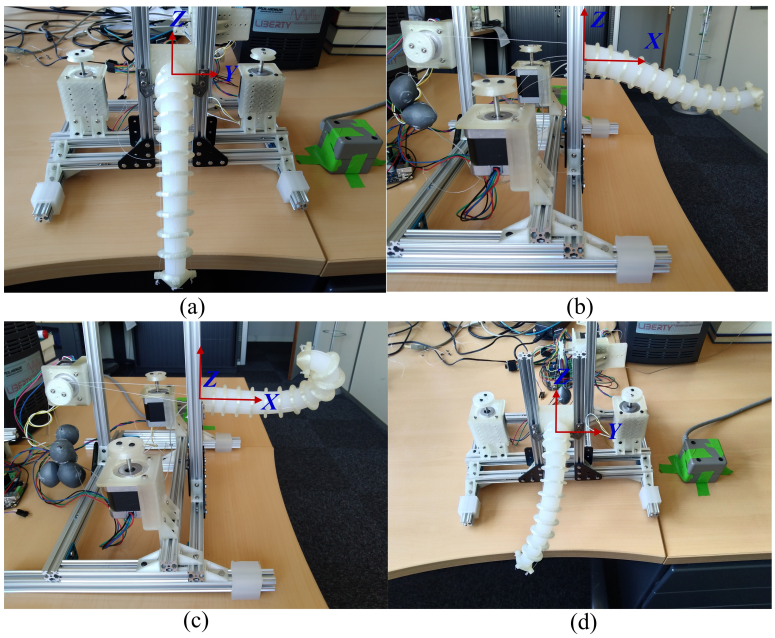


Figure 3.9: Several snapshots of the different experiments for implementing model parameter identification.

3.6.2 Model Validation and Discussion

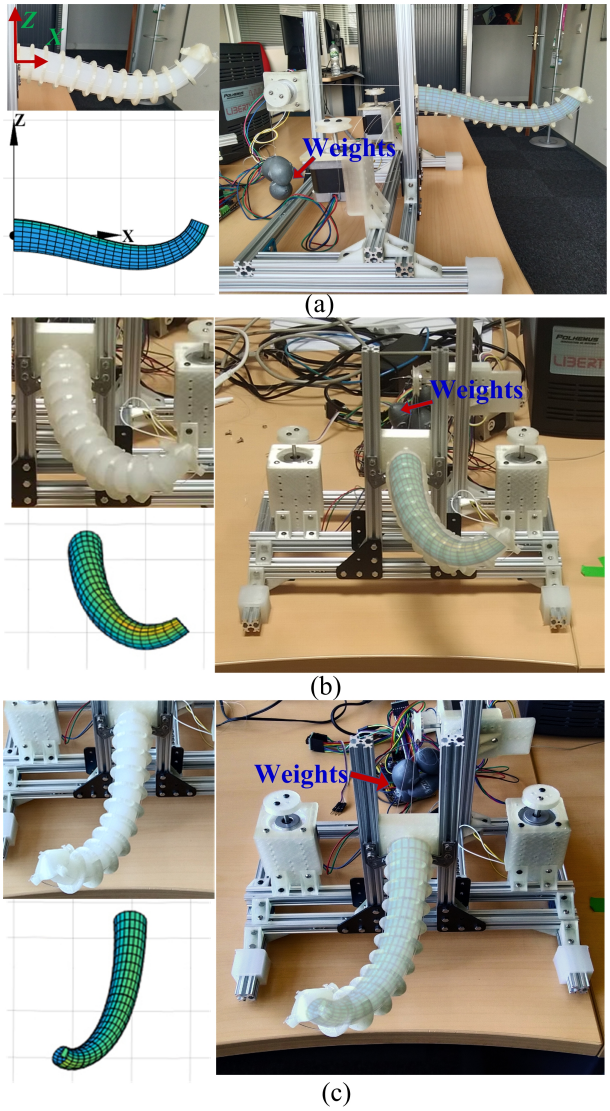


Figure 3.10: Configuration comparison between simulation and experiment under different cables' tension. (a) $\boldsymbol{T} = [0, 0, 0, 2.94]^\top$. (b) $\boldsymbol{T} = [0, 0, 1.96, 1.96]^\top$. (c) $\boldsymbol{T} = [0.98, 0, 0, 2.94]^\top$.

The model validation which determines whether the model is proper enough for its intended use is implemented. Six sets of experiments are performed to acquire the position information of end-effector under the effect of different loads exerted by weights equivalent to the cables' tension. Table 3.4 provides specific input and output values of the experiments, and the position as well as orientation of the manipulator in several cases are displayed in Fig. 3.9. Subsequently, the experimental outputs obtained by the position sensor are utilized to realize the proposed parameter identification algorithm, and thus the material parameters $\boldsymbol{\theta}$ can be calculated.

It should be pointed out that the proposed parameter identification scheme for the PLS

Cosserat model has been established to identify the material-related parameter $\boldsymbol{\theta}$. Thus, if we swap one soft manipulator to another with the same material, or change the configuration of the same robot with the same material, then it is not necessary to repeat the experiments.

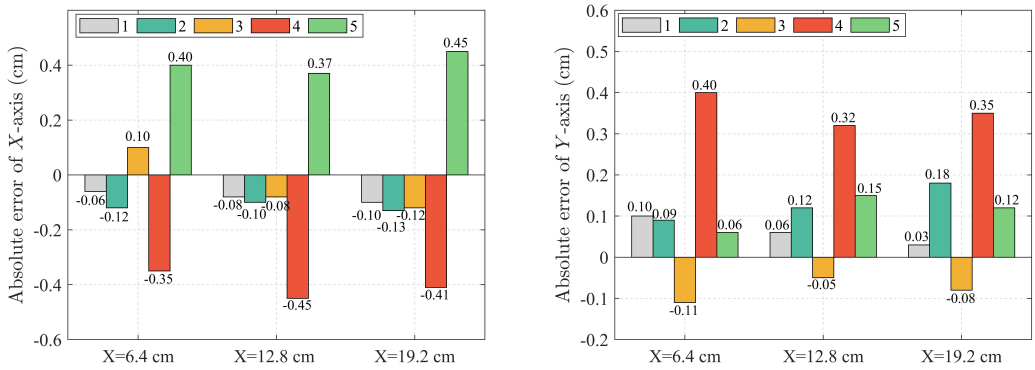
The material-related parameters obtained by solving the NLP problem (3.39) are as follows: Young's modulus $E = 2.563 \times 10^5$ Pa, shear modulus $G = 8.543 \times 10^4$ Pa, and density of material $\rho = 1.41 \times 10^3$ kg/m^3.

Table 3.5: Different experiments for PLS Cosserat model validation.

Order of the control input	Cables' tension (Unit: N)	Average of the waypoints along the soft arm measured by the position sensors (Unit: cm)		
		$X = 6.4$	$X = 12.8$	$X = 19.2$
1 (10 times)	$[0, 0, 0, 2.94]^\top$	$\begin{bmatrix} 6.15 \\ 0.10 \\ -0.60 \end{bmatrix}$	$\begin{bmatrix} 12.29 \\ 0.06 \\ -0.66 \end{bmatrix}$	$\begin{bmatrix} 16.81 \\ 0.03 \\ 2.87 \end{bmatrix}$
2 (10 times)	$[0, 0, 1.96, 1.96]^\top$	$\begin{bmatrix} 5.91 \\ 0.93 \\ -1.03 \end{bmatrix}$	$\begin{bmatrix} 10.97 \\ 3.96 \\ -2.56 \end{bmatrix}$	$\begin{bmatrix} 12.96 \\ 9.42 \\ -2.55 \end{bmatrix}$
3 (10 times)	$[0.98, 0, 0, 2.94]^\top$	$\begin{bmatrix} 6.24 \\ -0.57 \\ -0.55 \end{bmatrix}$	$\begin{bmatrix} 11.83 \\ -2.40 \\ -0.58 \end{bmatrix}$	$\begin{bmatrix} 14.97 \\ -6.03 \\ 2.53 \end{bmatrix}$
4 (10 times)	$[0, 0, 2.94, 6.86]^\top$	$\begin{bmatrix} 4.89 \\ 1.75 \\ 1.95 \end{bmatrix}$	$\begin{bmatrix} 3.96 \\ 5.16 \\ 4.74 \end{bmatrix}$	$\begin{bmatrix} 1.50 \\ 3.69 \\ 2.37 \end{bmatrix}$
5 (10 times)	$[0, 0, 0, 5.88]^\top$	$\begin{bmatrix} 6.30 \\ 0.06 \\ 0.84 \end{bmatrix}$	$\begin{bmatrix} 8.68 \\ 0.15 \\ 5.84 \end{bmatrix}$	$\begin{bmatrix} 4.28 \\ 0.12 \\ 6.89 \end{bmatrix}$

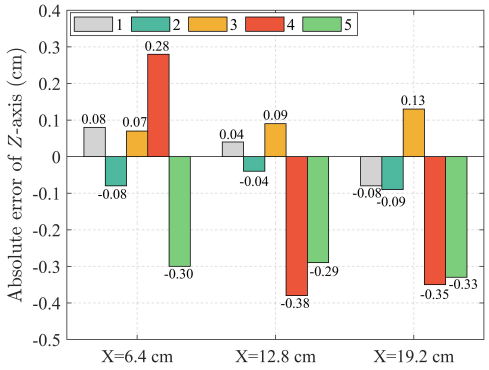
After that, model validation is performed to verify the accuracy of the PLS Cosserat model with the identified parameters. It should be emphasized that the loading conditions (e.g. order-method of applies loads) have a minor or negligible impact on the results. To avoid these uncertain disturbances in the experiments, we repeated 10 times for each input, and then recorded the average of the representative waypoints at $X = 6.4$ cm, $X = 12.8$ cm and $X = 19.2$ cm along the soft manipulator, respectively. Five different groups of control inputs and average of outputs presented in Table 3.5 are selected to compare the position and orientation of the soft arm between the experiments and simulations. The comparison results demonstrate the position vectors at different waypoints along the PLS Cosserat model are almost the same as those of the experiments in three cases illustrated in Fig. 3.10. As for the remaining two sets of experiments, there are larger absolute errors of the selected waypoints between the model

and manipulator than those from the other three cases, which may be due to the occurrence of tiny pleats on the silicone surface caused by relatively larger cables' tension.



(a) X-axis

(b) Y-axis



(c) Z-axis

Figure 3.11: The illustration of absolute errors of the different waypoints (i.e., $X = 6.4$ cm, $X = 12.8$ cm, $X = 19.2$ cm,) for the PLS Cosserat model and soft prototype under five different sets of control inputs.

The absolute errors of the typical waypoints of the model with respect to those of the experiments are all within ± 5 mm, as shown in Fig. 3.11, further showing the effectiveness of the parameter identification method and the accuracy of the PLS Cosserat model.

3.7 Conclusion

In this chapter, a PLS Cosserat model for the slender soft manipulator has been developed for the first time, which combines the strengths of the PCS and GVS Cosserat models. This method depends on a rigorous mathematical framework via the Lie group theory which facilitates a natural coupling of the position and the orientation variables, exhibits an important advantage over avoiding the difficulty in the parameterization of rotation matrix, and contributes to the dynamics modeling of soft manipulators.

The PLS Cosserat static model has been compared with the discrete models reported in the published literature, showing comparable to the FEM and even better results than the PCS Cosserat model in terms of accuracy. In addition, the comparison results of computational efficiency for PLS Cosserat dynamic model solved by adopting both the implicit Euler and explicit Runghe-Kutta methods indicate that there is a significant improvement of the PLS dynamics simulation using implicit Euler method in terms of computation time. All in all, as proved in Section 3.4, the PLS Cosserat model shows great potential to be universally applied to the modeling of slender soft manipulators in the real scenario.

The optimization-based parameter identification scheme of PLS Cosserat model can be described as a NLP problem with several nonlinear equality constraints, which is applicable to identify the material-related parameters of the soft manipulators with arbitrary cross-sectional shape and actuation manner. To carry out the model validation, we designed a soft manipulator prototype, and established the experimental platform. Both simulation and experiment results indicate the proposed scheme is capable of predicting the position along the manipulator with high precision, and provides a foundation for developing the model-based controllers.

Although quite general, the other two parameter identification frameworks proposed in Section 3.5 present a severe limitation that has not yet be addressed. In particular, the assumption of the measurable generalized strain vector allows us to identify the physical parameters of the PLS Cosserat model via (3.33) and (3.38). However, it is rather difficult to acquire the strain of the soft manipulator in practical applications. To address this issue, we will propose a state estimation strategy to estimate the generalized strain vector, which will be introduced in Chapter 4.

Chapter 4

PLS Cosserat Static Model-Based Control for Slender Soft Manipulator

4.1 Introduction

Static/Kinematic model-based controllers are the most widely developed strategies for control of soft continuum robots. Although many researchers have proposed in-depth investigations of control techniques based on different modeling methods (e.g., CC, PCC, FEM, etc.), model-based control schemes for soft robotics have remained an active research topic with several challenges such as improved accuracy, robustness, and adaptability.

The PLS Cosserat model is capable of providing an excellent level of accuracy when handling extreme deformations of the soft slender manipulator, and simultaneously it has low computational complexity due to the utilization of the minimum number of DoFs. Consequently, this modeling technique is easy to implement and friendly to model-based control design.

Until now, to our best of knowledge, there have been few control schemes via the Cosserat models (either static, or dynamic). In this chapter, we will focus on the local and global control of end-effector position of the cable-driven soft manipulator depicted in Fig. 3.8a based on the PLS Cosserat static model, respectively.

In summary, compared to other static or kinematic model-based control schemes for slender soft robots, the contents of this chapter include:

1. A brief introduction of the PLS Cosserat static model suitable to end-effector position control.

2. The PLS Cosserat model-based controller design, theoretical convergence proof as well as experimental validation for the control of the end-effector position of the soft manipulator in a certain sub-workspace.

3. A hybrid approach to globally control the end effector of the cable-driven soft manipulator by unifying PLS Cosserat static model and neural network in order to track the fixed point or different trajectories within its whole workspace, and several physical experimental evaluations for the performances of the proposed controller.

4. The use of the state estimation method to calculate the Jacobian matrix between the actuator space and the task space such that the classical pseudo-inverse method can be directly applied to the robust controller design.

4.2 PLS Cosserat Static Model

As stated in Section 3, the PLS Cosserat method is a spatial discretization technique for solving the differential equations. For the Cosserat models, the strain field can be re-formulated as

$$\boldsymbol{\xi}(X) = \boldsymbol{\xi}_0 + \boldsymbol{\Phi}(X)\boldsymbol{q}(t)$$

where $\boldsymbol{\Phi}(X) \in \mathbb{R}^{6\times 6(N+1)}$ for the PLS Cosserat is a generalized matrix comprised of coefficient matrices of the strain interpolation nodes. According to (3.16) in Section 3.3.3, the generalized PLS Cosserat static model can be obtained, which can be written as the following nonlinear algebraic equation [132]

$$\underline{\boldsymbol{K}}\boldsymbol{q} + \underline{\boldsymbol{G}}(\boldsymbol{q}) = \underline{\boldsymbol{H}}\boldsymbol{T} \tag{4.1}$$

with

$$\underline{\boldsymbol{K}} = \begin{bmatrix} \boldsymbol{K} \\ \boldsymbol{\Sigma}(L_N)\boldsymbol{\Phi}(L_N) \end{bmatrix},\ \underline{\boldsymbol{G}}(\boldsymbol{q}) = \begin{bmatrix} \boldsymbol{G}(\boldsymbol{q})\mathrm{Ad}_{\boldsymbol{g}_r}^{-1}\boldsymbol{\mathcal{G}} \\ -\boldsymbol{\mathcal{F}}_e(L_N) \end{bmatrix},\ \underline{\boldsymbol{H}} = -\begin{bmatrix} \boldsymbol{H} \\ \boldsymbol{\Lambda}(L_N) \end{bmatrix}.$$

Remark 5. *We would like to emphasize that the stiffness matrix $\boldsymbol{K}$ is independent of the rod state $\boldsymbol{q}$ due to a linear constitutive law used for the material modeling. Likewise, considering the boundary condition and simplification approach of $\boldsymbol{K}$, the actuation matrix $\boldsymbol{H}$ is also not related to $\boldsymbol{q}$. Due to PLS assumption, the accurate modeling result can be achieved with less DoFs compared to the common PCS Cosserat modeling technique even for complicated deformations. Remarkably, by means of the PLS Cosserat method, the static system with the boundary condition is converted into an equivalent but simplified nonlinear algebraic equation (4.1) with a finite number of DoFs, which significantly facilitates the analysis and design of control laws.*

Remark 6. *For the soft slender manipulator modeled by PLS Cosserat, the strain-stress relation is linear, with constant Young's modulus. Therefore, the stiffness matrix in the constitutive law is independent of the 6×1 strain twist. In the PLS framework, all types of deformation, including compression/elongation, can be modeled. It is true that the linear strain-stress assumption will introduce certain approximation error in the static PLS model, especially when large deformation occurs. Such an approximation error might be eliminated or attenuated by closed-loop control.*

4.3 Local Controller Design within An Individual Sub-workspace

4.3.1 Local Robust Controller Design

To achieve the manipulator's motion control within a certain sub-workspace, a locally valid closed-loop controller is derived first. Note that we are interested in the end-effector position control of the manipulator, and the problem can be considered as an output tracking of the following nonlinear Cosserat static model

$$\underline{\boldsymbol{K}}\boldsymbol{q} + \underline{\boldsymbol{G}}(\boldsymbol{q}) = \underline{\boldsymbol{H}}\boldsymbol{T} \tag{4.2a}$$

$$\boldsymbol{p}(\boldsymbol{q}) = \mathbf{P}\ \boldsymbol{g}(\boldsymbol{q}, L)\ \mathbf{E} \tag{4.2b}$$

with $\mathbf{P}=\begin{bmatrix}\mathbf{I}_3 & \mathbf{0}\end{bmatrix}$, $\mathbf{E}=\begin{bmatrix}\mathbf{0}_3 & 1\end{bmatrix}^\top$, where $\boldsymbol{g}(\boldsymbol{q},L)$ stands for the position and orientation of the end-effector, $\boldsymbol{p}(\boldsymbol{q})$ implies the measurable end effector position which needs to be controlled with well pre-defined matrices $\mathbf{P}$ and $\mathbf{E}$.

Assumption 3. *For the soft manipulator investigated in the Fig. 3.8a, it is assumed that*

1. *the workspace of the soft manipulator is bounded, i.e., $\boldsymbol{p}(\boldsymbol{q})\in\boldsymbol{\Xi}$, where $\boldsymbol{\Xi}$ stands for the workspace boundedness.*
2. *the control input $\boldsymbol{T}$ is physically bounded, i.e., $\boldsymbol{T}\in\boldsymbol{\mathcal{T}}$, where $\boldsymbol{\mathcal{T}}=\boldsymbol{\mathcal{T}}_1\times\boldsymbol{\mathcal{T}}_2\times\cdots\times\boldsymbol{\mathcal{T}}_{n_u}$ with $\boldsymbol{\mathcal{T}_i}=\left[T^i_{\min},T^i_{\max}\right]$ being the minimal and maximal tension bounds of the ith cable for $i=1,\cdots,n_u$.*
3. *the control input $\boldsymbol{T}$ and output $\boldsymbol{p}(\boldsymbol{q})$ are differentiable, i.e., $\boldsymbol{p}(\boldsymbol{q}),\boldsymbol{T}\in\mathcal{C}^\infty$.*

Based on the PLS Cosserat static model (4.2), the derivative relation between the end-effector position $\boldsymbol{p}(\boldsymbol{q})$ and the cables' tension $\boldsymbol{T}$ can be deduced as follows:

$$\dot{\boldsymbol{p}}(\boldsymbol{q})=\boldsymbol{\Gamma}(\boldsymbol{q})\dot{\boldsymbol{T}} \tag{4.3}$$

where $\boldsymbol{\Gamma}(\boldsymbol{q})\in\mathbb{R}^{3\times4}$ takes the partial derivative of $\boldsymbol{p}$ with respect to $\boldsymbol{T}$, and represents a Jacobian matrix which is a measurement of the mechanical coupling between end-effector position and the actuators.

4.3.1.1 Analytical Solution of $\boldsymbol{\Gamma}$

The calculation of $\boldsymbol{\Gamma}(\boldsymbol{q})$ will be exhaustively deduced. According to its definition, $\boldsymbol{\Gamma}(\boldsymbol{q})$ can be formulated as

$$\boldsymbol{\Gamma}(\boldsymbol{q})=\frac{\partial\boldsymbol{p}(\boldsymbol{q})}{\partial\boldsymbol{T}}=\frac{\partial\boldsymbol{p}(\boldsymbol{q})}{\partial\boldsymbol{q}}\frac{\partial\boldsymbol{q}}{\partial\boldsymbol{T}}=\boldsymbol{p_q}\boldsymbol{q_T} \tag{4.4}$$

In the light of the matrix representation of the velocity twist $\widehat{\boldsymbol{\eta}}(\boldsymbol{q},L)$ of the end-effector, the linear velocity $\boldsymbol{V}(\boldsymbol{q},L)$ in the local frame can be derived

$$\boldsymbol{V}(\boldsymbol{q},L)=\boldsymbol{R}^{-1}(\boldsymbol{q},L)\dot{\boldsymbol{p}}(\boldsymbol{q})=\mathbf{P}\;\widehat{\boldsymbol{\eta}}(\boldsymbol{q},L)\;\mathbf{E}$$

with $\widehat{\boldsymbol{\eta}}(\boldsymbol{q},L)=\boldsymbol{g}^{-1}(\boldsymbol{q},L)\;\dot{\boldsymbol{g}}(\boldsymbol{q},L)$, we then calculate the term $\widehat{\boldsymbol{\eta}}(\boldsymbol{q},L)\mathbf{E}$

$$\widehat{\boldsymbol{\eta}}(\boldsymbol{q},L)\;\mathbf{E}=\boldsymbol{g}^{-1}(\boldsymbol{q},L)\;\dot{\boldsymbol{g}}(\boldsymbol{q},L)\;\mathbf{E}=\boldsymbol{g}^{-1}(\boldsymbol{q},L)\;\frac{\partial\boldsymbol{g}(\boldsymbol{q},L)}{\partial\boldsymbol{q}}\;(\mathbf{I}_4\otimes\dot{\boldsymbol{q}})\;\mathbf{E}$$

Using the definition of the configuration matrix $\boldsymbol{g}(\boldsymbol{q},L)$, the above equation is re-formulated and yields

$$\begin{aligned}\widehat{\boldsymbol{\eta}}(\boldsymbol{q},L)\;\mathbf{E}&=\begin{bmatrix}\boldsymbol{R}^{-1}(\boldsymbol{q},L) & -\boldsymbol{R}^{-1}(\boldsymbol{q},L)\boldsymbol{p}(\boldsymbol{q})\\ \mathbf{0} & 1\end{bmatrix}\begin{bmatrix}\dfrac{\partial\boldsymbol{R}(\boldsymbol{q},L)}{\partial\boldsymbol{q}} & \dfrac{\partial\boldsymbol{p}(\boldsymbol{q})}{\partial\boldsymbol{q}}\\ \mathbf{0} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{0}\\ \dot{\boldsymbol{q}}\end{bmatrix}\\&=\begin{bmatrix}\boldsymbol{R}^{-1}(\boldsymbol{q},L)\dfrac{\partial\boldsymbol{p}(\boldsymbol{q})}{\partial\boldsymbol{q}}\dot{\boldsymbol{q}} & 0\end{bmatrix}^\top\end{aligned}$$

Substituting the above equation of $\widehat{\boldsymbol{\eta}}(\boldsymbol{q},L)\mathbf{E}$ in the linear velocity $\dot{\boldsymbol{p}}(\boldsymbol{q})$, we achieve the following expression of $\dot{\boldsymbol{p}}(\boldsymbol{q})$

$$\dot{\boldsymbol{p}}(\boldsymbol{q})=\boldsymbol{R}(\boldsymbol{q},L)\mathbf{P}\;\widehat{\boldsymbol{\eta}}(\boldsymbol{q},L)\;\mathbf{E}=\frac{\partial\boldsymbol{p}(\boldsymbol{q})}{\partial\boldsymbol{q}}\;\dot{\boldsymbol{q}} \tag{4.5}$$

Combining the definition of velocity twist with (3.10), the relation between velocity twist $\boldsymbol{\eta}(\boldsymbol{q}, L)$ and geometric Jacobian $\boldsymbol{J}(\boldsymbol{q}, L)$ is given by

$$\boldsymbol{\eta}(\boldsymbol{q}, L) = \begin{bmatrix} \boldsymbol{\Omega}^\top(\boldsymbol{q}) & (\boldsymbol{R}^{-1}(\boldsymbol{q}, L)\dot{\boldsymbol{p}}(\boldsymbol{q}))^\top \end{bmatrix}^\top = \boldsymbol{J}(\boldsymbol{q}, L)\, \dot{\boldsymbol{q}}$$

which arrives at

$$\dot{\boldsymbol{p}}(\boldsymbol{q}) = \boldsymbol{R}(\boldsymbol{q}, L)[\mathbf{0}_3\ \mathbf{I}_3]\ \boldsymbol{J}(\boldsymbol{q}, L)\, \dot{\boldsymbol{q}} \tag{4.6}$$

Using the expressions (4.5) and (4.6), the following equation can be deduced

$$\frac{\partial \boldsymbol{p}(\boldsymbol{q})}{\partial \boldsymbol{q}}\, \dot{\boldsymbol{q}} = \boldsymbol{R}(\boldsymbol{q}, L)\ [\mathbf{0}_3\ \mathbf{I}_3]\ \boldsymbol{J}(\boldsymbol{q}, L)\, \dot{\boldsymbol{q}}$$

Accordingly, the partial derivative of $\boldsymbol{p}(\boldsymbol{q})$ with respect to $\boldsymbol{q}$ is given by

$$\boldsymbol{p}_{\boldsymbol{q}}(\boldsymbol{q}) = \frac{\partial \boldsymbol{p}(\boldsymbol{q})}{\partial \boldsymbol{q}} = \boldsymbol{R}(\boldsymbol{q}, L)\ [\mathbf{0}_3\ \mathbf{I}_3]\boldsymbol{J}(\boldsymbol{q}, L) = [\mathbf{0}_3 \boldsymbol{R}(\boldsymbol{q}, L)]\boldsymbol{J}(\boldsymbol{q}, L)$$

Using the principle of variable separation, the derivative of (4.2a) with respect to $\boldsymbol{T}$ is equivalent to

$$\left(\underline{\boldsymbol{K}} + \frac{\partial[\underline{\boldsymbol{G}}(\boldsymbol{q})]}{\partial \boldsymbol{q}}\right)\frac{\partial \boldsymbol{q}}{\partial \boldsymbol{T}} = \underline{\boldsymbol{H}}$$

It is then supposed that the cables are installed in a manner such that the matrix $\left(\underline{\boldsymbol{K}} + \frac{\partial[\underline{\boldsymbol{G}}(\boldsymbol{q})]}{\partial \boldsymbol{q}}\right)$ deduced is invertible, then the partial derivative of $\boldsymbol{q}$ with respect to $\boldsymbol{T}$ can be written as

$$\boldsymbol{q}_{\boldsymbol{T}} = \frac{\partial \boldsymbol{q}}{\partial \boldsymbol{T}} = \left(\underline{\boldsymbol{K}} + \frac{\partial[\underline{\boldsymbol{G}}(\boldsymbol{q})]}{\partial \boldsymbol{q}}\right)^\dagger \underline{\boldsymbol{H}}$$

where $(\cdot)^\dagger$ implies the pseudo-inverse of $(\cdot)$. Finally, the analytical solution of the matrix $\boldsymbol{\Gamma}$ yields

$$\boldsymbol{\Gamma}(\boldsymbol{q}) = \boldsymbol{p}_{\boldsymbol{q}}(\boldsymbol{q})\boldsymbol{q}_{\boldsymbol{T}} = [\mathbf{0}_3\ \boldsymbol{R}(\boldsymbol{q}, L)]\ \boldsymbol{J}(\boldsymbol{q}, L)\left(\underline{\boldsymbol{K}} + \frac{\partial[\underline{\boldsymbol{G}}(\boldsymbol{q})]}{\partial \boldsymbol{q}}\right)^\dagger \underline{\boldsymbol{H}} \tag{4.7}$$

Remark 7. *It is worth noting that, for any desired position $\boldsymbol{p}_r \in \boldsymbol{\Xi}$, there always exists $\boldsymbol{T} \in \mathcal{T}$ such that the end-effector position $\boldsymbol{p}$ can be driven to $\boldsymbol{p}_r$. Consequently, this implies that $\boldsymbol{\Gamma}(\boldsymbol{q})$ defined in (4.4) is right invertible, i.e., $\exists\ \boldsymbol{\Gamma}_R^{-1} \in \mathbb{R}^{4\times 3}$ such that $\boldsymbol{\Gamma}\boldsymbol{\Gamma}_R^{-1} = \mathbf{I}_3$. Compared to other methods (such as FEM, PCC, PCS, etc.), the PLS Cosserat model can obtain analytical input-output relationship of the slender soft robots accurately with low dimension, which is favorable to realize control objective in real time.*

4.3.1.2 Control Law and System Stability Proof

Given a known end-effector position $\boldsymbol{p}^* \in \boldsymbol{\Xi}$ associated to a pair $(\boldsymbol{q}^*, \boldsymbol{T}^*)$ satisfying (4.1), we denote its equilibrium neighborhood as $\mathcal{N}(\boldsymbol{q}^*, \boldsymbol{T}^*) \in \boldsymbol{\Xi}$. Then, for any end-effector position $\boldsymbol{p}(t) \in \boldsymbol{\Xi}$ and a given desired end-effector position $\boldsymbol{p}_r \in \mathcal{N}$, the objective of our work is to design a robust controller $\boldsymbol{T}(t)$ such that $\boldsymbol{p}(t) \to \boldsymbol{p}_r$ as $t \to \infty$.

When designing such a controller, the main difficulty is that the generalized strain vector $\boldsymbol{q}$ is generally not measurable in practice, thus conventional kinematic model-based control methods which require the knowledge of $\boldsymbol{\Gamma}(\boldsymbol{q})$ in real time, such as pseudo-inverse, cannot be directly applied in this study. To overcome this difficulty, the following proposes to approximate $\boldsymbol{\Gamma}(\boldsymbol{q})$ around its equilibrium $(\boldsymbol{q}^*, \boldsymbol{T}^*)$ by a constant matrix $\boldsymbol{\Gamma}_0(\boldsymbol{q}^*)$ which can be analytically computed due to the knowledge of the pair $(\boldsymbol{q}^*, \boldsymbol{T}^*)$ in advance, and then design a robust controller based on $\boldsymbol{\Gamma}_0(\boldsymbol{q}^*)$.

Concretely, according to the deduced input-output relation (4.3), the following robust controller is designed:

$$\dot{\boldsymbol{T}} = -\lambda \boldsymbol{\Gamma}_{0R}^{-1}(\boldsymbol{q}^*)\boldsymbol{e} \tag{4.8}$$

where $\lambda > 0$, $\boldsymbol{e} = \boldsymbol{p}(t) - \boldsymbol{p}_r$. Then, the following result theoretically ensures the convergence of $\boldsymbol{p}(t)$ to the desired constant position $\boldsymbol{p}_r$.

Theorem 2. *For any $\boldsymbol{p}(t) \in \mathcal{N}$ and a desired $\boldsymbol{p}_r \in \mathcal{N}$ satisfying (4.2), if the following inequality is satisfied:*

$$\|\Delta\boldsymbol{\Gamma}\boldsymbol{\Gamma}_{0R}^{-1}\|_2 \leqslant \varepsilon < 1 \tag{4.9}$$

where $\Delta\boldsymbol{\Gamma}(\boldsymbol{q}) = \boldsymbol{\Gamma}(\boldsymbol{q}) - \boldsymbol{\Gamma}_0(\boldsymbol{q}^*)$, then with the proposed controller (4.8), $\boldsymbol{p}(t) \in \boldsymbol{\Xi}$ can asymptotically converge to $\boldsymbol{p}_r \in \boldsymbol{\Xi}$, i.e., $\lim_{t\to\infty} \|\boldsymbol{p}(t) - \boldsymbol{p}_r\|_2 = 0$.

Proof. Note that we want to prove $\boldsymbol{e}(t) \to \boldsymbol{0}$ when $t \to \infty$, and this is equivalent to prove $V(\boldsymbol{e}) \to 0$ when $t \to \infty$, where $V(\boldsymbol{e})$ is a Lyapunov function defined as $V(\boldsymbol{e}) = \boldsymbol{e}^\top\boldsymbol{e}/2$. Then, taking the derivative of $V(\boldsymbol{e})$ w.r.t. time t yields

$$\begin{aligned}\frac{\partial V}{\partial t} = \dot{V} &= \boldsymbol{e}^\top\dot{\boldsymbol{e}} = \boldsymbol{e}^\top\dot{\boldsymbol{p}} = \boldsymbol{e}^\top\boldsymbol{\Gamma}\dot{\boldsymbol{T}} \\ &= -\lambda\boldsymbol{e}^\top\left(\boldsymbol{\Gamma}_0 + \Delta\boldsymbol{\Gamma}\right)\boldsymbol{\Gamma}_{0R}^{-1}\boldsymbol{e} \\ &= -2\lambda V - \lambda\boldsymbol{e}^\top\Delta\boldsymbol{\Gamma}\boldsymbol{\Gamma}_{0R}^{-1}\boldsymbol{e}\end{aligned}$$

Consequently, if (4.9) holds, we can come to a conclusion that

$$\dot{V} \leqslant -2\lambda V + \lambda\boldsymbol{e}^\top\varepsilon\boldsymbol{e} = -2\lambda\left(1-\varepsilon\right)V = -2\lambda\beta V < 0$$

with $\lambda > 0$ and $\beta = 1 - \varepsilon > 0$, which guarantees that V exponentially converges to 0. Thus, we can derive that $\boldsymbol{e}(t)$ exponentially converges to $\boldsymbol{0}$. □

Remark 8. *In Theorem 2, the inequality of (4.9) implies that the approximation of $\boldsymbol{\Gamma}(\boldsymbol{q})$ by $\boldsymbol{\Gamma}_0(\boldsymbol{q}^*)$ should be close in the norm sense. For the trajectory tracking problem, the detailed proof of the convergence by using the same controller (4.8) can refer to [32]. Furthermore, we would like to emphasize that it is possible to replace $\dot{\boldsymbol{T}} = -\lambda\boldsymbol{\Gamma}_{0R}^{-1}\boldsymbol{e}$ by $\dot{\boldsymbol{T}} = -\lambda\boldsymbol{\Gamma}_0^\top\boldsymbol{e}$ in order to avoid the inverse calculation of $\boldsymbol{\Gamma}_0$. For this, by the same idea of V, we can prove as well the convergence of $\boldsymbol{e}$ to $\boldsymbol{0}$.*

4.3.2 Closed-Loop Experiments on Soft Prototype

Several experiments were conducted on the soft manipulator shown in Fig. 3.8 to demonstrate the capabilities of the proposed controller, which mainly consist of point-to-point test, time-varying trajectory tracking tests, and robustness analysis under temporary and permanent disturbances. Also, the related video demonstration is available [1].

[1] Video demonstration: `https://1drv.ms/v/s!Ak7hK-nVdKMei1zE-wEtbS1piJ_7?e=bqQOUm`

4.3.2.1 Point-to-Point Test

Table 4.1: Equilibrium points selected from its workspace.

Equilibrium points	A	B	C	D	E
X(cm)	15.431	15.822	10.885	14.967	13.921
Y(cm)	4.330	-5.454	9.104	-4.483	0.000
Z(cm)	3.391	-0.460	-5.494	-6.372	-8.788

In this test, 4 different equilibrium points (see Table 4.1) from its bounded workspace ($\boldsymbol{p}_r \in \boldsymbol{\Xi}$) were selected as the desired points to follow for the soft manipulator. The experimental results have been depicted in Fig. 4.1. It can be clearly seen that the end-effector position of the arm converges to its desired points with a comparable tracking precision through the proposed control scheme. When following the paths (A-B-C, and D-E), despite the presence of small overshoot in the X-axis direction, the soft manipulator is always able to rapidly converge to the targeted points within 2 s. However, the soft arm converges very slowly in the X-axis direction of the segment D, and takes about 8 s to stabilize from point C to point D. For this phenomenon, we think it occurs since there exists the difference between the real Jacobian matrix and the matrix $\boldsymbol{\Gamma}_0$ obtained from the local linearization of the PLS Cosserat just at the point E.

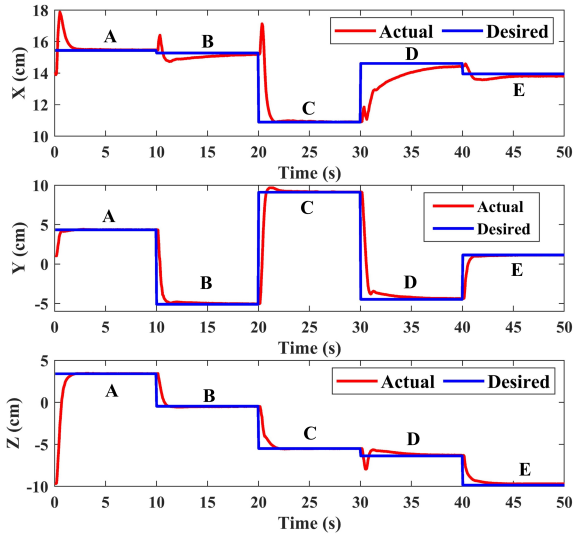


Figure 4.1: The results of point-to-point test by using the proposed controller.

4.3.2.2 Trajectory Tracking Tests

In this experiment, the manipulator was expected to track the circular and star-shaped trajectories using the proposed controller with a fixed $\boldsymbol{\Gamma}_0$ obtained from the PLS Cosserat static model. The results are displayed in Fig. 4.2, Fig. 4.3 and Fig. 4.4, which indicate that the end-effector of the manipulator was capable of rapidly reacting to the different slowly time-varying trajectories. To put it another way, the proposed controller is of high feasibility and effectiveness for the manipulator moving around its workspace.

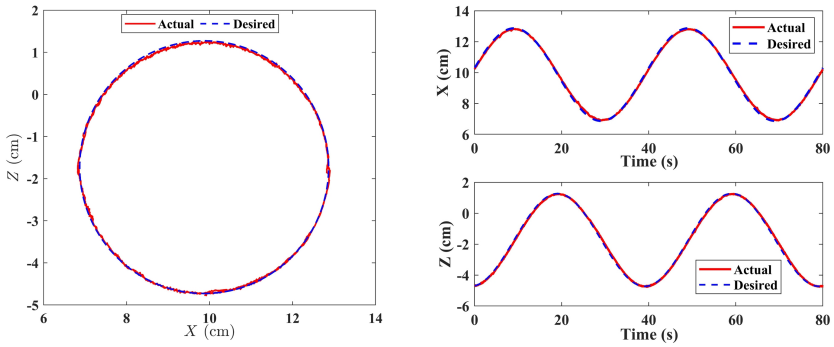


Figure 4.2: Time-varying circular trajectory tracking in X-Z plane by using the proposed controller.

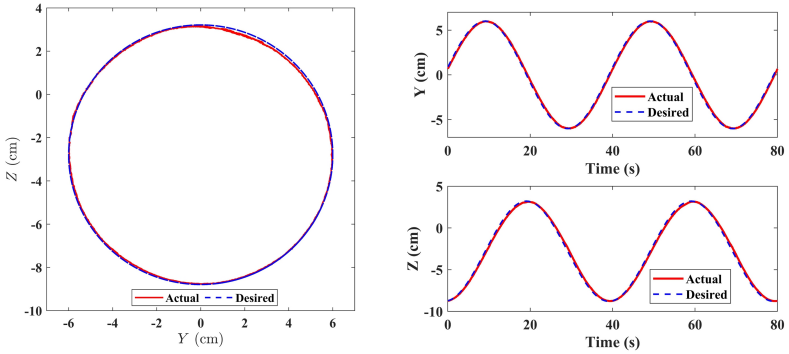


Figure 4.3: Time-varying circular trajectory tracking in Y-Z plane by using the proposed controller.

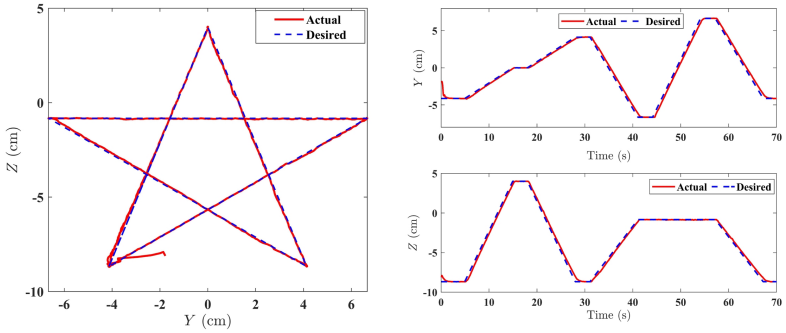


Figure 4.4: Time-varying star-shaped trajectory tracking by applying the proposed controller.

4.3.2.3 Robustness Analysis

As we all know, the need for a closed-loop control strategy is more important in the presence of external disturbances. To prove that our model-based controller proposed is robust to external disturbances, we carried out the following tracking tasks with external disturbances.

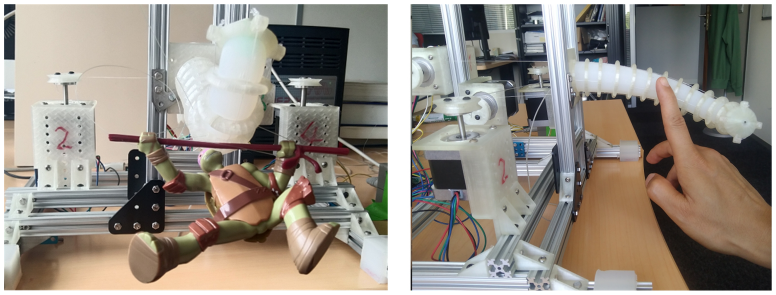


Figure 4.5: From the left, the permanent disturbance by adding a TMNT under the soft manipulator, followed by the temporary disturbance by touching the manipulator with hand.

Permanent disturbance: The manipulator carrying a Teenage Mutant Ninja Turtles (TMNT) (0.032 kg) which can be regarded as an externally permanent disturbance was following a slowly time-varying circular trajectory, as shown on the left of Fig. 4.5. In view of the results from Fig. 4.6, the end-effector of the soft manipulator steadily tracks the trajectory, which demonstrates that the closed-loop control scheme via the PLS Cosserat static model can resist the externally permanent disturbances.

Temporary disturbance: The manipulator reaches the point D from the initial equilibrium position E. The externally temporary disturbances from different directions are exerted on the manipulator with hand displayed on the right of Fig. 4.5, and then we observe whether the manipulator can return to D. From the results of Fig. 4.7, the end-effector of the manipulator quickly converges to the point D, which shows great robustness of the proposed controller against temporary disturbances.

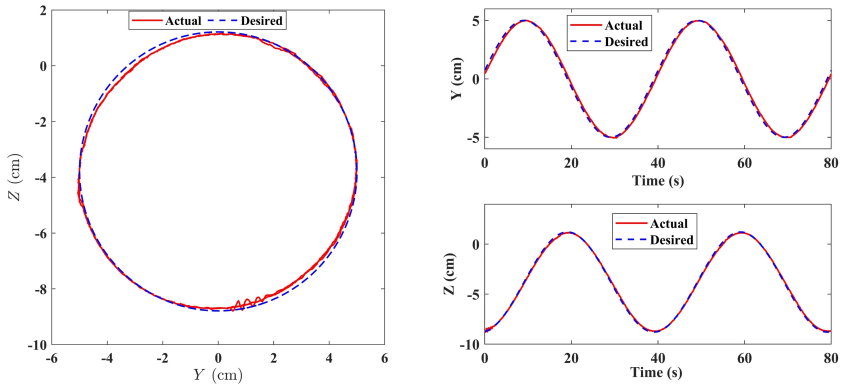


Figure 4.6: The behavior of the manipulator for tracking time-varying circular trajectory in Y–Z plane under permanent disturbance.

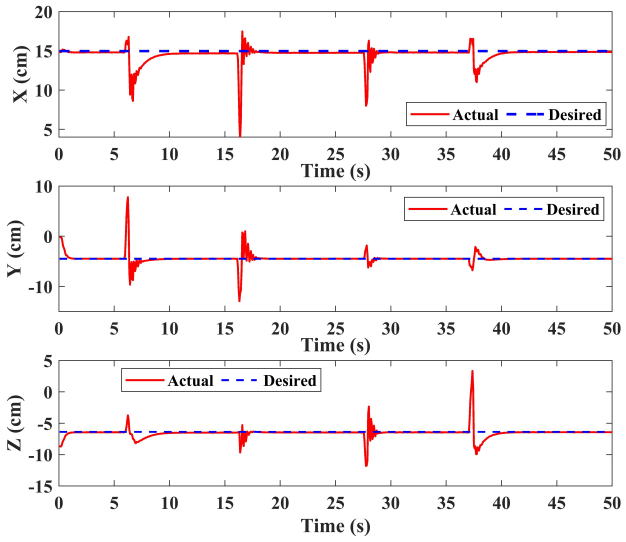


Figure 4.7: The behavior of the manipulator for reaching the target point in X–Y–Z axis with temporary disturbances.

Remark 9. *Compared to the other model-based control schemes (such as FEM-based controller [133]), there exist several advantages for the PLS Cosserat static-based controller: the analytical solution of the model can be obtained, which significantly reduces the model computation time and achieves real-time control; the static model via the PLS assumption has a good accuracy and much less DoFs. To the authors' knowledge, this is the first time that such a robust static model-based controller is proposed, theoretically proved and experimentally validated on the designed soft prototype.*

4.3.3 Discussion and Limitations

The Cosserat model of the soft manipulator accounts for the large deformations due to extension, shear, bending and torsion, and builds on top of the Cosserat rod theory whose differential equations are formulated on the Lie group.

Based on the PLS Cosserat static model, the local controller has been designed for the soft manipulator in its sub-workspace. Then, the control system stability has been proven by a constructed Lyapunov function, and the result indicates that the control scheme can always guarantee the exponential convergence of the end-effector position of the soft manipulator towards the desired reference point or trajectory. Finally, the experimental validation has been implemented by controlling the soft manipulator to track the slowly time-varying trajectories within its available workspace, which corroborates the high accuracy and great robustness of the proposed controller.

In conclusion, the proposed local control scheme can be used in specific unstructured environments for the soft manipulators to perform some positioning tasks with high precision. However, the local controller for the soft manipulator in a specific sub-workspace is not necessarily suitable to the control in another sub-workspace, it is therefore essential to put forward a global controller which is valid in its whole workspace.

4.4 Global Control of Soft Manipulator by Unifying Cosserat and Neural Network

In order to achieve the global control of soft manipulator via the PLS Cosserat static model, this section aims to globally (i.e., for all admissible $\boldsymbol{q}$) approximate $\boldsymbol{\Gamma}(\boldsymbol{q})$ in advance, and thus design a robust pseudo-inverse controller, based only on the measurement of end-effector position, to realize the global control of $\boldsymbol{p}$.

Concretely, we propose to divide the whole workspace $\mathcal{W}_E$ into several sub-workspaces, denoted as $\mathcal{W}_i \subseteq \mathcal{W}_E$, $i = 1, 2, \cdots, m$, where m represents the number of the sub-workspaces, as shown in Fig. 4.8, choose a representative equilibrium point $\boldsymbol{p}_i \in \mathcal{W}_i$ from the ith sub-workspace, and calculate its associated Jacobian matrix $\boldsymbol{\Gamma}_i$ in advance. Subsequently, the global approximation of $\boldsymbol{\Gamma}(\boldsymbol{q})$ will be realized via a neural network (with parameter α_i to be trained) as

$$\boldsymbol{\Gamma}(\boldsymbol{q}) = \widehat{\boldsymbol{\Gamma}}(\boldsymbol{p}) = \sum_{i=1}^{m} \alpha_i(\boldsymbol{p}_i)\boldsymbol{\Gamma}_i$$

where $\boldsymbol{\Gamma}_i$ represents the ith representative Jacobian matrix.

In summary, to design the presented global controller by unifying PLS Cosserat static model and neural network, the following 4 questions need to be answered:

Q1: How to estimate and divide the global workspace $\mathcal{W}_E$?

Q2: For each sub-workspace, how to calculate the corresponding $\boldsymbol{\Gamma}_i$?

Q3: With the pre-calculated $\boldsymbol{\Gamma}_i$, how to construct the neural network to approximate $\boldsymbol{\Gamma}(\boldsymbol{q})$?

Q4: With the obtained approximation $\widehat{\boldsymbol{\Gamma}}$, how to design a robust controller to realize the global control objective of $\boldsymbol{p}$?

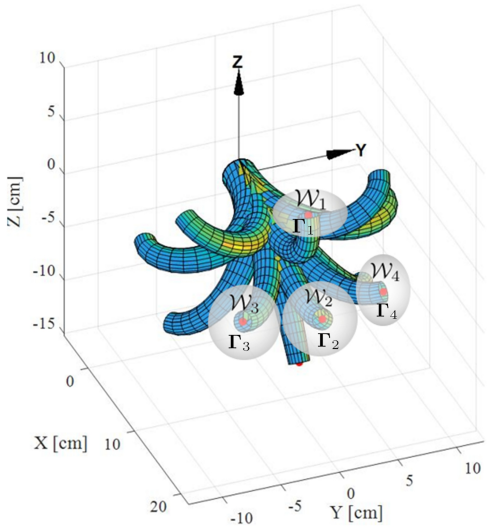


Figure 4.8: Sub-workspaces and their associated configurations for the soft manipulator.

In the following, we firstly present a workspace estimation and decomposition approach for the PLS Cosserat model, which provides the benefit for the soft robotic applications mainly

related to their control. Next, an optimization approach is presented to solve the inverse PLS Cosserat statics for the purpose of computing $\mathbf{\Gamma}_i$. After that, a neural network based strategy for the offline estimation of the Jacobian matrix between the input and output of the soft manipulator in its whole workspace is proposed. Finally, we design a closed-loop feedback controller, provide the theoretical convergence proof, and carry out the experiments on the studied soft manipulator to validate the control performance.

4.4.1 Workspace Estimation and Decomposition

The subject of workspace determination has been widely studied by the soft robotic community, which includes all its equilibrium points where the end-effector of soft robots can reach. The goal of this subsection is to estimate and decompose the feasible workspace of the soft manipulator based on the PLS Cosserat model by using the optimization-based approach [134]. The workspace $\mathcal{W}_E$ of the robot's end-effector can be defined as

$$\mathcal{W}_E = \left\{ \boldsymbol{p} \in \mathbb{R}^3 | \boldsymbol{p} = \mathbf{P}\boldsymbol{g}(\boldsymbol{q})\mathbf{E}, \underline{\boldsymbol{K}}\boldsymbol{q} + \underline{\boldsymbol{G}}(\boldsymbol{q}) = \underline{\boldsymbol{H}}\boldsymbol{T}, \forall \boldsymbol{T} \in \boldsymbol{\mathcal{T}} \right\}$$

where $\boldsymbol{\mathcal{T}} = \boldsymbol{\mathcal{T}}_{\mathbf{1}} \times \cdots \times \boldsymbol{\mathcal{T}}_{n_u}$ with $\boldsymbol{\mathcal{T}}_{\boldsymbol{i}} = \left[T^i_{\min}, T^i_{\max}\right]$ being the minimal and maximal tension bounds of the ith cable for $i = 1, \cdots, n_u$.

To obtain the correct estimation of the whole workspace, an optimization-based method is implemented on the soft manipulator that has been modeled by the PLS Cosserat approach. As a result, the whole workspace estimation algorithm can be formulated as

$$\begin{aligned} &\underset{\boldsymbol{q}^{(i)},\ \boldsymbol{T}^{(i)}}{\arg\min} \ \|\boldsymbol{p}(\boldsymbol{q}^{(i)}) - \boldsymbol{\mathcal{E}}^{(i)}\|_2^2 \\ &s.t. \quad \begin{cases} \underline{\boldsymbol{K}}\boldsymbol{q}^{(i)} + \underline{\boldsymbol{G}}(\boldsymbol{q}^{(i)}) - \underline{\boldsymbol{H}}\boldsymbol{T}^{(i)} = \mathbf{0} \\ \boldsymbol{T}^{(i)} \in \boldsymbol{\mathcal{T}} \end{cases} \end{aligned} \tag{4.10}$$

with

$$\begin{aligned} \boldsymbol{p}(\boldsymbol{q}^{(i)}) &= \mathbf{P}\boldsymbol{g}(\boldsymbol{q}^{(i)})\mathbf{E} \\ \boldsymbol{\mathcal{E}}^{(i)} &= \boldsymbol{\delta}_x^{(i)}\boldsymbol{\delta}_y^{(i)}\boldsymbol{\delta}_z^{(i)}\boldsymbol{\mathcal{E}}_0 \end{aligned}$$

where $\boldsymbol{\mathcal{E}}^{(i)}$ represents the selected radiating vector outside of the workspace $\mathcal{W}_E$, $\boldsymbol{\mathcal{E}}_0$ is the initial radiating vector, $\boldsymbol{\delta}^{(i)}_{x,y,z}$ stand for the basic rotation matrices about the x-, y-, and z-axis, respectively. The detailed algorithm procedures of the optimization-based method along with the scheme to address the non-convexity problem (i.e., how to accurately determine the interior boundary of the workspace) can be found in [134]. Eventually, we can estimate the corresponding end-effector workspace for the studied soft manipulator by solving the minimization problem (4.10).

With the obtained whole workspace $\mathcal{W}_E$, the scheme of workspace decomposition will be then introduced. In terms of two different points $\boldsymbol{p}_i \in \mathcal{W}_E$ and $\boldsymbol{p}_{i+l} \in \mathcal{W}_E$, for $l = 1, 2, \cdots, S$ (any point around $\boldsymbol{p}_i$), the Jacobian matrices (i.e., $\mathbf{\Gamma}_i$, and $\mathbf{\Gamma}_{i+l}$) corresponding to the end-effector positions (i.e., $\boldsymbol{p}_i$, and $\boldsymbol{p}_{i+l}$) can be obtained, respectively. Thus, we propose the following decomposition rule along the workspace surface to determine whether $\boldsymbol{p}_i$ and $\boldsymbol{p}_{i+l}$ belong to the same sub-workspace or not:

$$\mathrm{Sgn}(\mathbf{\Gamma}_i) = \mathrm{Sgn}(\mathbf{\Gamma}_{i+l}) \tag{4.11}$$

where the symbol $\mathrm{Sgn}(\cdot)$ applies the conventional sign function to the matrix via an element-wise way. The sign for the elements of the Jacobian matrix $\mathbf{\Gamma}$ represents the directional relation

between cables' tension and end-effector position. If the rule formulated in (4.11) holds, then $\boldsymbol{p}_i$ and $\boldsymbol{p}_{i+l}$ will be classified into one sub-workspace $\mathcal{W}_i$ where $\boldsymbol{p}_i$ is defined as the representative equilibrium point of the sub-workspace. Otherwise, the two distinct sub-workspaces will be generated for $\boldsymbol{p}_i$ and $\boldsymbol{p}_{i+l}$, respectively. Through the comparison of the elements' sign from the Jacobian matrices around all equilibrium points in $\mathcal{W}_E$ iteratively, we can finally decompose the whole workspace of the slender soft manipulator via the PLS Cosserat into m different sub-workspaces.

4.4.2 Calculation of $\boldsymbol{\Gamma}_i$ for the ith Sub-Workspace

To compute $\boldsymbol{\Gamma}_i$, we need to know the analytical formula of $\boldsymbol{\Gamma(q)}$ and the corresponding strain value of $\boldsymbol{q}$ for a given equilibrium end-effector position $\boldsymbol{p}_i \in \mathcal{W}_i$ which in fact is the well-known problem of inverse static model. The analytical formula of $\boldsymbol{\Gamma}_i$ has been deduced in subsection 4.3.1.1, and then we will present how to solve the inverse PLS Cosserat static model.

With the analytical formula of $\boldsymbol{\Gamma}(\boldsymbol{q})$ in (4.7), for a given $\boldsymbol{p}_i \in \mathcal{W}_i$, if we can solve the inverse statics of (4.2) to get the associated $\boldsymbol{q}_i$, then we can obtain the value of $\boldsymbol{\Gamma}_i$ for $\mathcal{W}_i$. Therefore, the following presents how to solve this inverse problem within an optimization framework.

For a given end-effector position $\boldsymbol{p}_{\mathrm{d}}$, the first intuitive approach to solve the inverse statics is to minimize the following cost function

$$\begin{aligned} &\underset{(\boldsymbol{q},\ \boldsymbol{T})}{\arg\min}\ \|\boldsymbol{p}(\boldsymbol{q}) - \boldsymbol{p}_{\mathrm{d}}\|_2^2 \\ &s.t. \quad \begin{cases} \underline{\boldsymbol{K}}\boldsymbol{q} + \underline{\boldsymbol{G}}(\boldsymbol{q}) = \underline{\boldsymbol{H}}\boldsymbol{T} \\ \boldsymbol{T} \in \boldsymbol{\mathcal{T}} \end{cases} \end{aligned} \tag{4.12}$$

with $\boldsymbol{p}(\boldsymbol{q}) = \mathbf{P}\boldsymbol{g}(\boldsymbol{q})\mathbf{E}$, where $\|\cdot\|$ denotes the standard Euclidean norm. Theoretically, multiple solutions might exist for the above optimization problem, which heavily depends on the initial guess values of $\boldsymbol{q}$ and $\boldsymbol{T}$.

In order to overcome the issue of multiple solutions, we propose to integrate the minimum potential energy principle into the minimal distance based cost function, since such a principle guarantees a unique configuration of any mechanical system in a static equilibrium. Motivated by this idea, for the given end-effector position $\boldsymbol{p}_{\mathrm{d}}$ with its associated strain configuration $\boldsymbol{q}$, the corresponding elastic potential energy $\mathcal{P}_E$ and gravitational potential energy $\mathcal{P}_G$ of total length L of the soft manipulator can be written as

$$\mathcal{P}_E = \frac{1}{2}\int_0^L \boldsymbol{q}^\top \boldsymbol{\Phi}^\top \boldsymbol{\Sigma}(X)\boldsymbol{\Phi}\boldsymbol{q}\mathrm{d}X$$

and

$$\mathcal{P}_G = \int_0^L \boldsymbol{\mathcal{O}}^\top(X)\boldsymbol{\mathcal{M}}(X)\boldsymbol{\mathcal{G}}\mathrm{d}X$$

where $\boldsymbol{\mathcal{O}}(s) = [\mathbf{0}_{3\times 1}^\top\ \boldsymbol{p}^\top(X)]^\top$ represents the position vector of each cross section with respect to the inertial frame. Consequently, the inverse problem is solved by minimizing the following cost function

$$\begin{aligned} &\underset{(\boldsymbol{q},\ \boldsymbol{T})}{\arg\min}\ \|\boldsymbol{p}(\boldsymbol{q}) - \boldsymbol{p}_{\mathrm{d}}\|_2^2 + \beta(\mathcal{P}_E + \mathcal{P}_G) \\ &s.t. \quad \begin{cases} \underline{\boldsymbol{K}}\boldsymbol{q} + \underline{\boldsymbol{G}}(\boldsymbol{q}) = \underline{\boldsymbol{H}}\boldsymbol{T} \\ \boldsymbol{T} \in \boldsymbol{\mathcal{T}} \end{cases} \end{aligned} \tag{4.13}$$

with β being the positive weight.

To handle the inequality constraint in (4.13), the nonlinear interior point method [135] is used. Specifically, adding the logarithmic barrier function $\boldsymbol{\mathcal{B}}$ to the objective function in order to remove the inequality constraints, the original NLP problem (4.13) can be then approximately re-formulated as

$$\begin{aligned} &\underset{\overline{\boldsymbol{x}}=(\boldsymbol{q},\ \boldsymbol{T},\ \boldsymbol{\mu})}{\arg\min} \ \|\boldsymbol{p}(\boldsymbol{q})-\boldsymbol{p}_{\mathrm{d}}\|_2^2+\beta(\mathcal{P}_E+\mathcal{P}_G)-\boldsymbol{\mu}\boldsymbol{\mathcal{B}} \\ &s.t. \quad \underline{\boldsymbol{K}}\boldsymbol{q}+\underline{\boldsymbol{G}}(\boldsymbol{q})=\underline{\boldsymbol{H}}\boldsymbol{T} \end{aligned} \tag{4.14}$$

with

$$\boldsymbol{\mu}=\begin{bmatrix}\mu_1 & \cdots & \mu_{2n_u}\end{bmatrix},\ \boldsymbol{\mathcal{B}}=\begin{bmatrix}\log(T^1-T^1_{\min})\\ \vdots \\ \log(T^{n_u}-T^{n_u}_{\min})\\ \log(T^1_{\max}-T^1)\\ \vdots \\ \log(T^{n_u}_{\max}-T^{n_u})\end{bmatrix},$$

where $\boldsymbol{\mu}$ is the barrier parameter vector. The Newton-type method is then used to find the optimal solution $\overline{\boldsymbol{x}}^*$ of such an optimization problem with the nonlinear equality constraints.

4.4.3 Global Estimation of $\boldsymbol{\Gamma}(\boldsymbol{q})$ via Neural Network

Inspired by the thought of Section 4.4.2, a series of Jacobian matrices corresponding to the representative end-effector positions of the m sub-workspaces are computed to construct a set of matrix basis, represented by $\overline{\boldsymbol{\Gamma}}=\begin{bmatrix}\boldsymbol{\Gamma}_1^\top & \boldsymbol{\Gamma}_2^\top & \cdots & \boldsymbol{\Gamma}_m^\top\end{bmatrix}^\top\in\mathbb{R}^{3m\times n_u}$, and the weight vector $\boldsymbol{\alpha}(\boldsymbol{p})=\begin{bmatrix}\alpha_1(\boldsymbol{p}) & \alpha_2(\boldsymbol{p}) & \cdots & \alpha_m(\boldsymbol{p})\end{bmatrix}\in\mathbb{R}^{1\times m}$ is then defined. Finally, we propose to use the matrix basis $\overline{\boldsymbol{\Gamma}}$ obtained from the PLS Cosserat model to globally approximate $\boldsymbol{\Gamma}(\boldsymbol{q})$ corresponding to the arbitrary end-effector position $\boldsymbol{p}$ while its associated weight vector $\boldsymbol{\alpha}(\boldsymbol{p})$ will be determined via a trained neural network. In other words, for any end-effector position $\boldsymbol{p}\in\mathcal{W}_E$ in the whole workspace, its associated Jacobian matrix approximation problem can be then transformed to a regression one, and it yields

$$\widehat{\boldsymbol{\Gamma}}(\boldsymbol{p})=\sum_{i=1}^{m}\alpha_i(\boldsymbol{p})\boldsymbol{\Gamma}_i$$

To obtain the value of $\boldsymbol{\alpha}(\boldsymbol{p})$, we use the radial basis function neural network (RBFNN) which is a commonly used three-layer feedforward network. Next, I will present how to approximate the parameter $\boldsymbol{\alpha}(\boldsymbol{p})$. According to the input-output relation (4.3), the regression problem can be then formulated as the following linear optimization one:

$$\underset{\boldsymbol{W}^{\mathrm{T}}}{\arg\min}\ f=\sum_{j=1}^{k}\|\boldsymbol{p}_j-\boldsymbol{p}_{j-1}-\big(\sum_{i=1}^{m}\alpha_i(\boldsymbol{p}_j)\boldsymbol{\Gamma}_i\big)(\boldsymbol{T}_j-\boldsymbol{T}_{j-1})\|_2^2 \tag{4.15}$$

with

$$\begin{aligned} \boldsymbol{\alpha}^\top(\boldsymbol{p}_j)&=\boldsymbol{W}^\top\overline{\boldsymbol{h}}(\boldsymbol{p}_j)\\ \overline{\boldsymbol{h}}(\boldsymbol{p}_j)&=\begin{bmatrix}h_1(\boldsymbol{p}_j) & h_2(\boldsymbol{p}_j) & \cdots & h_n(\boldsymbol{p}_j)\end{bmatrix}^\top\\ h_l(\boldsymbol{p}_j)&=\exp\left(-\frac{\|\boldsymbol{p}_j-\boldsymbol{c}_l\|^2}{2b_l^2}\right),\ l=1,2,\cdots,n. \end{aligned}$$

where $\boldsymbol{p}_j \in \mathbb{R}^3$ is the end-effector position vector selected, k represents the number of the training samples, $\boldsymbol{W}^\top \in \mathbb{R}^{m\times n}$ is the estimated weight matrix, n is the number of the neurons, and $\overline{\boldsymbol{h}}(\boldsymbol{p}_j) \in \mathbb{R}^n$ with $h_l(\boldsymbol{p}_j)$ being the Gaussian kernel function of the l^{th} neuron, $\boldsymbol{c}_l$ and b_l are the center and width of the Gaussian kernel function, respectively.

From the objective function of (4.15), we try to find the data pairs of $(\boldsymbol{T}_j,\ \boldsymbol{T}_{j-1})$ and $(\boldsymbol{p}_j,\ \boldsymbol{p}_{j-1})$. To generate the training data sets, we start from an equilibrium point j and change a little cable's tension from $\boldsymbol{T}_j$ to $\boldsymbol{T}_{j-1}$, and then the end-effector position will be changed from $\boldsymbol{p}_j$ to $\boldsymbol{p}_{j-1}$. In this way, we can obtain a set of training data. We can select k points and repeat this operation to generate multiple training data sets. Finally, the gradient descent method is used to solve the above optimization problem.

Remark 10. *It must be pointed out that one possible solution is to approximate the Jacobian matrix $\boldsymbol{\Gamma}(\boldsymbol{q})$ by using for example back propagation (BP) neural network as presented in [136]. Nevertheless, it is susceptible to getting stuck in local minima during the optimization process, and more prone to over-fitting, especially when the data is scarce. In this section, we use a hybrid strategy of both the PLS Cosserat model and RBFNN to calculate the approximation of $\boldsymbol{\Gamma}(\boldsymbol{q})$, which effectively ensures accurate control since it takes full advantage of the model information. On the one hand, the offline calculation method of the Jacobian matrix $\boldsymbol{\Gamma}(\boldsymbol{q})$ by using the RBFNN greatly improves control accuracy since RBFNN can avoid local minima, over-fitting, and better approximate the real Jacobian matrix. On the other hand, compared to the FEM-based gain-scheduling control of a soft trunk robot [137], this method globally approximates $\boldsymbol{\Gamma}(\boldsymbol{q})$ of any end-effector position $\boldsymbol{u} \in \mathcal{W}_E$, and allows the soft manipulator to track different time-varying trajectories in real time.*

4.4.4 Global Controller Design

Specifically, according to the deduced input-output relationship in (4.3), a robust controller (valid for the whole workspace) for the soft manipulator is developed in this subsection. The feedback control law is designed as follows

$$\dot{\boldsymbol{T}} = \widehat{\boldsymbol{\Gamma}}_R^{-1}(\boldsymbol{p})(-\boldsymbol{\lambda}_1\boldsymbol{e} - \boldsymbol{\lambda}_2\int_0^t \boldsymbol{e}\mathrm{d}s) \tag{4.16}$$

where $\boldsymbol{\lambda}_1 \in \mathbb{R}^{3\times3}$ and $\boldsymbol{\lambda}_2 \in \mathbb{R}^{3\times3}$ are constant diagonal, positive-definite matrices of feedback gains; $\widehat{\boldsymbol{\Gamma}}_R^{-1}$ represents the right inverse of $\widehat{\boldsymbol{\Gamma}}$; $\boldsymbol{e} = \boldsymbol{p}-\boldsymbol{p}_r$ with $\boldsymbol{p}_r$ being a slow reference end-effector position signal. With the introduced notations, (4.3) can be written as

$$\dot{\boldsymbol{e}} = -(\boldsymbol{\lambda}_1\boldsymbol{e} + \boldsymbol{\lambda}_2\int_0^t \boldsymbol{e}\mathrm{d}s) - \Delta\boldsymbol{\Gamma}\widehat{\boldsymbol{\Gamma}}_R^{-1}(\boldsymbol{\lambda}_1\boldsymbol{e} + \boldsymbol{\lambda}_2\int_0^t \boldsymbol{e}\mathrm{d}s) \tag{4.17}$$

with $\boldsymbol{\Gamma} = \widehat{\boldsymbol{\Gamma}} + \Delta\boldsymbol{\Gamma}$, where $\Delta\boldsymbol{\Gamma}$ represents the estimation error of the Jacobian matrix.

Assumption 4. *With enough samples to train the RBFNN described in Section 4.4.3, it is assumed that there exists a constant $\varepsilon > 0$ such that*

$$\|\boldsymbol{\Upsilon}\| \leqslant \varepsilon,\ \forall \boldsymbol{T} \in \boldsymbol{\mathcal{T}} \tag{4.18}$$

with $\boldsymbol{\Upsilon} = \begin{bmatrix} \Delta\boldsymbol{\Gamma}\widehat{\boldsymbol{\Gamma}}_R^{-1} & \boldsymbol{0}_3 \\ \boldsymbol{0}_3 & \Delta\boldsymbol{\Gamma}\widehat{\boldsymbol{\Gamma}}_R^{-1} \end{bmatrix} \in \mathbb{R}^{6\times6}$, *where $\|\boldsymbol{\Upsilon}\|$ represents the spectral norm of matrix $\boldsymbol{\Upsilon}$, and ε is a prescribed approximation accuracy error. It is worth noting that the smaller ε is, the more accurate Jacobian matrix we have obtained by using RBFNN.*

Theorem 3. *In terms of the studied soft manipulator described by (4.3), for given* $\boldsymbol{\lambda}_1$, $\boldsymbol{\lambda}_2$, *there exist symmetric positive-definite matrices* $\mathbf{P}$ *and* $\mathbf{Q}$ *that fulfill the Lyapunov equation*

$$\mathbf{P}\mathbf{A} + \mathbf{A}^\top \mathbf{P} = -2\mathbf{Q}, \tag{4.19}$$

with $\mathbf{A} = \begin{bmatrix} \mathbf{0}_3 & \mathbf{I}_3 \\ -\boldsymbol{\lambda}_2 & -\boldsymbol{\lambda}_1 \end{bmatrix}$, *if the approximation accuracy error* ε *satisfies*

$$\varepsilon \leqslant \frac{\lambda_{\min}(\mathbf{Q})}{\lambda_{\max}(\mathbf{P})\|\mathbf{B}\|}, \tag{4.20}$$

with $\mathbf{B} = \begin{bmatrix} \mathbf{0}_3 & \mathbf{0}_3 \\ -\boldsymbol{\lambda}_2 & -\boldsymbol{\lambda}_1 \end{bmatrix}$, *the proposed controller (4.16) can then drive any* $\boldsymbol{p} \in \mathcal{W}_E$ *to the desired constant position* $\boldsymbol{p}_r \in \mathcal{W}_E$, *i.e.,*

$$\lim_{t\to\infty} \|\boldsymbol{p}(t) - \boldsymbol{p}_r\|_2 = 0$$

Proof. To prove the stability of the system, we introduce the state variable: $\boldsymbol{\zeta} = \begin{bmatrix} \int_0^t \boldsymbol{e}\mathrm{d}s \\ \boldsymbol{e} \end{bmatrix}$. Then, the system (4.17) can be re-written as the following state-space representation:

$$\dot{\boldsymbol{\zeta}} = \mathbf{A}\boldsymbol{\zeta} + \boldsymbol{\Upsilon}\mathbf{B}\boldsymbol{\zeta}$$

Note that we want to prove $\boldsymbol{\zeta}(t) \to \mathbf{0}$ when $t \to \infty$, and this is equivalent to prove $V(\boldsymbol{\zeta}) \to 0$ when $t \to \infty$, where $V(\boldsymbol{\zeta})$ is a Lyapunov function defined as

$$V(\boldsymbol{\zeta}) = \boldsymbol{\zeta}^\top \mathbf{P}\boldsymbol{\zeta}$$

Then, the derivative of $V(\boldsymbol{\zeta})$ w.r.t. time t yields

$$\frac{\partial V}{\partial t} = \dot{V} = \boldsymbol{\zeta}^\top \mathbf{P}\dot{\boldsymbol{\zeta}} + \dot{\boldsymbol{\zeta}}^\top \mathbf{P}\boldsymbol{\zeta} = \boldsymbol{\zeta}^\top \left(\mathbf{P}\mathbf{A} + \mathbf{A}^\top \mathbf{P}\right)\boldsymbol{\zeta} + \boldsymbol{\zeta}^\top \left(\mathbf{P}\boldsymbol{\Upsilon}\mathbf{B} + \mathbf{B}^\top \boldsymbol{\Upsilon}^\top \mathbf{P}\right)\boldsymbol{\zeta}$$

Logically, if both (4.19) and (4.20) hold, we can conclude that

$$\begin{aligned}
\dot{V} &= -2\boldsymbol{\zeta}^\top \mathbf{Q}\boldsymbol{\zeta} + 2\boldsymbol{\zeta}^\top \mathbf{P}\boldsymbol{\Upsilon}\mathbf{B}\boldsymbol{\zeta} \leqslant -2\boldsymbol{\zeta}^\top \mathbf{Q}\boldsymbol{\zeta} + 2\|\mathbf{P}\boldsymbol{\Upsilon}\mathbf{B}\|\boldsymbol{\zeta}^\top\boldsymbol{\zeta} \\
&\leqslant -2\boldsymbol{\zeta}^\top \mathbf{Q}\boldsymbol{\zeta} + 2\lambda_{\max}(\mathbf{P})\|\boldsymbol{\Upsilon}\mathbf{B}\|\boldsymbol{\zeta}^\top\boldsymbol{\zeta} \leqslant -2\boldsymbol{\zeta}^\top \mathbf{Q}\boldsymbol{\zeta} + 2\frac{\lambda_{\max}(\mathbf{P})}{\lambda_{\min}(\mathbf{Q})}\|\boldsymbol{\Upsilon}\mathbf{B}\|\boldsymbol{\zeta}^\top \mathbf{Q}\boldsymbol{\zeta} \\
&\leqslant -2\left(1 - \frac{\lambda_{\max}(\mathbf{P})}{\lambda_{\min}(\mathbf{Q})}\|\boldsymbol{\Upsilon}\|\|\mathbf{B}\|\right)\frac{\lambda_{\min}(\mathbf{Q})}{\lambda_{\max}(\mathbf{P})}\boldsymbol{\zeta}^\top \mathbf{P}\boldsymbol{\zeta} \leqslant -2\left(1 - \varepsilon\frac{\lambda_{\max}(\mathbf{P})}{\lambda_{\min}(\mathbf{Q})}\|\mathbf{B}\|\right)\frac{\lambda_{\min}(\mathbf{Q})}{\lambda_{\max}(\mathbf{P})}V < 0
\end{aligned}$$

which guarantees that V exponentially converges to 0. Thus, we can derive that $\boldsymbol{\zeta}(t)$ exponentially converges to $\mathbf{0}$. □

4.4.5 Experimental Testing on the Soft Manipulator

First of all, we describe the parameters setting of workspace estimation, and parameter tuning of RBFNN-based optimization as well. Next, a series of experiments are conducted to demonstrate the effectiveness and robustness of the proposed controller. In the end, the experimental comparison between the local control scheme which is only valid in a certain sub-workspace and the global controller suitable for the whole workspace is carried out, and the related video demonstration is available [2].

[2]Video demonstration: `https://1drv.ms/v/s!Ak7hK-nVdKMei1so43zW4C57rJyZ?e=Cj4kF3`

4.4.5.1 Experimental Description

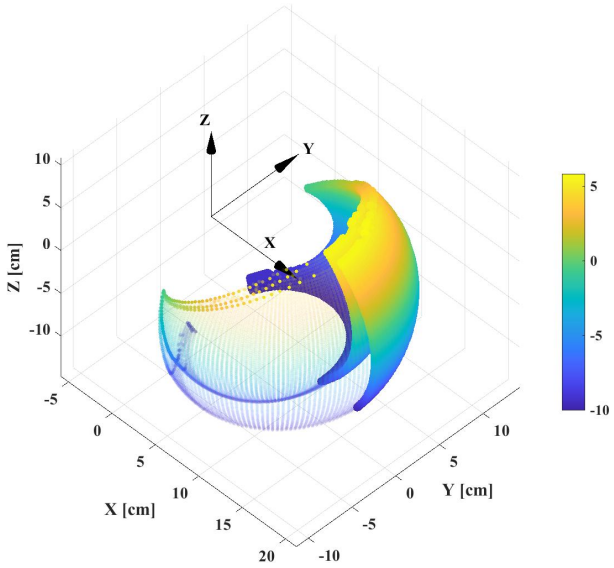


Figure 4.9: The whole workspace of the investigated soft manipulator.

Fig. 3.8b shows an overview of the control system of the soft manipulator. As for the optimization-based approach, we propose to discretize the angles with a discretization step size of 0.05 Radian (Unit), 200 successive rays with respective direction vector $\boldsymbol{\mathcal{E}}^{(i)}$, $i = 1, 2, \cdots, 200$ emanating at angular intervals of the angles $\gamma_x = \gamma_y = \gamma_z = 2\pi/200$ from the initial radiating point which is exterior to the workspace. In this case, the PLS Cosserat model is divided into three sections. Finally, the whole workspace of the cable-driven soft manipulator shown in Fig. 4.9 can be obtained.

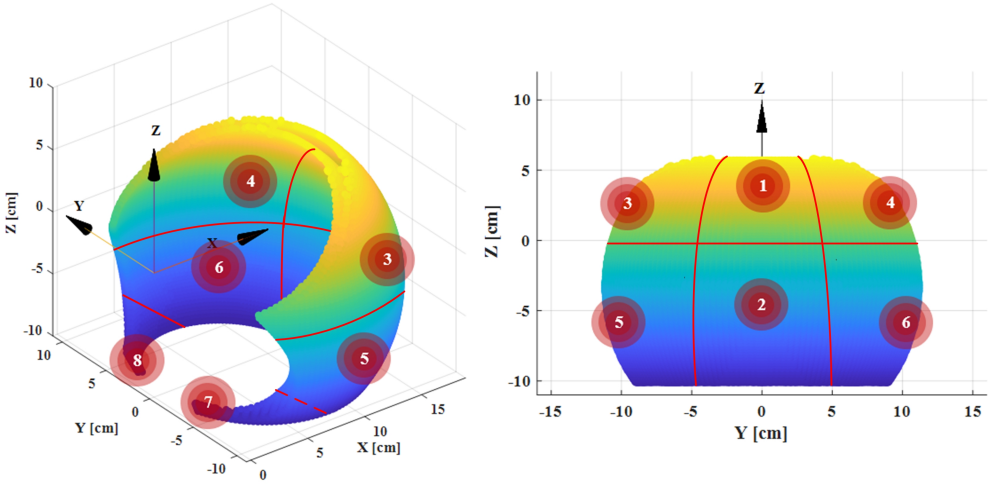


Figure 4.10: From the left, the distribution of the representative end-effector position selected from all sub-workspaces, followed by the front view of the representative positions.

According to the proposed decomposition rule (4.11), the whole workspace was finally divided into 8 sub-workspaces where the 8 representative end-effector position points (as illustrated in Fig. 4.10) that are respectively distributed in their corresponding sub-workspace can be chosen. Calculating their associated strain vector by using the inverse model, we can then acquire 8 Jacobian matrices to form a matrix basis for estimating (offline) $\boldsymbol{\Gamma}(\boldsymbol{q})$ corresponding to any end-effector position $\boldsymbol{p} \in \mathcal{W}_E$. In our test, the number of neurons required for the approximation of the matrix $\boldsymbol{\Gamma}(\boldsymbol{q})$ is determined by recording the achievable control performance with a given number of neurons, and then, increasing the number of neurons until the desired tracking effects are obtained. Initially, 9 neurons are used, and it is subsequently increased to 24 to get better performance. During the training process, the estimated weight matrix in (4.15) is initialized to zero, and the center of each neuron depends on the representative end-effector position $\boldsymbol{p}_i$. In order to well train the RBFNN, we tried to evenly choose 20 sample points in the whole workspace of the manipulator. For each sample point, we can then collect 4 groups of data pairs near itself by applying a small perturbation to each cable. Thus, 80 sets of data pairs can be obtained to train the RBFNN.

4.4.5.2 Experimental Testing

To evaluate the performance of the proposed controller with the Jacobian matrix approximation, different experiments including time-varying trajectory tracking tests and robustness analysis are conducted on the soft manipulator prototype.

1. Time-varying Trajectory Tracking Tests: In this test, we used the global controller to track circle and star-shaped slowly time-varying trajectories. The experimental results have been demonstrated in Fig. 4.11 and Fig. 4.12, clearly showing that the soft manipulator can rapidly track the different slowly time-varying trajectories. Basically, the proposed PLS Cosserat static model-based controller is feasible and effective for the slender soft manipulator to move in its global workspace.

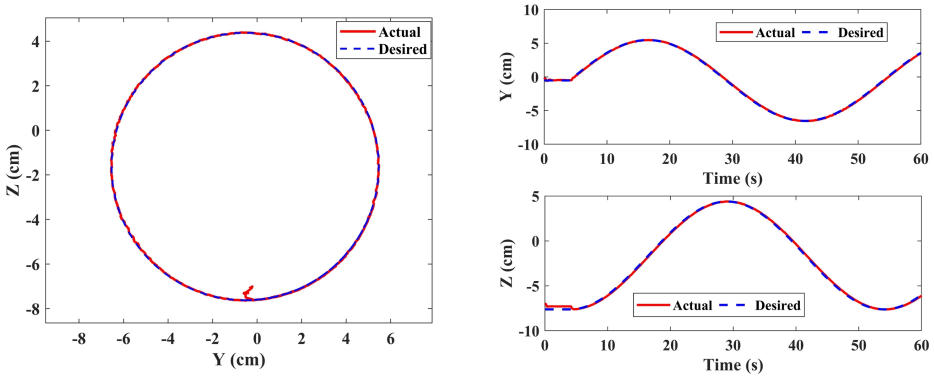


Figure 4.11: Circular time-varying trajectory in Y-Z plane by applying the global controller.

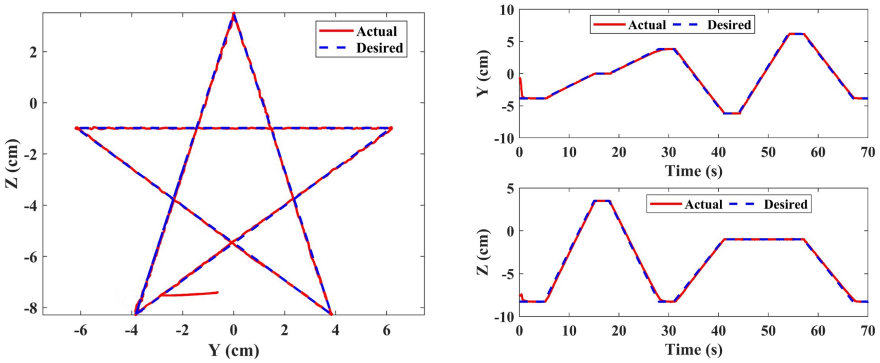


Figure 4.12: Star-shaped time-varying trajectory in Y-Z plane by using the global controller.

2. Robustness Analysis: In an effort to demonstrate the robustness of the proposed controller, a small Teenage Mutant Ninja Turtles (TMNT) which can be considered as an externally permanent disturbance hanged on the soft body. Simultaneously, the temporary disturbances are exerted on the soft arm with hand. The manipulator reaches the desired point $\boldsymbol{p}_r = \begin{bmatrix}15.44 & 4.33 & 3.40\end{bmatrix}^\top$ from the initial equilibrium point $\boldsymbol{p}_0 = \begin{bmatrix}14.83 & 0.00 & -7.53\end{bmatrix}^\top$. It can be clearly seen from the experimental results depicted in Fig. 4.13 that the proposed controller is robust in the sense that end-effector of the soft manipulator still returns to the desired position after a small fluctuation.

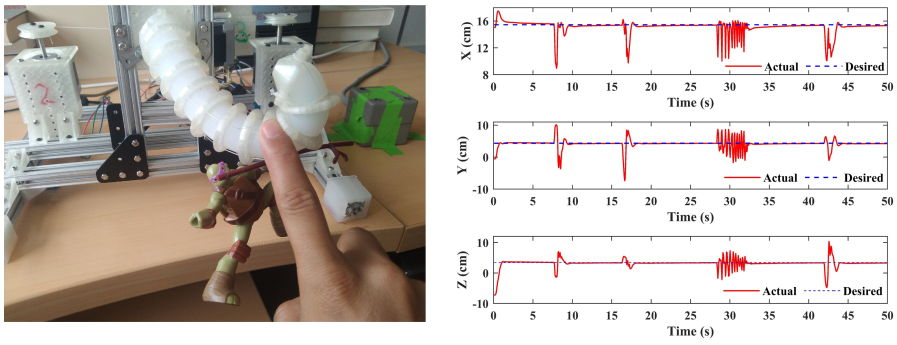


Figure 4.13: The robustness validation of the soft manipulator from one point to anther one under a combination of the temporary and permanent disturbances.

4.4.5.3 Comparison Results and Discussion

It can be observed that the circle and star-shaped trajectories for the global controller tracking are slightly smaller than those for the local controller tracking designed in Section 4.3, which may not fully highlight the strengths of the proposed control scheme despite its good tracking performances. Therefore, to intuitively demonstrate the advantages of the global controller, we have implemented a comparison of the tracking performance for the time-varying curve trajectory between the global control scheme and the local controller. The result indicates that the controller with the constant Jacobian matrix for the soft manipulator to track a slowly

time-varying curve trajectory around the whole workspace leads to the divergence of the system, as illustrated by the green line in Fig. 4.14. The tracking effect for the global controller proves to be excellent since the Jacobian matrices can be obtained when the end-effector position moves around the whole workspace. Thus, we can conclude that global tracking performance can be achieved by adding the RBFNN to the controller.

Through the tracking effect over the period of operation of the system, we have found that the fine motion control with the global controller is extremely useful for the positioning tasks since it enables all the capabilities of the soft manipulator to be exploited.

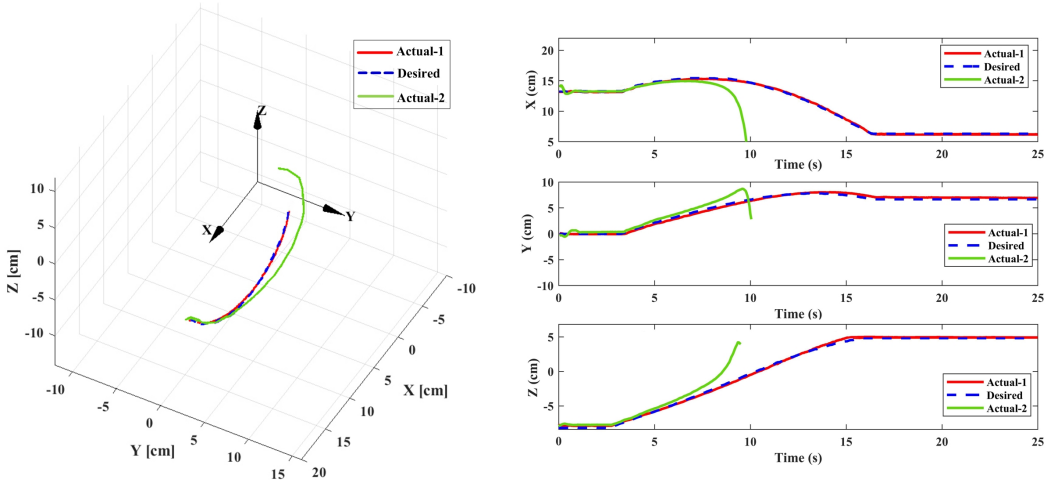


Figure 4.14: Tracking effects comparison of the time-varying curve trajectory in 3D space between the local controller (green line) with a constant Jacobian matrix [132] and the global controller (red line).

4.4.6 Discussion and Limitations

This section opens, for the first time, the possibility of controlling the soft manipulator via a hybrid approach of both the PLS Cosserat static model and RBFNN estimation scheme to approximate the Jacobian matrix between the task space and the actuator space of the manipulator in its whole workspace, based on which we designed a global controller to control the end-effector position of the soft manipulator. Experimental results for the soft manipulator following the fixed point in the presence of external disturbances or moving along different time-varying trajectories are presented to demonstrate the performance (i.e., feasibility and robustness). Additionally, the proposed control scheme is independent of any specific robot's cross section, and thus it can be easily adapted for controlling a wide range of soft manipulators.

Although the global control of $\boldsymbol{p}$ has been realized in this section, we need to rely on the RBFNN to approximate the Jacobian matrix $\boldsymbol{\Gamma}(\boldsymbol{q})$ within the whole workspace in advance. The classical pseudo-inverse method, based on the inverse of $\boldsymbol{\Gamma}(\boldsymbol{q})$, can also be applied to design the global controller. However, the real-time calculation of $\boldsymbol{\Gamma}(\boldsymbol{q})$ requires the knowledge of the generalized strain vector $\boldsymbol{q}$ which is practically difficult to be measured. To solve this issue, an alternative global control strategy with the state estimation for the soft manipulator based on PLS Cosserat static model will be presented in the following.

4.5 State Estimation-Based Control via PLS Cosserat Static Model

To achieve the global control of the studied soft robots, the techniques of measuring their states in real time will be necessary. Nevertheless, the conventional approaches are difficult or inappropriate to sense the strain vector. To this end, it is essential to put forward a state estimation strategy, which is still an inherently challenging issue due to the high dimensionality of the system states [138]. There are 2 different strategies which are possible to address this issue. The first is to use less marks, but need to design the observer via the PLS Cosserat dynamic model, while the second is to employ more markers to get the strain vector $\boldsymbol{q}$ via the PLS Cosserat output equations.

Throughout this section, we will propose a nonlinear programming scheme via the PLS Cosserat kinematic model to estimate the system states, which is online numerically implemented. Subsequently, the accuracy of the state estimation method is validated by the experimental test. Finally, the robust control based on state estimation can be achieved due to the real-time calculation of $\boldsymbol{\Gamma}(\boldsymbol{q})$.

To the best of our knowledge, it is the first work to combine the state estimation and control technique for the cable-driven soft manipulator via Cosserat rod theory, and it is also an extension to the aforementioned work.

4.5.1 Estimation of $\boldsymbol{q}$

It is noteworthy that the dimension of $\boldsymbol{q}$ is $6(N+1)$ if the soft manipulator is decomposed into N sections and all 6 strain components (3 linear strains and 3 angular strains) are used for each strain twist of the interpolation node for modeling via Cosserat rod theory. The dimension of each node can be in fact reduced to $3(N+1)$ if we allow only 2 angular rotation strains and 1 linear strain (i.e., the extensible Euler-Bernoulli beam). If neglecting the extension and shear modes, the dimension of each node can even decreased to $2(N+1)$, which corresponds to 3D Euler-Bernoulli beam. These strain mode choice schemes are widely employed in the literature [54, 130]. In this thesis, we only take into account 3 strain components (i.e., 2 bending modes and 1 extension mode) of each interpolation node to model the studied soft manipulator depicted in Fig. 3.8.

Consequently, it is possible to place $N+1$ position markers which can provide $3(N+1)$ measurements. In this case, the measurement model can be written as

$$\boldsymbol{\mathcal{Z}} = \boldsymbol{\psi}(\boldsymbol{q})$$

where the mapping $\boldsymbol{\psi}$ now is $\mathbb{R}^{3N+3} \rightarrow \mathbb{R}^{3N+3}$. By well placing the markers in an independent manner, it is then possible to solve the inverse of $\boldsymbol{\psi}$ to find $\boldsymbol{q}$ numerically. To tackle this problem, we denote $\overline{\boldsymbol{q}}$ as the estimation of $\boldsymbol{q}$, and define the corresponding estimation error as

$$\boldsymbol{e}_{\mathcal{Z}}(\overline{\boldsymbol{q}}) = \boldsymbol{\mathcal{Z}} - \boldsymbol{\psi}(\overline{\boldsymbol{q}}) \tag{4.21}$$

then the estimation problem is equivalent to solve the following optimization problem:

$$\arg\min_{\overline{\boldsymbol{q}}} \mathcal{S}(\overline{\boldsymbol{q}}) = \frac{1}{2}\boldsymbol{e}_{\mathcal{Z}}^{\top}\boldsymbol{e}_{\mathcal{Z}} \tag{4.22}$$

Note the gradient of the cost function $\mathcal{S}(\overline{\boldsymbol{q}})$ as $\nabla_{\overline{\boldsymbol{q}}}\mathcal{S}$, then the well-known finite-time gradient method [139] is used, and it is given by

$$\dot{\overline{\boldsymbol{q}}} = -\frac{\mathbf{H}^{-1}}{\|\nabla_{\overline{\boldsymbol{q}}}\mathcal{S}\|}\nabla_{\overline{\boldsymbol{q}}}\mathcal{S} \tag{4.23}$$

with $\mathbf{H} = \nabla^2_{\overline{\boldsymbol{q}}}\mathcal{S}$ being the Hessian matrix of $\mathcal{S}$, which can guarantee that $\overline{\boldsymbol{q}}$ converges to $\boldsymbol{q}$ in a finite time, i.e., there exists a positive constant $\mathcal{T}_1$ such that

$$\|\overline{\boldsymbol{q}}(t) - \boldsymbol{q}(t)\| = 0, \text{ if } t \geqslant \mathcal{T}_1.$$

Proof. To prove the finite-time stability, we define the following Lyapunov function

$$V = \frac{1}{2}\nabla_{\overline{\boldsymbol{q}}}\mathcal{S}^\top \nabla_{\overline{\boldsymbol{q}}}\mathcal{S}$$

Taking the derivative of V w.r.t. time t yields

$$\dot{V} = \nabla_{\overline{\boldsymbol{q}}}\mathcal{S}^\top \nabla^2_{\overline{\boldsymbol{q}}}\mathcal{S}\dot{\overline{\boldsymbol{q}}} = \nabla_{\overline{\boldsymbol{q}}}\mathcal{S}^\top \nabla^2_{\overline{\boldsymbol{q}}}\mathcal{S}\left(-\frac{\mathbf{H}^{-1}}{\|\nabla_{\overline{\boldsymbol{q}}}\mathcal{S}\|}\nabla_{\overline{\boldsymbol{q}}}\mathcal{S}\right) = -\frac{\|\nabla_{\overline{\boldsymbol{q}}}\mathcal{S}\|^2}{\|\nabla_{\overline{\boldsymbol{q}}}\mathcal{S}\|} = -\|\nabla_{\overline{\boldsymbol{q}}}\mathcal{S}\| = -\sqrt{2}V^{\frac{1}{2}}$$

i.e., $\dot{V} \leqslant -cV^\alpha$ with constant $c > 0$ and $\alpha < 1$. According to [140], exact convergence can be achieved in a finite time, i.e., $\overline{\boldsymbol{q}}(t) = \boldsymbol{q}(t)$ when $t \geqslant \mathcal{T}_1$. □

4.5.2 Accuracy Validation of State Estimation and Implementation of State Estimation-Based Control

The PLS Cosserat model with three sections for the soft manipulator is investigated. By using the strain mode choice scheme of the PLS Cosserat presented in [130], the reduced order model consists of 12 unknown strain variables, which include bending about Y-axis and Z-axis (Unit: 1/cm), and extension (or compression) along the X-axis (Unit: 1) for the strain twist of each interpolation node. Four magnetic position sensors are placed in an independent manner depicted in blue point in Fig. 3.8a to solve the above optimization problem.

To validate the accuracy of state estimation, the comparison between the selected positions obtained by the position sensors and those based on the state estimation has been performed. It is evident that the proposed state estimation method is capable of estimating the position of the soft manipulator with lower error, as shown in Fig. 4.15, especially for the end-effector position with negligible error.

Although this approach comes at the cost of placing more position sensors inside the soft manipulator, it can online compute the Jacobian matrix to globally control the studied soft manipulator. It should be emphasized that the estimation of the generalized strain vector $\boldsymbol{q}$ can also be applied to implement the model parameter identification schemes in Section 3.5.1 and Section 3.5.2 when $\boldsymbol{q}$ is not measurable, as the configuration of the soft manipulator for each test can be substituted with its estimation value.

Substituting the estimation of $\boldsymbol{q}$ into the analytical solution of Jacobian matrix, we can design another global controller, which is of the following form:

$$\dot{\boldsymbol{T}} = \boldsymbol{\Gamma}_R^{-1}(\overline{\boldsymbol{q}})(-\boldsymbol{\lambda}_1\boldsymbol{e} - \boldsymbol{\lambda}_2\int_0^t \boldsymbol{e}\mathrm{d}s)$$

This controller can achieve the global control of the end-effector position of the soft manipulator within its whole workspace. Here, we did not carry out the state estimation-based control experiments, as the performances of the designed controller will be validated by the estimation-based control via PLS Cosserat dynamics in the following chapter.

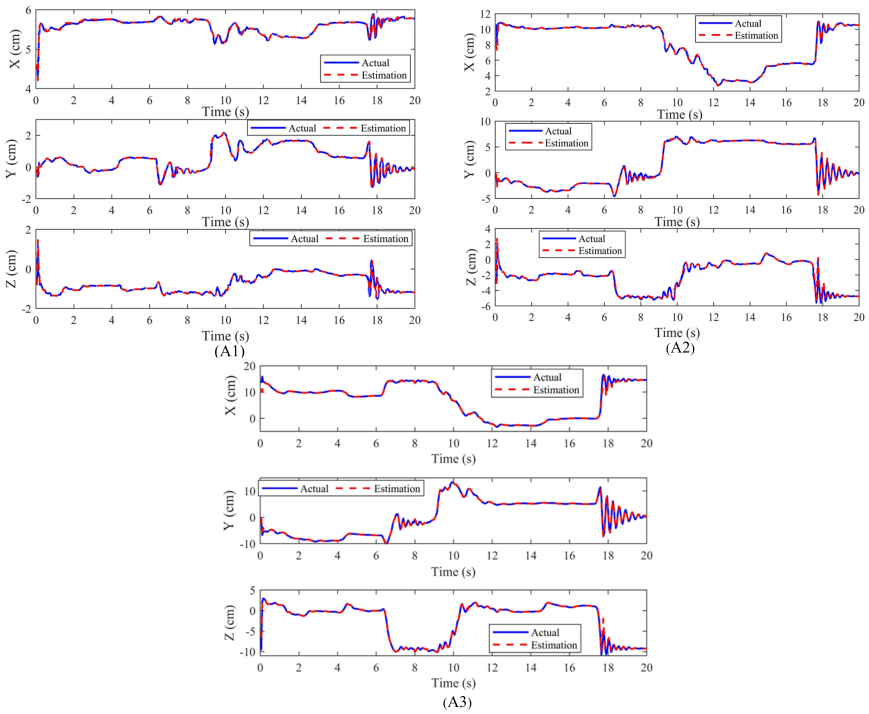


Figure 4.15: Comparison of sensor readings and estimations for the different positions of the soft manipulator at $X = 6.4$ cm (A1), $X = 12.8$ cm (A2), $X = 19.2$ cm (A3), respectively.

4.6 Conclusion

In this chapter, we first design a locally robust controller via the PLS Cosserat static model, which is only valid in a certain sub-workspace due to the constant Jacobian matrix used. To further achieve the control of the soft manipulator within its whole workspace, we unify the RBFNN and PLS Cosserat to globally approximate (offline) the Jacobian matrix $\boldsymbol{\Gamma}(\boldsymbol{q})$. Theoretical proofs and experimental results on the studied soft manipulator demonstrate that the proposed controllers can always drive the end-effector to the desired position, and have the great robustness to the externally bounded disturbances. However, all these control schemes allow the soft manipulator to track the trajectories with low velocity. That is logical because we have neglected the velocity and acceleration of the model.

Finally, the generalized strain vector $\boldsymbol{q}$ can be obtained by using the finite-time gradient method, and the experiments have been executed to validate the state estimation accuracy. By virtue of the estimated strain vector, we can online get the Jacobian matrix to realize the global control of the end-effector position $\boldsymbol{p}$.

Chapter 5

Cosserat-Based Dynamic Control of Slender Soft Manipulator

5.1 Introduction

The PLS modeling method combines the advantages of PCS and GVS to present the analytical solutions of the models and facilitate the control design. Based on PLS Cosserat static model, the local and global control of soft manipulator have been achieved in Chapter 4. However, these controllers can only allow the soft manipulator to track the slow signals since the influences of velocity and acceleration are neglected. To achieve fast motion control, the dynamic properties of the model will be taken into account in this chapter.

The central idea behind this chapter is to propose an estimation-based control framework to achieve the control of the soft manipulator via PLS Cosserat dynamics. Within this generic framework, the strain and end-effector position control of the soft manipulator will be treated as two special realizations. From control point of view, the slender soft manipulator control problem can be converted into a nonlinear output tracking problem. The proposed control methods enable the soft manipulator to rapidly and accurately track 3D configurations or time-varying trajectories, and the related video demonstration is available [1]. Our specific contributions include the following:

1. Based on the PLS Cosserat dynamic model, a general state estimation-based control architecture for the soft manipulator has been proposed with the rigorous mathematical proofs.
2. The strain and end-effector position nonlinear feedback control schemes via the established estimation-based control framework have been deduced.
3. Tracking performances of the designed controllers have been validated through the numerical simulations and experiments.

5.2 Problem Statement

Within the framework of PLS Cosserat dynamics developed in Chapter 3, the state variables include the generalized strain $\boldsymbol{q}$, time derivative of strain $\dot{\boldsymbol{q}}$ as well as cables' tension $\boldsymbol{T}$, and the

[1] Video demonstration: `https://1drv.ms/v/s!Ak7hK-nVdKMei1p0oUXoN75la9O6?e=sShvAS`

time derivative of cables' tension $\dot{\boldsymbol{T}}$ is regarded as the virtual system input. For the investigated soft manipulator, several cables are attached at the free end and go through the soft manipulator from its base to the tip, as shown in Fig. 3.8.

The properties of those matrices in (3.20) is of particular interest, where $\underline{\boldsymbol{M}}(\boldsymbol{q})$ is symmetric positive-definite, $\underline{\boldsymbol{C}}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ is Coriolis and centrifugal matrix, and $\overline{\boldsymbol{K}}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ models the internal and external wrenches of the soft manipulator, which depends on the constitutive law and mass of the soft material. $\overline{\boldsymbol{H}}(\boldsymbol{q})$ and $\overline{\boldsymbol{\mathcal{H}}}$ are all rectangular matrices, where the former is generalized actuation matrix, while the latter represents the directions of cables' tension due to the cables attached at the end cross section of the soft manipulator, and has non-zero values at the points where the actuators are applied (i.e., the rank of $\overline{\boldsymbol{\mathcal{H}}}$ is no greater than the number of the actuators). Therefore, the PLS dynamic system is a typically under-actuated system.

In practice, the objectives to be controlled might be various, which depends on the nature of the system, the available measurement provided by the sensor we used, and the desired behavior. For instance, for the soft manipulator described in Fig. 3.8, the objectives might include controlling the position, orientation, strain or acceleration. Additionally, other objectives might be inclusion of reducing the overall energy consumption of the system, or achieving a specific level of control precision. Motivated by this fact, the aim of this chapter is to establish a general control framework which covers all those abovementioned control objectives.

5.3 Output Tracking Formulation for Under-Actuated System

In this section, the system (3.20) is first reformulated in a canonical state-form suited to control design. This includes a model of sensors and controlled outputs. Then, the input-output map of the nonlinear under-actuated system is exactly linearized with a partial feedback linearization. After that, a control strategy for the resulting linear control system is presented to achieve various control objectives. In order to implement such a control law, several estimation approaches are proposed to estimate the system states in a finite time. Finally, a general estimation-based control scheme is proposed and its stability analysis is performed on several case studies.

5.3.1 State-Space Representation

By introducing the following state variable $\boldsymbol{\mathcal{X}} \in \mathcal{D} \subset \mathbb{R}^{n_x}$ with $n_x = 12(N+1) + n_u$ and control variable $\boldsymbol{u} \in \mathcal{U} \subset \mathbb{R}^{n_u}$:

$$\boldsymbol{\mathcal{X}} = \begin{bmatrix} \boldsymbol{\mathcal{X}}_1 \\ \boldsymbol{\mathcal{X}}_2 \\ \boldsymbol{\mathcal{X}}_3 \end{bmatrix} = \begin{bmatrix} \boldsymbol{q} \\ \dot{\boldsymbol{q}} \\ \boldsymbol{T} \end{bmatrix}, \boldsymbol{u} = \dot{\boldsymbol{T}} \tag{5.1}$$

the nonlinear dynamic system (3.20) can be formulated as the following state-space representation:

$$\dot{\boldsymbol{\mathcal{X}}} = \boldsymbol{f}(\boldsymbol{\mathcal{X}}) + \boldsymbol{\phi}(\boldsymbol{\mathcal{X}})\boldsymbol{u} = \boldsymbol{f}(\boldsymbol{\mathcal{X}}) + \sum_{i=1}^{n_u} \boldsymbol{\phi}_i(\boldsymbol{\mathcal{X}}) u_i \tag{5.2}$$

where

$$\boldsymbol{f}(\boldsymbol{\mathcal{X}}) = [f_1(\boldsymbol{\mathcal{X}}), \cdots, f_{n_x}(\boldsymbol{\mathcal{X}})]^\top = \begin{bmatrix} \boldsymbol{\mathcal{X}}_2 \\ -\underline{\boldsymbol{M}}^{-1}\underline{\boldsymbol{C}}\boldsymbol{\mathcal{X}}_2 + \underline{\boldsymbol{M}}^{-1}\overline{\boldsymbol{H}}\boldsymbol{\mathcal{X}}_3 + \underline{\boldsymbol{M}}^{-1}\overline{\boldsymbol{K}} \\ \boldsymbol{0} \end{bmatrix}$$

$$\boldsymbol{\phi}(\boldsymbol{\mathcal{X}}) = [\boldsymbol{\phi}_1(\boldsymbol{\mathcal{X}}), \cdots, \boldsymbol{\phi}_{n_u}(\boldsymbol{\mathcal{X}})] = \begin{bmatrix} \boldsymbol{0} \\ \underline{\boldsymbol{M}}^{-1}\overline{\boldsymbol{\mathcal{H}}} \\ \mathbf{I} \end{bmatrix}$$

Now suppose that various sensors have been equipped on the soft manipulator, providing the measurement noted as

$$\boldsymbol{\mathcal{Z}} = \boldsymbol{\psi}(\boldsymbol{\mathcal{X}}) \in \mathbb{R}^{n_z} \tag{5.3}$$

where $\boldsymbol{\psi} : \mathbb{R}^{n_x} \to \mathbb{R}^{n_z}$. It is assumed as well that the variables to be controlled are linearly independent and can be modeled as follows:

$$\boldsymbol{\mathcal{Y}} = \boldsymbol{h}(\boldsymbol{\mathcal{X}}) = \left[h_1(\boldsymbol{\mathcal{X}}), \cdots, h_{n_y}(\boldsymbol{\mathcal{X}})\right]^\top \in \mathbb{R}^{n_y} \tag{5.4}$$

where $\boldsymbol{h} : \mathbb{R}^{n_x} \to \mathbb{R}^{n_y}$.

Remark 11. *In the above modeling, sensors and the variables to be controlled are modeled in a quite general manner in order to cover as larger as possible the varieties. More precisely, the sensor could be:*

- *Strain sensor, if* $\boldsymbol{\psi}(\boldsymbol{\mathcal{X}}) = \boldsymbol{\mathcal{X}}_1$*;*
- *Tension sensor, if* $\boldsymbol{\psi}(\boldsymbol{\mathcal{X}}) = \boldsymbol{\mathcal{X}}_3$*;*
- *Position sensor, if* $\boldsymbol{\psi}(\boldsymbol{\mathcal{X}}) = \boldsymbol{p}(\boldsymbol{\mathcal{X}}, X)$*;*
- *Inertial Measurement Unit (IMU), if* $\boldsymbol{\psi}(\boldsymbol{\mathcal{X}}) = \boldsymbol{R}^{-1}\ddot{\boldsymbol{p}}$ *or* $\boldsymbol{\psi}(\boldsymbol{\mathcal{X}}) = (\boldsymbol{R}^{-1}\dot{\boldsymbol{R}})^\vee$ *where the symbol* $\vee$ *represents an operator about mapping a matrix into a vector;*

Also, the control objectives can also be various:

- *Position control, if* $\boldsymbol{h}(\boldsymbol{\mathcal{X}}) = \boldsymbol{p}(\boldsymbol{\mathcal{X}}_1, X)$*;*
- *Strain control, if* $\boldsymbol{h}(\boldsymbol{\mathcal{X}}) = \boldsymbol{\mathcal{X}}_1$*;*
- *Configuration control, if* $\boldsymbol{h}(\boldsymbol{\mathcal{X}}) = \boldsymbol{g}(\boldsymbol{\mathcal{X}}, X)$*;*
- *Velocity control, if* $\boldsymbol{h}(\boldsymbol{\mathcal{X}}) = [\boldsymbol{g}^{-1}(\boldsymbol{\mathcal{X}}, s)\dot{\boldsymbol{g}}(\boldsymbol{\mathcal{X}}, X)]^\vee$*.*

Indeed, there are additional control objectives that can be considered within the above modeling framework, such as contact force control or hybrid position-force control.

Note that system (5.2) is a nonlinear under-actuated system, therefore, given a pre-defined reference $\boldsymbol{\mathcal{Y}}_d$, with the available measurement $\boldsymbol{\mathcal{Z}}$, the soft manipulator control problem has been converted into a nonlinear output tracking one [141], i.e., design a controller $\boldsymbol{u}(t)$ based on the knowledge of $\boldsymbol{\mathcal{Z}}(t)$ such that the output $\boldsymbol{\mathcal{Y}}(t)$ of the nonlinear system (5.2) asymptotically converges to the desired reference $\boldsymbol{\mathcal{Y}}_d(t)$, i.e., $\lim_{t\to\infty} ||\boldsymbol{\mathcal{Y}}(t) - \boldsymbol{\mathcal{Y}}_d(t)|| = 0$.

In the following, we will combine various approaches developed in control community to solve the above nonlinear output tracking problem.

5.3.2 Partial Feedback Linearization

Let us first introduce differential geometric tools which will be used in the sequel to partially feedback linearize the studied system (5.2) and then design a controller.

For any function $h_i(\boldsymbol{\mathcal{X}})$ defined in (5.4), its Lie derivative in the direction of $\boldsymbol{f}(\boldsymbol{\mathcal{X}})$ is defined as

$$L_{\boldsymbol{f}} h_i = \frac{\partial h_i}{\partial \boldsymbol{\mathcal{X}}} \boldsymbol{f}(\boldsymbol{\mathcal{X}})$$

Iteratively, we can define the jth Lie derivative as $L^j_{\boldsymbol{f}} h_i = L_{\boldsymbol{f}} L^{j-1}_{\boldsymbol{f}} h_i$ for $j \geqslant 1$. Based on the above notation, the relative degree, for the purpose of measuring the number of times each output needs to be differentiated to have at least one of the inputs appears explicitly, is recalled below.

Definition 1. *For system (5.2), the relative degree of the ith output: $h_i(\boldsymbol{\mathcal{X}})$ defined in (5.4) with $1 \leqslant i \leqslant n_y$, is equal to r_i if the following conditions are satisfied for $\boldsymbol{\mathcal{X}} \in \mathcal{D}$ [142]:*

$$\begin{cases} L_{\boldsymbol{\phi}_k} L_{\boldsymbol{f}}^{j-1} h_i = 0, & \text{for all } 1 \leqslant k \leqslant n_u, 1 \leqslant j \leqslant r_i - 1 \\ L_{\boldsymbol{\phi}_k} L_{\boldsymbol{f}}^{r_i-1} h_i \neq 0, & \exists k, \text{ for } 1 \leqslant k \leqslant n_u \end{cases}$$

Then the relative degree for system (5.2) is $\{r_1, \cdots, r_{n_y}\}$.

For the soft manipulator modeled by (5.2), assuming that its relative degree is $[r] = \left[r_1, \cdots, r_{n_y}\right]$, thus we can define the following error variable between the actual output and the desired output:

$$\boldsymbol{\mathcal{E}} = \begin{bmatrix} \boldsymbol{\mathcal{E}}_1 \\ \vdots \\ \boldsymbol{\mathcal{E}}_{n_y} \end{bmatrix} = \begin{bmatrix} \mathcal{E}_{1,1} \\ \vdots \\ \mathcal{E}_{1,r_1} \\ \vdots \\ \mathcal{E}_{n_y,1} \\ \vdots \\ \mathcal{E}_{n_y,r_{n_y}} \end{bmatrix} = \begin{bmatrix} h_1 - \mathcal{Y}_{d_1} \\ \vdots \\ L_{\boldsymbol{f}}^{r_1-1} h_1 - \mathcal{Y}_{d_1}^{(r_1-1)} \\ \vdots \\ h_{n_y} - \mathcal{Y}_{d_{n_y}} \\ \vdots \\ L_{\boldsymbol{f}}^{r_{n_y}-1} h_{n_y} - \mathcal{Y}_{d_{n_y}}^{(r_{n_y}-1)} \end{bmatrix}$$

where $\boldsymbol{\mathcal{E}}_i = [\mathcal{E}_{i,1}, \cdots, \mathcal{E}_{i,r_i}]^\top$ with $\dot{\mathcal{E}}_{i,j-1} = \mathcal{E}_{i,j} = L_f^{j-1} h_i - \mathcal{Y}_{d_i}^{(j-1)}$ for $1 \leqslant i \leqslant n_y$ and $1 \leqslant j \leqslant r_i$.

It is clear that system (5.2) is under-actuated, which implies that no all states of $\boldsymbol{\mathcal{X}}$ can be controlled. This fact implicitly means $\sum_{i=1}^{n_y} r_i < n_x$. With the above defined $\boldsymbol{\mathcal{E}}$, as stated in [142], there always exists a complementary of $\boldsymbol{\mathcal{E}}$, noted as

$$\boldsymbol{\mathcal{Q}} = \boldsymbol{\nu}(\boldsymbol{\mathcal{X}}) \in \mathbb{R}^{n_x - \sum_{i=1}^{n_y} r_i}$$

such that the following variables

$$\begin{bmatrix} \boldsymbol{\mathcal{E}} \\ \boldsymbol{\mathcal{Q}} \end{bmatrix} = \boldsymbol{\Phi}(\boldsymbol{\mathcal{X}}) \in \mathbb{R}^{n_x} \tag{5.5}$$

form a diffeomorphism, i.e., $\boldsymbol{\Phi} : \mathbb{R}^{n_x} \to \mathbb{R}^{n_x}$ is bijective and differentiable, and its inverse $\boldsymbol{\Phi}^{-1}$ is also differentiable. With the deduced $\boldsymbol{\Phi}(\boldsymbol{\mathcal{X}})$, system (5.2) can be partially feedback linearized with an additional zero dynamics (i.e., states that are not observable from the output of the system) [142], which is of the following form:

$$\begin{aligned} \dot{\boldsymbol{\mathcal{E}}}_i &= \boldsymbol{A}_i \boldsymbol{\mathcal{E}}_i + \boldsymbol{B}_i \left[L_{\boldsymbol{f}}^{r_i} h_i - \mathcal{Y}_{d_i}^{(r_i)} + \textstyle\sum_{k=1}^{n_u} L_{\boldsymbol{\phi}_k} L_{\boldsymbol{f}}^{r_i-1} h_i u_k \right] \\ \dot{\boldsymbol{\mathcal{Q}}} &= \alpha_{\boldsymbol{f}}(\boldsymbol{\mathcal{E}}, \boldsymbol{\mathcal{Q}}) + \alpha_{\boldsymbol{\phi}}(\boldsymbol{\mathcal{E}}, \boldsymbol{\mathcal{Q}}) \boldsymbol{u} \end{aligned} \tag{5.6}$$

for $1 \leqslant i \leqslant n_y$ where

$$\boldsymbol{A}_i = \begin{bmatrix} 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \\ 0 & 0 & \cdots & 0 \end{bmatrix} \in \mathbb{R}^{r_i \times r_i}, \quad \boldsymbol{B}_i = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \in \mathbb{R}^{r_i}$$

and

$$\begin{aligned} \alpha_{\boldsymbol{f}}(\boldsymbol{\mathcal{E}}, \boldsymbol{\mathcal{Q}}) &= \tfrac{\partial \boldsymbol{\nu}(\boldsymbol{\mathcal{X}})}{\partial \boldsymbol{\mathcal{X}}} \boldsymbol{f}(\boldsymbol{\mathcal{X}})|_{\boldsymbol{\mathcal{X}} = \boldsymbol{\Phi}^{-1}(\boldsymbol{\mathcal{E}}, \boldsymbol{\mathcal{Q}})} \\ \alpha_{\boldsymbol{\phi}}(\boldsymbol{\mathcal{E}}, \boldsymbol{\mathcal{Q}}) &= \tfrac{\partial \boldsymbol{\nu}(\boldsymbol{\mathcal{X}})}{\partial \boldsymbol{\mathcal{X}}} \boldsymbol{\phi}(\boldsymbol{\mathcal{X}})|_{\boldsymbol{\mathcal{X}} = \boldsymbol{\Phi}^{-1}(\boldsymbol{\mathcal{E}}, \boldsymbol{\mathcal{Q}})} \end{aligned} \tag{5.7}$$

5.3.3 Control Design

It can be seen that system (5.6) contains n_y sub-systems of chain of integrator $\boldsymbol{\mathcal{E}}_i$ and a zero dynamics $\boldsymbol{\mathcal{Q}}$. To solve the output tracking problem originally posed for (5.2), it is indispensable to consider the last equation in each sub-system $\boldsymbol{\mathcal{E}}_i$, which can be written into the following compact one:

$$\left[\dot{\mathcal{E}}_{1,r_1},\cdots,\dot{\mathcal{E}}_{i,r_i},\cdots,\dot{\mathcal{E}}_{n_y,r_{n_y}}\right]^\top = \boldsymbol{\Gamma}_1(\boldsymbol{\mathcal{X}}) + \boldsymbol{\Gamma}_2(\boldsymbol{\mathcal{X}})\boldsymbol{u} \tag{5.8}$$

where

$$\boldsymbol{\Gamma}_1(\boldsymbol{\mathcal{X}}) = \left[L_{\boldsymbol{f}}^{r_1}h_1 - \mathcal{Y}_{d_1}^{(r_1)},\cdots,L_{\boldsymbol{f}}^{r_{n_y}}h_{n_y} - \mathcal{Y}_{d_{n_y}}^{(r_{n_y})}\right]^\top$$

and

$$\boldsymbol{\Gamma}_2(\boldsymbol{\mathcal{X}}) = \begin{bmatrix} L_{\boldsymbol{\phi}_1}L_{\boldsymbol{f}}^{r_1-1}h_1 & \cdots & L_{\boldsymbol{\phi}_{n_u}}L_{\boldsymbol{f}}^{r_1-1}h_1 \\ \vdots & \ddots & \vdots \\ L_{\boldsymbol{\phi}_1}L_{\boldsymbol{f}}^{r_{n_y}-1}h_{n_y} & \cdots & L_{\boldsymbol{\phi}_{n_u}}L_{\boldsymbol{f}}^{r_{n_y}-1}h_{n_y} \end{bmatrix} \in \mathbb{R}^{n_y\times n_u} \tag{5.9}$$

According to (5.8), it is obvious that, to achieve output tracking $\boldsymbol{\mathcal{Y}} \to \boldsymbol{\mathcal{Y}}_d$, the dimension of input should be greater or equal to that of the output, i.e., $n_u \geqslant n_y$. Therefore, the following assumption is imposed.

Assumption 5. *For the soft manipulator modeled by (5.2) (or equivalently by (5.6) and (5.8)), it is assumed that $n_u \geqslant n_y$ and the coupling matrix $\boldsymbol{\Gamma}_2(\boldsymbol{\mathcal{X}})$ defined in (5.9) is full row rank for all $\boldsymbol{\mathcal{X}} \in \mathcal{D}$, or equivalently there exists a right inverse matrix of $\boldsymbol{\Gamma}_2(\boldsymbol{\mathcal{X}})$, noted as $[\boldsymbol{\Gamma}_2(\boldsymbol{\mathcal{X}})]_R^{-1}$, such that $\boldsymbol{\Gamma}_2(\boldsymbol{\mathcal{X}})[\boldsymbol{\Gamma}_2(\boldsymbol{\mathcal{X}})]_R^{-1} = \mathbf{I}_{n_y}$.*

Remark 12. *The above mentioned assumption is not restrictive, which is attributed to the following two factors:*

1. *It is easy to determine that the number of the control inputs is not less than that of the variables to be controlled in advance when we install actuators. Otherwise, it is only needed to add more actuators;*
2. *The existence of the right inverse matrix of $\boldsymbol{\Gamma}_2(\boldsymbol{\mathcal{X}})$ is dependent on the way how we mount the cables along the soft manipulator. In general, the cables can be always installed in an independent manner such that the deduced matrix $\boldsymbol{\Gamma}_2(\boldsymbol{\mathcal{X}})$ is right invertible.*

Suppose that Assumption 5 is satisfied, we can then design the following controller:

$$\boldsymbol{u} = [\boldsymbol{\Gamma}_2(\boldsymbol{\mathcal{X}})]_R^{-1}\left[-\boldsymbol{\Gamma}_1(\boldsymbol{\mathcal{X}}) + \boldsymbol{v}\right] \tag{5.10}$$

where $\boldsymbol{\mathcal{X}} = \boldsymbol{\Phi}^{-1}(\boldsymbol{\mathcal{E}},\boldsymbol{\mathcal{Q}})$ and $\boldsymbol{v} = \left[v_1,\cdots,v_i,\cdots,v_{n_y}\right]^\top$ with

$$v_i = -\sum_{j=0}^{r_i-1} K_{i,r_i-j}\mathcal{E}_{i,r_i-j}, \forall 1 \leqslant i \leqslant n_y$$

Substituting (5.10) into the linear system of (5.6) yields the following linear closed-loop sub-system:

$$\begin{bmatrix} \dot{\mathcal{E}}_{i,1} \\ \vdots \\ \dot{\mathcal{E}}_{i,r_i-1} \\ \dot{\mathcal{E}}_{i,r_i} \end{bmatrix} = \begin{bmatrix} 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \\ -K_{i,1} & -K_{i,2} & \cdots & -K_{i,r_i} \end{bmatrix} \begin{bmatrix} \mathcal{E}_{i,1} \\ \vdots \\ \mathcal{E}_{i,r_i-1} \\ \mathcal{E}_{i,r_i} \end{bmatrix}$$

Thus, according to linear control theory [143], there always exist $K_{i,j}$ for $1 \leqslant i \leqslant n_y$ and $1 \leqslant j \leqslant r_i$ such that each closed-loop sub-system is stable. Consequently, we can obtain

$$\lim_{t\to\infty} ||\mathcal{E}_{i,1}(t)|| = \lim_{t\to\infty} ||\mathcal{Y}_i(t) - \mathcal{Y}_{d_i}(t)|| = 0,\ \forall\ 1 \leqslant i \leqslant n_y$$

which is equivalent to $\lim_{t\to\infty} ||\boldsymbol{\mathcal{Y}}(t) - \boldsymbol{\mathcal{Y}}_d(t)|| = 0$.

However, the realization of this controller requires that the state $\boldsymbol{\mathcal{X}}$ (or equivalently $\boldsymbol{\mathcal{E}}$ and $\boldsymbol{\mathcal{Q}}$) should be bounded, which depends on the stability of the zero dynamics of (5.6). To address this problem, let us consider the closed-loop system (5.6) and (5.10) when the output tracking has been achieved (i.e., $\boldsymbol{\mathcal{E}} = \mathbf{0}$ and $\boldsymbol{v} = \mathbf{0}$), which yields

$$\dot{\boldsymbol{\mathcal{Q}}} = \alpha_{\boldsymbol{f}}(\mathbf{0}, \boldsymbol{\mathcal{Q}}) - \alpha_{\boldsymbol{\phi}}(\mathbf{0}, \boldsymbol{\mathcal{Q}})[\boldsymbol{\Gamma}_2(\boldsymbol{\Phi}^{-1}(\mathbf{0}, \boldsymbol{\mathcal{Q}}))]_R^{-1}\boldsymbol{\Gamma}_1(\boldsymbol{\Phi}^{-1}(\mathbf{0}, \boldsymbol{\mathcal{Q}})) \tag{5.11}$$

If the above nonlinear dynamics is stable, then $\boldsymbol{\mathcal{Q}}$ is bounded, which ensures that the controller (5.10) can solve the output tracking problem. All the above analysis can be summarized in the following result.

Theorem 4. *Suppose that Assumption 5 is satisfied for the soft manipulator modeled by (5.2). If $\boldsymbol{\mathcal{X}}$ is known, then with the control defined in (5.10), the related output $\boldsymbol{\mathcal{Y}}$ will asymptotically converge to the desired reference $\boldsymbol{\mathcal{Y}}_d$ if system (5.11) is stable.*

Remark 13. *It is clear that the property of zero dynamics (5.11) is generally determined by the output model $\boldsymbol{h}(\boldsymbol{\mathcal{X}})$ which we want to control. However, for the soft manipulator modeled by (5.2), some choices of $\boldsymbol{h}(\boldsymbol{\mathcal{X}})$, such as strain or position, will naturally yield to a stable zero dynamics once the tracking error of strain or position converges to zero. This is due to the dissipativity of mechanical system, such as the viscosity of materials, air resistance, friction, etc.*

It is worth noting that, even if the zeros dynamics (5.11) is stable, the implementation of the controller (5.10) is not realizable in practice since it requires the knowledge of the state variable $\boldsymbol{\mathcal{X}}$, which contradicts the fact that only the input $\boldsymbol{T}$ (i.e., $\boldsymbol{\mathcal{X}}_3$) and $\boldsymbol{\mathcal{Z}}$ (which might be a part of $\boldsymbol{\mathcal{X}}$) are available. Therefore, the following investigates how to estimate $\boldsymbol{\mathcal{X}}_1$ and $\boldsymbol{\mathcal{X}}_2$ from the available measurement $\boldsymbol{\mathcal{Z}}$. In such a way, we will show that, by a simple modification, the deduced controller (5.10) can be still applied to solve the output tracking problem.

5.3.4 Estimation of $\boldsymbol{\mathcal{X}}$ from $\boldsymbol{\mathcal{Z}}$

According to the measurable output model, $\boldsymbol{\mathcal{Z}} = \boldsymbol{\psi}(\boldsymbol{\mathcal{X}})$, with the smooth mapping $\boldsymbol{\psi} : \mathbb{R}^{n_x} \to \mathbb{R}^{n_z}$, there exist various manners to estimate $\boldsymbol{\mathcal{X}}$.

The simplest case is that the mapping is invertible. In other words, if $\operatorname{rank}\frac{\partial \boldsymbol{\psi}}{\partial \boldsymbol{\mathcal{X}}} = n_x$, then the implicit theorem can guarantee that $\boldsymbol{\psi}$ is invertible. However, this condition requires that $n_z \geqslant n_x$, i.e., the number of measurable states should be larger or equal to the dimension of all states. Such a requirement is in fact difficult to be fulfilled. Indeed, for the soft manipulator modeled in (5.2), neither the generalized strain nor its derivative can be measured from sensors in practice. One solution to treat the estimation problem when $\operatorname{rank}\frac{\partial \boldsymbol{\psi}}{\partial \boldsymbol{\mathcal{X}}} < n_x$ is to take into account system's dynamic model. In this case, the state of system needs to be observable via the measurable output, where complex observability rank condition should be verified [142].

Note that for the soft manipulator modeled in (5.2), among the state variable $\boldsymbol{\mathcal{X}} = [\boldsymbol{\mathcal{X}}_1^\top, \boldsymbol{\mathcal{X}}_2^\top, \boldsymbol{T}^\top]^\top$, the cables' tension $\boldsymbol{T}$ is in fact known, thus we only need to estimate the strain $\boldsymbol{\mathcal{X}}_1$ and its derivative $\boldsymbol{\mathcal{X}}_2$ from the measurement $\boldsymbol{\mathcal{Z}}$. In addition, due to the fact that positions are the simplest signals to be measured (in this case the measurement $\boldsymbol{\mathcal{Z}}$ is only a function of $\boldsymbol{\mathcal{X}}_1$, i.e., $\boldsymbol{\mathcal{Z}} = \boldsymbol{\psi}(\boldsymbol{\mathcal{X}}_1)$), hence this section proposes a simple solution to solve the

estimation problem via the following two steps: firstly estimate $\boldsymbol{\mathcal{X}}_1$ from $\boldsymbol{\mathcal{Z}}$, noted as $\overline{\boldsymbol{\mathcal{X}}}_1$, which has been developed in Section 4.5.1, and then estimate $\boldsymbol{\mathcal{X}}_2$ from $\overline{\boldsymbol{\mathcal{X}}}_1$. The following presents the details how to achieve estimation of $\boldsymbol{\mathcal{X}}_2$ from $\overline{\boldsymbol{\mathcal{X}}}_1$.

It is noteworthy that $\boldsymbol{\mathcal{X}}_2 = \dot{\boldsymbol{\mathcal{X}}}_1$ as defined in (5.1), thus to estimate $\boldsymbol{\mathcal{X}}_2$, the intuitive way is to directly calculate its derivative via the so-called finite difference method [144]. However, such a method causes the noise existing in the signal for the real system to be amplified in the differentiation signal, leading to distort the differential of the extracted original signal. To address this issue, different techniques have been presented in the literature to design differentiator (either asymptotic or finite-time).

Generally speaking, the idea of differentiator is to use high-order time polynomial to approximate the input signal, and then get the model with bounded disturbance. Hence, the estimation of $\boldsymbol{\mathcal{X}}_1$ is considered as the input signal for which we want to calculate its high-order derivatives, and it is equivalent to

$$\overline{\boldsymbol{\mathcal{X}}}_1(t) = \sum_{i=1}^{n-1} \boldsymbol{a}_i t^i + \boldsymbol{\beta}(t) \in \mathbb{R}^{6(N+1)}$$

with $\boldsymbol{a}_i = \left[a_{i,1}, \cdots, a_{i,n_y}\right]^\top$ and $\boldsymbol{\beta}(t) = \left[\beta_1, \cdots, \beta_{n_y}\right]^\top$ where $\beta_i(t) \in \mathcal{C}^\infty$ thus is bounded. By introducing the new variable $\overline{\boldsymbol{\mathcal{X}}}_i = \dot{\overline{\boldsymbol{\mathcal{X}}}}_{i-1}$, which implies that $\overline{\boldsymbol{\mathcal{X}}}_i = \overline{\boldsymbol{\mathcal{X}}}_1^{(i-1)}$, then $\overline{\boldsymbol{\mathcal{X}}}_1$ can be regarded as the output of the following linear system with bounded disturbance:

$$\begin{bmatrix} \dot{\overline{\boldsymbol{\mathcal{X}}}}_1 \\ \vdots \\ \dot{\overline{\boldsymbol{\mathcal{X}}}}_{n-1} \\ \dot{\overline{\boldsymbol{\mathcal{X}}}}_n \end{bmatrix} = \underbrace{\begin{bmatrix} \mathbf{0} & \mathbf{I} & \cdots & \mathbf{0} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{I} \\ \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} \end{bmatrix}}_{\boldsymbol{A}} \begin{bmatrix} \overline{\boldsymbol{\mathcal{X}}}_1 \\ \vdots \\ \overline{\boldsymbol{\mathcal{X}}}_{n-1} \\ \overline{\boldsymbol{\mathcal{X}}}_n \end{bmatrix} + \begin{bmatrix} \mathbf{0} \\ \vdots \\ \mathbf{0} \\ \boldsymbol{\beta}^{(n)}(t) \end{bmatrix}$$

$$\overline{\boldsymbol{\mathcal{X}}}_1 = \underbrace{[\mathbf{I}, \mathbf{0}, \cdots, \mathbf{0}]}_{\boldsymbol{C}} \begin{bmatrix} \overline{\boldsymbol{\mathcal{X}}}_1 \\ \vdots \\ \overline{\boldsymbol{\mathcal{X}}}_{n-1} \\ \overline{\boldsymbol{\mathcal{X}}}_n \end{bmatrix}$$

Note that in the above system $\boldsymbol{\beta}^{(n)}(t)$ is bounded and the pair $(\boldsymbol{A}, \boldsymbol{C})$ is observable in the sense that the Kalman rank condition is satisfied. Consequently, different techniques, including high-gain observer [145], sliding mode observer [146] etc, have been proposed in the literature. Here, we adopt the following dynamics

$$\begin{aligned} \dot{\overline{\boldsymbol{\mathcal{X}}}}_1^* &= \overline{\boldsymbol{\mathcal{X}}}_2^* - k_1 \ulcorner \overline{\boldsymbol{\mathcal{X}}}_1^* - \overline{\boldsymbol{\mathcal{X}}}_1 \lrcorner^{\alpha} \\ &\vdots \\ \dot{\overline{\boldsymbol{\mathcal{X}}}}_{n-1}^* &= \overline{\boldsymbol{\mathcal{X}}}_n^* - k_{n-1} \ulcorner \overline{\boldsymbol{\mathcal{X}}}_1^* - \overline{\boldsymbol{\mathcal{X}}}_1 \lrcorner^{(n-1)\alpha-(n-2)} \\ \dot{\overline{\boldsymbol{\mathcal{X}}}}_n^* &= -k_n \ulcorner \overline{\boldsymbol{\mathcal{X}}}_1^* - \overline{\boldsymbol{\mathcal{X}}}_1 \lrcorner^{n\alpha-(n-1)} \end{aligned} \tag{5.12}$$

where $\ulcorner \boldsymbol{a} \lrcorner^b = |\boldsymbol{a}|^b \text{sign}(\boldsymbol{a})$ in the component-wise sense, k_i was chosen such that the roots of $s^n + k_1 s^{n-1} + \cdots + k_{n-1} s + k_n = 0$ have negative real parts, and the symbol $(\cdot)^*$ represents the estimation value of a vector or scalar. Depending on different choices of α, the above dynamics (5.12) is called a homogeneous differentiator [147] if $\alpha \in (\frac{n-1}{n}, 1)$, and it represents a high-order sliding mode differentiator [148] if $\alpha = \frac{n-1}{n}$.

It has been proven in [145] that, the aforementioned dynamics can provide the high-order derivative estimation of $\overline{\boldsymbol{\mathcal{X}}}_1$ in a finite time, i.e., there exists a positive constant $\mathcal{T}_2 > \mathcal{T}_1$ where $\mathcal{T}_1$ is defined in Section 4.5.1 such that

$$\|\overline{\boldsymbol{\mathcal{X}}}_1^*(t) - \overline{\boldsymbol{\mathcal{X}}}_1(t)\| = \|\overline{\boldsymbol{\mathcal{X}}}_2^*(t) - \dot{\overline{\boldsymbol{\mathcal{X}}}}_1(t)\| = 0, \text{ if } t \geqslant \mathcal{T}_2. \tag{5.13}$$

5.3.5 Estimation-Based Control and Stability Analysis

In summary, for a soft manipulator modeled by (5.2), with the available measurement from sensors modeled by (5.3), we can then calculate the estimation error via (4.21), based on which the states $\boldsymbol{\mathcal{X}}_1$ and $\boldsymbol{\mathcal{X}}_2$ can be accurately estimated by (4.23) and 5.12 in a finite time. Now denoting

$$\boldsymbol{\mathcal{X}}^* = \begin{bmatrix} \overline{\boldsymbol{\mathcal{X}}}_1^* \\ \overline{\boldsymbol{\mathcal{X}}}_2^* \\ \boldsymbol{T} \end{bmatrix} \tag{5.14}$$

we can then design the following feasible estimation-based controller:

$$\boldsymbol{u} = [\boldsymbol{\Gamma}_2(\boldsymbol{\mathcal{X}}^*)]_R^{-1} \left[-\boldsymbol{\Gamma}_1(\boldsymbol{\mathcal{X}}^*) + \boldsymbol{v}^* \right] \tag{5.15}$$

where $\boldsymbol{v}^* = \left[\widetilde{v}_1, \cdots, \widetilde{v}_i, \cdots, \widetilde{v}_{n_y}\right]^\top$ with

$$\widetilde{v}_i = -\sum_{j=0}^{r_i-1} K_{i,r_i-j} \widetilde{\mathcal{E}}_{i,r_i-j}, \forall 1 \leqslant i \leqslant n_y$$

with $\begin{bmatrix} \boldsymbol{\mathcal{E}}^* \\ \boldsymbol{\mathcal{Q}}^* \end{bmatrix} = \boldsymbol{\Phi}(\boldsymbol{\mathcal{X}}^*)$, which enables us to state the following result.

Theorem 5. *Suppose that Assumption 5 is satisfied for the soft manipulator modeled by (5.2). If (5.11) is stable, then the output tracking problem can be solved by the estimation-based controller (4.23), 5.12 and (5.15). The related output $\boldsymbol{\mathcal{Y}}$ will asymptotically converge to the desired reference $\boldsymbol{\mathcal{Y}}_d$, i.e., $\lim_{t\to\infty} ||\boldsymbol{\mathcal{Y}}(t) - \boldsymbol{\mathcal{Y}}_d(t)|| = 0$.*

Proof. The solvability of the output tracking problem via the proposed estimation-based controller is due to the finite-time convergence property (5.13), according to which when $t > \mathcal{T}_2$, we have $\boldsymbol{\mathcal{X}}^* = \boldsymbol{\mathcal{X}}$ and $\boldsymbol{v}^* = \boldsymbol{v}$ in (5.10) and (5.15). Therefore, the estimation-based controller (5.15) is quantitatively equivalent to the controller defined in (5.10) after $t > \mathcal{T}_2$. In addition, it has been shown in Section 5.3.3 that the output tracking problem can be solved via the proposed controller (5.10) when $t \to \infty$. Accordingly, the estimation-based control scheme (5.15) can also guarantee that $\lim_{t\to\infty} ||\boldsymbol{\mathcal{Y}}(t) - \boldsymbol{\mathcal{Y}}_d(t)|| = 0$. □

5.4 Strain and Position Control

Soft manipulators are widely used for applications such as grasping and manipulation. Generally, the realization of those tasks need to track a desired (Cartesian) position (in order to reach the object) and a desired shape parameterized with strain (in order to compensate the object's gravity). Therefore, accurate strain and position control are crucial in those applications. Within the developed estimation-based control framework, this section is devoted to realizing both strain and position control by following the result stated in Theorem 5, where the stability condition of zero dynamics (5.11) is naturally satisfied when considering strain and position control of soft manipulator, as stated in Remark 13.

5.4.1 Strain Control

In order to control the soft manipulator to track the desired strain via available position measurements, we can write the explicit sensor and output vector of the system (5.4) as

$$\begin{aligned} \boldsymbol{\mathcal{Y}} &= \boldsymbol{h}(\boldsymbol{\mathcal{X}}) = \boldsymbol{W}\boldsymbol{\mathcal{X}}_1 \\ \boldsymbol{\mathcal{Z}} &= \boldsymbol{\psi}(\boldsymbol{\mathcal{X}}_1) \end{aligned} \tag{5.16}$$

where $\boldsymbol{W} = \left[\boldsymbol{W}_1^\top, \cdots, \boldsymbol{W}_{n_y}^\top\right]^\top \in \mathbb{R}^{n_y \times 6(N+1)}$ is a matrix that selects the strains to be controlled.

Based on the PLS Cosserat nonlinear dynamics, we will focus on the relative degree of each strain output i for $1 \leqslant i \leqslant n_y$. Taking the time derivative of the first strain output in (5.16), it yields

$$\begin{aligned} \dot{h}_1(\boldsymbol{\mathcal{X}}) &= L_{\boldsymbol{f}} h_1 + \sum_{k=1}^{n_u} L_{\boldsymbol{\phi}_k} h_1 u_k = \boldsymbol{W}_1 \boldsymbol{\mathcal{X}}_2, \\ \ddot{h}_1(\boldsymbol{\mathcal{X}}) &= L_{\boldsymbol{f}}^2 h_1 + \sum_{k=1}^{n_u} L_{\boldsymbol{\phi}_k} L_{\boldsymbol{f}} h_1 u_k = \boldsymbol{W}_1 \underline{\boldsymbol{M}}^{-1} \left(-\boldsymbol{C}\boldsymbol{\mathcal{X}}_2 + \overline{\boldsymbol{K}} + \overline{\boldsymbol{H}}\boldsymbol{\mathcal{X}}_3\right) + \boldsymbol{W}_1 \underline{\boldsymbol{M}}^{-1} \overline{\boldsymbol{\mathcal{H}}} \boldsymbol{u} \end{aligned}$$

According to Definition 1, the relative degree of the first output equals to 2. By performing the above operation iteratively, we can obtain the relative degree of each output, which is equal to 2. Then, based on the developed framework in Section 5.3.2, the system (5.2) can be linearized as the following one

$$\begin{aligned} \dot{\mathcal{E}}_{1,1} &= L_{\boldsymbol{f}} h_1 - \dot{\mathcal{Y}}_{d_1} = \boldsymbol{W}_1 \boldsymbol{\mathcal{X}}_2 - \dot{\mathcal{Y}}_{d_1}, \\ \dot{\mathcal{E}}_{1,2} &= L_{\boldsymbol{f}}^2 h_1 - \ddot{\mathcal{Y}}_{d_1} + \sum_{k=1}^{n_u} L_{\boldsymbol{\phi}_k} L_{\boldsymbol{f}} h_1 u_k = \boldsymbol{W}_1 \underline{\boldsymbol{M}}^{-1} \left(-\boldsymbol{C}\boldsymbol{\mathcal{X}}_2 + \overline{\boldsymbol{K}} + \overline{\boldsymbol{H}}\boldsymbol{\mathcal{X}}_3\right) - \ddot{\mathcal{Y}}_{d_1} + \boldsymbol{W}_1 \underline{\boldsymbol{M}}^{-1} \overline{\boldsymbol{\mathcal{H}}} \boldsymbol{u} \\ &\vdots \\ \dot{\mathcal{E}}_{n_y,1} &= L_{\boldsymbol{f}} h_{n_y} - \dot{\mathcal{Y}}_{d_{n_y}} = \boldsymbol{W}_{n_y} \boldsymbol{\mathcal{X}}_2 - \dot{\mathcal{Y}}_{d_{n_y}}, \\ \dot{\mathcal{E}}_{n_y,2} &= L_{\boldsymbol{f}}^2 h_{n_y} - \ddot{\mathcal{Y}}_{d_{n_y}} + \sum_{k=1}^{n_u} L_{\boldsymbol{\phi}_k} L_{\boldsymbol{f}} h_{n_y} u_k \\ &= \boldsymbol{W}_{n_y} \underline{\boldsymbol{M}}^{-1} \left(-\boldsymbol{C}\boldsymbol{\mathcal{X}}_2 + \overline{\boldsymbol{K}} + \overline{\boldsymbol{H}}\boldsymbol{\mathcal{X}}_3\right) - \ddot{\mathcal{Y}}_{d_{n_y}} + \boldsymbol{W}_{n_y} \underline{\boldsymbol{M}}^{-1} \overline{\boldsymbol{\mathcal{H}}} \boldsymbol{u} \end{aligned}$$

Finally, the last equation of each sub-system is considered to deduce the relation between the output (i.e., strain error) and the input of the system, from which we can obtain the matrices $\boldsymbol{\Gamma}_1(\boldsymbol{\mathcal{X}})$ and $\boldsymbol{\Gamma}_2(\boldsymbol{\mathcal{X}})$, and it is equivalent to

$$\left[\dot{\mathcal{E}}_{1,2}, \cdots, \dot{\mathcal{E}}_{i,2}, \cdots, \dot{\mathcal{E}}_{n_y,2}\right]^\top = \boldsymbol{\Gamma}_1(\boldsymbol{\mathcal{X}}) + \boldsymbol{\Gamma}_2(\boldsymbol{\mathcal{X}})\boldsymbol{u}$$

where the matrices can be respectively formulated as

$$\begin{aligned} \boldsymbol{\Gamma}_1(\boldsymbol{\mathcal{X}}) &= L_{\boldsymbol{f}}^2 \boldsymbol{h} - \ddot{\boldsymbol{\mathcal{Y}}}_d = \boldsymbol{W} \underline{\boldsymbol{M}}^{-1} \left(-\boldsymbol{C}\boldsymbol{\mathcal{X}}_2 + \overline{\boldsymbol{K}} + \overline{\boldsymbol{H}}\boldsymbol{\mathcal{X}}_3\right) - \ddot{\boldsymbol{\mathcal{Y}}}_d, \\ \boldsymbol{\Gamma}_2(\boldsymbol{\mathcal{X}}) &= \sum_{k=1}^{n_u} L_{\boldsymbol{\phi}_k} L_{\boldsymbol{f}} \boldsymbol{h} = \boldsymbol{W} \underline{\boldsymbol{M}}^{-1} \overline{\boldsymbol{\mathcal{H}}} \end{aligned}$$

With the above deduced formulas, and suppose that Assumption 5 is satisfied, we can then design the proposed controller (5.15) where the finite-time estimations of $\boldsymbol{\mathcal{X}}_1$ and $\boldsymbol{\mathcal{X}}_2$ are achieved by the dynamics (4.23) and 5.12.

5.4.2 End-effector Position Control

For the second case, we want to control the end-effector's position of the investigated soft manipulator, via available position measurements, to track a fixed point or a rapidly time-varying trajectory. In this case, the sensor and the output of the soft manipulator can be formulated as

$$\begin{aligned} \boldsymbol{\mathcal{Y}} &= \boldsymbol{h}(\boldsymbol{\mathcal{X}}) = \boldsymbol{\mathcal{A}}\boldsymbol{p}(\boldsymbol{\mathcal{X}}_1, L) = \boldsymbol{\mathcal{A}}\mathbf{E}_1\, \boldsymbol{g}(\boldsymbol{\mathcal{X}}_1, L)\, \mathbf{E}_2 \\ \boldsymbol{\mathcal{Z}} &= \boldsymbol{\psi}(\boldsymbol{\mathcal{X}}_1) \end{aligned} \tag{5.17}$$

where $\boldsymbol{\mathcal{A}} = \left[\boldsymbol{\mathcal{A}}_1^\top, \boldsymbol{\mathcal{A}}_2^\top, \boldsymbol{\mathcal{A}}_3^\top\right]^\top \in \mathbb{R}^{3\times 3}$ is an identity matrix which is used to select the output position component, $\mathbf{E}_1$ and $\mathbf{E}_2$ are two matrices used to select the output position component $\boldsymbol{p}(\boldsymbol{\mathcal{X}}_1, L)$ from the pose of the manipulator's end-effector $\boldsymbol{g}(\boldsymbol{\mathcal{X}}_1, L)$.

Recalling that the continuous Cosserat kinematic model presented in Section 2.2.2, it is of the following form:

$$\dot{\boldsymbol{g}} = \boldsymbol{g}\widehat{\boldsymbol{\eta}}$$

where the symbol $\widehat{\ }$ changes a vector of $\mathbb{R}^6$ into a twist matrix of $\mathbb{R}^{4\times 4}$. Moreover, thanks to the PLS Cosserat assumption, the relations of the geometric Jacobian matrix $\boldsymbol{J}(\boldsymbol{\mathcal{X}}_1, L)$ to velocity twist $\boldsymbol{\eta}(\boldsymbol{\mathcal{X}}_1, L)$ and acceleration twist $\dot{\boldsymbol{\eta}}(\boldsymbol{\mathcal{X}}_1, L)$ arrive at

$$\begin{aligned} \boldsymbol{\eta}(\boldsymbol{\mathcal{X}}_1, L) &= \boldsymbol{J}(\boldsymbol{\mathcal{X}}_1, L)\, \boldsymbol{\mathcal{X}}_2, \\ \dot{\boldsymbol{\eta}}(\boldsymbol{\mathcal{X}}_1, L) &= \dot{\boldsymbol{J}}(\boldsymbol{\mathcal{X}}_1, L)\, \boldsymbol{\mathcal{X}}_2 + \boldsymbol{J}(\boldsymbol{\mathcal{X}}_1, L)\, \dot{\boldsymbol{\mathcal{X}}}_2 \end{aligned}$$

Taking the time derivative of the first end-effector position output in (5.17), we have

$$\begin{aligned} \dot{h}_1 &= L_{\boldsymbol{f}} h_1 + \sum_{k=1}^{n_u} L_{\boldsymbol{\phi}_k} h_1 u_k = \boldsymbol{\mathcal{A}}_1\mathbf{E}_1\boldsymbol{g}\widehat{\boldsymbol{\eta}}\mathbf{E}_2, \\ \ddot{h}_1 &= L_{\boldsymbol{f}}^2 h_1 + \sum_{k=1}^{n_u} L_{\boldsymbol{\phi}_k} L_{\boldsymbol{f}} h_1 u_k = \boldsymbol{\mathcal{A}}_1\mathbf{E}_1\boldsymbol{g}(\boldsymbol{\mathcal{X}}_1, L)\big(\widehat{\dot{\boldsymbol{\eta}}}(\boldsymbol{\mathcal{X}}_1, L)\mathbf{E}_2 + \widehat{\boldsymbol{\eta}}^2(\boldsymbol{\mathcal{X}}_1, L)\mathbf{E}_2\big) \\ &= \boldsymbol{\mathcal{A}}_1\mathbf{E}_1\boldsymbol{g}\mathbf{Q}\left[\dot{\boldsymbol{J}}\, \boldsymbol{\mathcal{X}}_2 + \boldsymbol{J}\left(-\underline{\boldsymbol{M}}^{-1}\underline{\boldsymbol{C}}\boldsymbol{\mathcal{X}}_2 + \underline{\boldsymbol{M}}^{-1}\overline{\boldsymbol{H}}\boldsymbol{\mathcal{X}}_3 + \underline{\boldsymbol{M}}^{-1}\overline{\boldsymbol{K}}\right)\right] \\ &\quad + \boldsymbol{\mathcal{A}}_1\mathbf{E}_1\boldsymbol{g}\widehat{\boldsymbol{\eta}}^2\mathbf{E}_2 + \boldsymbol{\mathcal{A}}_1\mathbf{E}_1\boldsymbol{g}\mathbf{Q}\boldsymbol{J}\underline{\boldsymbol{M}}^{-1}\overline{\boldsymbol{\mathcal{H}}}\boldsymbol{u} \end{aligned} \tag{5.18}$$

with $\mathbf{Q} = \begin{bmatrix} \mathbf{0}_3 & \mathbf{I}_3 \\ \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} \end{bmatrix}$. Following the above steps and taking the time derivative of each position output, then we can obtain that the relative degree of each output i for $1 \leqslant i \leqslant 3$ equals to 2.

By using the feedback linearization framework in Section 5.3.2, the relation between the output (i.e., end-effector position error) and the input of the system can be written as

$$\left[\dot{\mathcal{E}}_{1,2},\ \dot{\mathcal{E}}_{2,2},\ \dot{\mathcal{E}}_{3,2}\right]^\top = \boldsymbol{\Gamma}_1(\boldsymbol{\mathcal{X}}) + \boldsymbol{\Gamma}_2(\boldsymbol{\mathcal{X}})\boldsymbol{u}$$

where

$$\begin{aligned} \boldsymbol{\Gamma}_1(\boldsymbol{\mathcal{X}}) &= L_{\boldsymbol{f}}^2\boldsymbol{h} - \ddot{\boldsymbol{\mathcal{Y}}}_d \\ &= \mathbf{E}_1\boldsymbol{g}\mathbf{Q}\left[\dot{\boldsymbol{J}}\, \boldsymbol{\mathcal{X}}_2 + \boldsymbol{J}\left(-\underline{\boldsymbol{M}}^{-1}\underline{\boldsymbol{C}}\boldsymbol{\mathcal{X}}_2 + \underline{\boldsymbol{M}}^{-1}\overline{\boldsymbol{H}}\boldsymbol{\mathcal{X}}_3 + \underline{\boldsymbol{M}}^{-1}\overline{\boldsymbol{K}}\right)\right] + \mathbf{E}_1\boldsymbol{g}\widehat{\boldsymbol{\eta}}^2\mathbf{E}_2 - \ddot{\boldsymbol{\mathcal{Y}}}_d, \\ \boldsymbol{\Gamma}_2(\boldsymbol{\mathcal{X}}) &= \sum_{k=1}^{n_u} L_{\boldsymbol{\phi}_k} L_{\boldsymbol{f}}\boldsymbol{h} = \mathbf{E}_1\boldsymbol{g}\mathbf{Q}\boldsymbol{J}\underline{\boldsymbol{M}}^{-1}\overline{\boldsymbol{\mathcal{H}}} \end{aligned}$$

Again, by incorporating the dynamics described in (4.23) and 5.12, we can obtain finite-time estimations of $\boldsymbol{\mathcal{X}}_1$ and $\boldsymbol{\mathcal{X}}_2$. Then, the above derived formulas can be utilized in designing the proposed controller (5.15) provided that Assumption 5 is met, to achieve end-effector position control.

Remark 14. *Compared to the static model-based controller for the end-effector position control [132], the proposed strategy can be used to deal with control objectives regarding fast trajectory tracking since it takes fully into account the influence of the velocity and acceleration of the model.*

5.5 Simulation Tests

A conical soft manipulator which is actuated by 4 cables anchored at the free end and passed through the rigid rings embedded inside the manipulator is studied. Besides, the material we used for the soft slender manipulator is isotropic silicone with the physical parameters listed in Table 5.1.

Table 5.1: Geometric and material parameters of the studied soft arm

Parameters	Description	Value	Unit
L	Total Length	0.20	m
E	Young's modulus	3.0×10^5	Pa
G	Shear modulus	1.035×10^5	Pa
ρ	Density	2.0×10^3	kg/m^3
$R_{\max}$	Base radius	1.3×10^{-2}	m
$R_{\min}$	Tip radius	5.0×10^{-3}	m
μ	Viscosity coefficient	300	Pa·s

Considering the trade-off between computational cost and accuracy, the soft manipulator should been carefully divided. The criteria about how to determine the number of section and disc, one can refer to Remark 2. Specifically, the model of the investigated soft manipulator is divided into three-section variable length: $L_1 = 9 \times 10^{-2}$ m, $L_2 = 7 \times 10^{-2}$ m, $L_3 = 4 \times 10^{-2}$ m, and totally discretized into 21 discs. The trunk-like soft manipulator actuated by cables is graphically discretized, as shown in Fig. 5.1, in which the three sections are composed of discs with different colors, respectively.

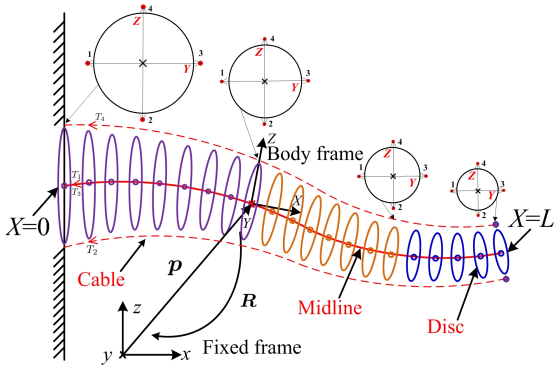


Figure 5.1: Representation of the discretized PLS Cosserat model actuated by cables.

5.5.1 Strain Control

To assess the potential interest of using the dynamic control law, the performance validation of the PLS Cosserat dynamic model-based controller for strain tracking is conducted. The soft slender manipulator under gravity tracks the specified strain through bending, shear, extension or a combination of these deformations in 3D space to realize the control purpose.

In the simulation, we define $\boldsymbol{\mathcal{Y}}_d(t)$ as a time-varying reference strain trajectory of the last disc at $X = L$, which can be generated by the following parametric equation:

$$\boldsymbol{\mathcal{Y}}_d(t) = \boldsymbol{\mathcal{Y}}_i + S(t)(\boldsymbol{\mathcal{Y}}_f - \boldsymbol{\mathcal{Y}}_i) \tag{5.19}$$

with

$$S(t) = 2\left(\frac{1}{1+e^{-\widetilde{p}t}} - \frac{1}{2}\right)$$

where $\boldsymbol{\mathcal{Y}}_i$, as the initial strain only under gravity, and $\boldsymbol{\mathcal{Y}}_f$, as the final strain under the combined effects of gravity and cables' tension, can be calculated by using the Gaussian-Newton method; $S(t)$ is a smooth and easily derivative sigmoid function with $\widetilde{p}$ being convergence rate of the defined sigmoid curve.

For the simulation setup, we first determine the final strain of the soft manipulator under gravity and the cables' tension $\boldsymbol{T} = [0, 0, 2, 1]^\top$. In detail, the sampling time dt is 0.01 s, and convergence rate $\widetilde{p}$ equals to 2. Then, the state variables to be controlled are selected, which includes bending about Y-axis and Z-axis (noted as q_a and q_b) and extension along the X-axis (noted as q_c) of the last disc at $X = L$. As for the controller, we set $K_{i,j}$ for $1 \leqslant i \leqslant 3$ and $1 \leqslant j \leqslant 2$ to the appropriate values such that each closed-loop sub-system is stable. With these parameters setting, the closed-loop control system is simulated for a duration of 20 s, and the corresponding simulation results are then obtained.

Fig. 5.2(a) reports the result that the time evolution of the controlled strain components error between the simulated and reference signal rapidly converges to zero under this controller, whereas Fig. 5.2(b) illustrates that the controlled strain components can well track the time-varying reference ones, and quickly converge to the targeted strain components. Furthermore, Fig. 5.3 (a-c) plots the snapshots of the simulated silicone manipulator actuated by the cables tracking the desired trajectory going from the initial strain under gravity to final stable state under the combination of gravity and the cables' tension, which accords with the good tracking of Fig. 5.2. All in all, the control results demonstrate that the strain tracking controller has effective tracking performance in term of control precision and stability.

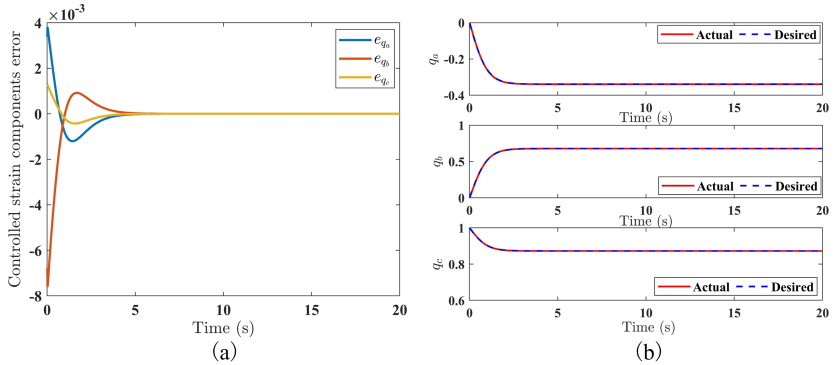


Figure 5.2: From the left, time evolution of the trajectory tracking of the controlled strain components, followed by tracking effects of the controlled ones by using the proposed controller.

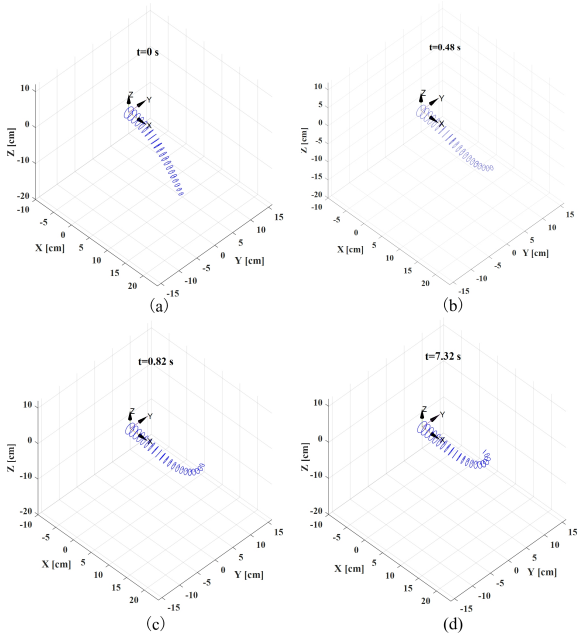


Figure 5.3: Simulation snapshots at time $t = 0$ s, 0.48 s, 0.82 s, and 7.32 s, respectively.

5.5.2 End-effector Position Control

The performance of the end-effector position controller will be tested by the following four sets of numerical simulation tests.

5.5.2.1 Point-to-point control

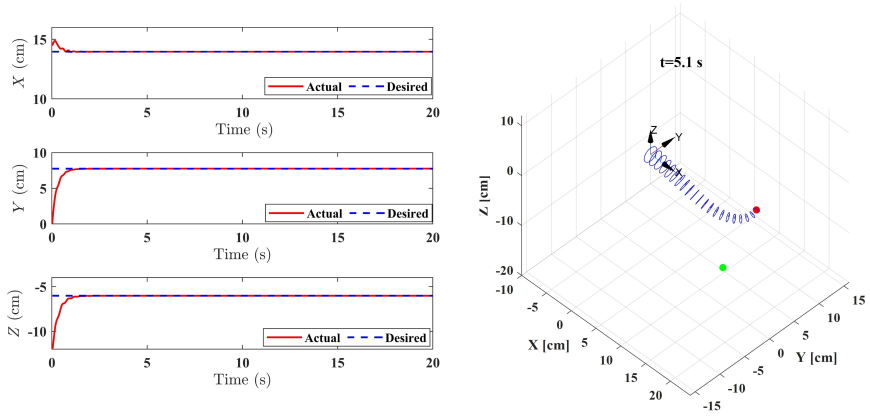


Figure 5.4: From the left, the results of point-to-point test using the proposed controller, followed by the simulation snapshot at time $t = 5.1$ s.

The simulation test is implemented where the objective is to control the end-effector of the soft manipulator from the position only under gravity to its deformed reference position. In

simulation, we set the sampling time to $\mathrm{d}t = 0.02$ s, and the cables' tension (corresponding to the deformed arm's end-effector position) to $\boldsymbol{T} = [0, 0, 1, 1]^\top$. The controller tried to drive the end-effector to reach the reference point. Fig. 5.4 presents a good tracking of the reference constant point, and intuitively plots snapshot of the simulated soft manipulator from the initial point (the green) to the targeted one (the red) using the proposed controller.

5.5.2.2 Straight-line trajectory tracking

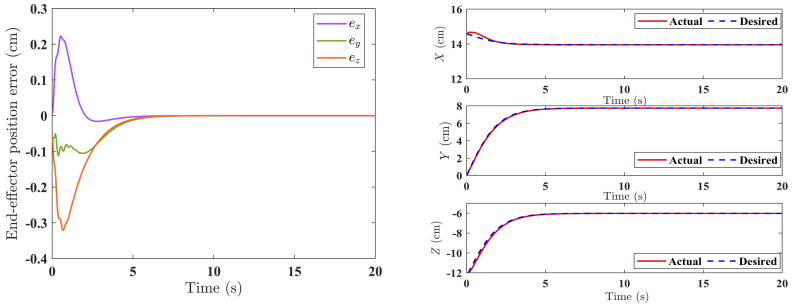


Figure 5.5: Time-varying straight-line trajectory tracking using the proposed controller.

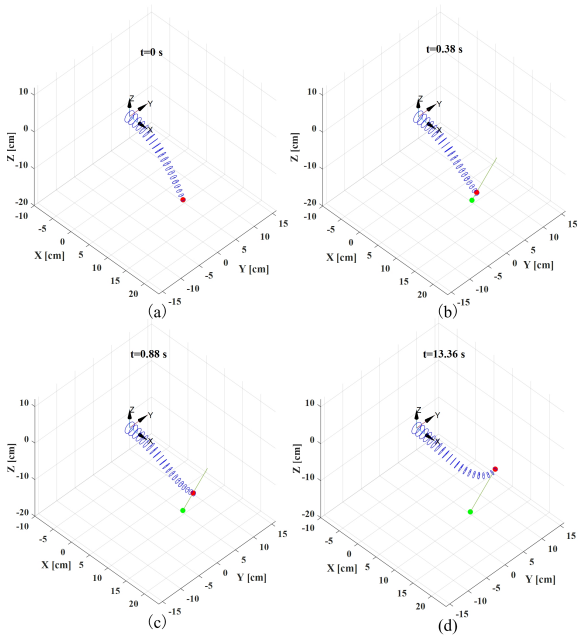


Figure 5.6: Simulation snapshots at time $t = 0$ s, 0.38 s, 0.88 s, and 13.36 s, respectively.

The simulated soft manipulator was expected to track a time-varying straight line trajectory, whose parametric equation is identical to the form of (5.19). Let us define $\boldsymbol{\mathcal{Y}}_i$ as the initial end-effector position of the robot under gravity, while $\boldsymbol{\mathcal{Y}}_f$ as the final position under a combination

of gravity and cables' tension. We set the sampling time to $\mathrm{d}t = 0.02$ s, the cables' tension to $\boldsymbol{T} = [0, 0, 1, 1]^\top$, and convergence rate to $\widetilde{p} = 1$.

Fig. 5.5 shows the time evolution of the arm's end-effector position errors between the desired and actual trajectories. It is evident that the controller allows p_x, p_y and p_z for the end-effector of the soft manipulator to quickly catch up with their reference counterparts. A set of snapshots of the simulation test is displayed in Fig. 5.6 where the green point is the start while the red one is a moving point along the straight-line trajectory.

5.5.2.3 Circular trajectory tracking

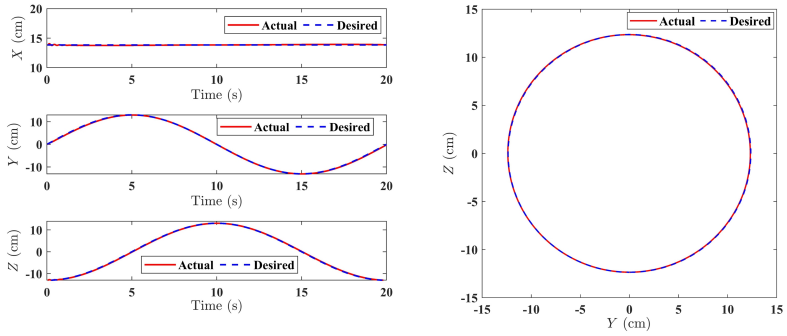


Figure 5.7: Circular trajectory tracking using the proposed controller.

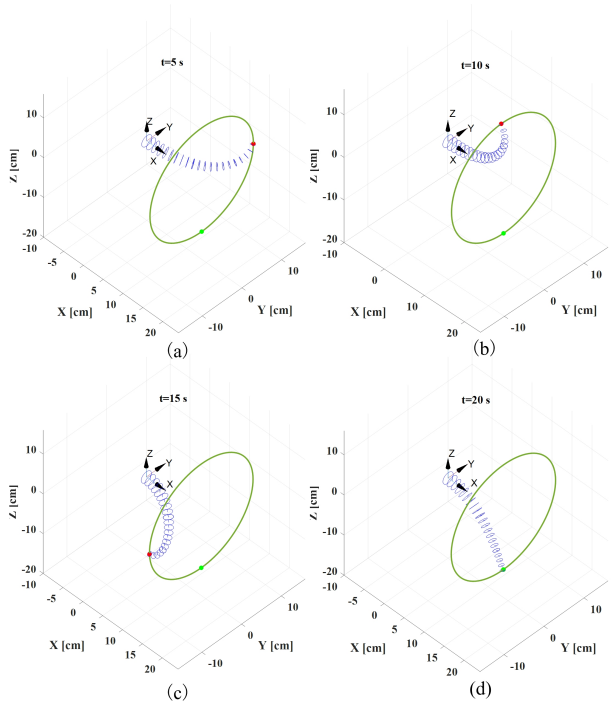


Figure 5.8: Simulation snapshots at time $t = 5$ s, 10 s, 15 s and 20 s, respectively.

In the third simulation, the end-effector of the manipulator was expected to follow the time-varying circular trajectory in order to further validate the tracking performance of the controller. We set the sampling time to dt = 0.02 s. The result of the tracking problem is reported in Fig. 5.7 and Fig. 5.8, which illustrates that the proposed position tracking controller allows the soft manipulator to stably track the desired circular trajectory.

5.5.2.4 Star-shaped trajectory tracking

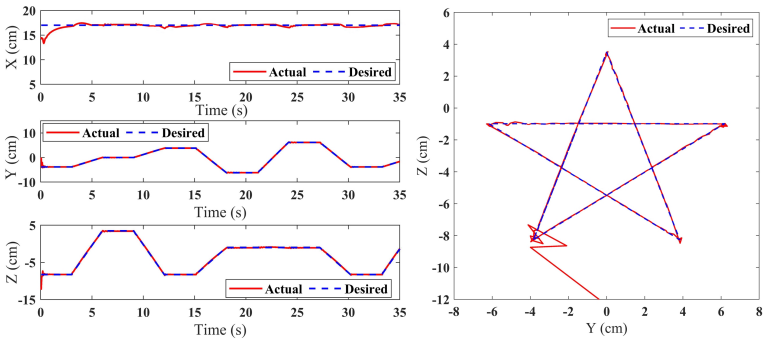


Figure 5.9: Star-shaped trajectory tracking using the proposed controller.

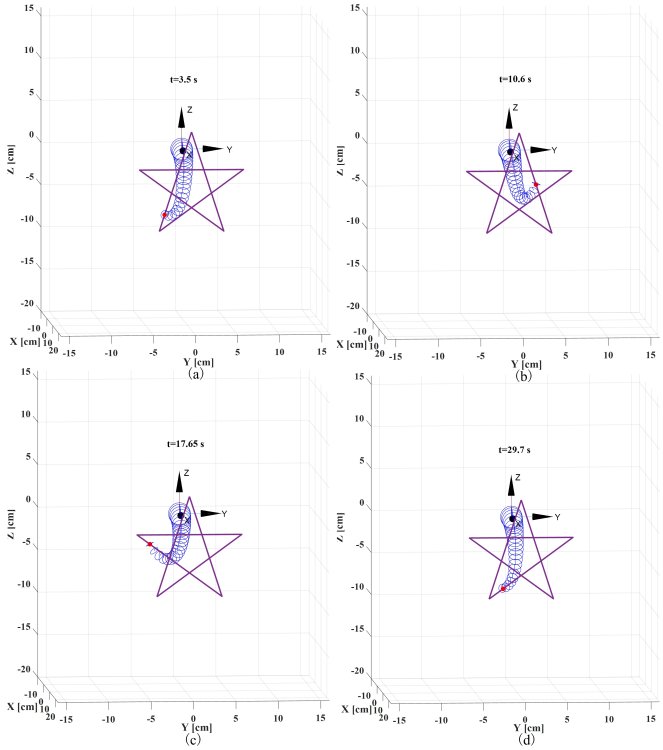


Figure 5.10: Simulation snapshots at time t = 3.5 s, 10.6 s, 17.65 s and 29.7 s, respectively.

In the fourth simulation, we used the proposed controller to control the manipulator to track the time-varying star-shaped trajectory. The sampling time is set to $\mathrm{d}t = 0.05$ s, and the corresponding results are displayed in Fig. 5.9 and Fig. 5.10. It can be clearly seen that the proposed controller enables the manipulator's end-effector to track the star-shaped trajectory with almost no hysteresis, and it allows the soft manipulator converge fast to the desired trajectory with the negligible oscillation in three dimensions.

5.6 Experimental Testing on the Soft Prototype

The trunk-like soft manipulator with the detailed geometric parameters depicted in Fig. 3.8 is selected as the controlled plant, and its material parameters are as follows: Young's modulus $E = 2.563 \times 10^5$ Pa, shear modulus $G = 8.543 \times 10^4$ Pa, and density of material $\rho = 1.41 \times 10^3$ kg/m^3, which have been identified in Section 3.6.

To demonstrate the precision of the differentiator, and the capabilities (i.e., feasibility and robustness) of the proposed controllers, we have conducted several experiments which mainly include the accuracy validation of the differentiator, time-varying trajectory tracking tests, and robustness analysis under the disturbances as well.

5.6.1 Accuracy Validation of Differentiator

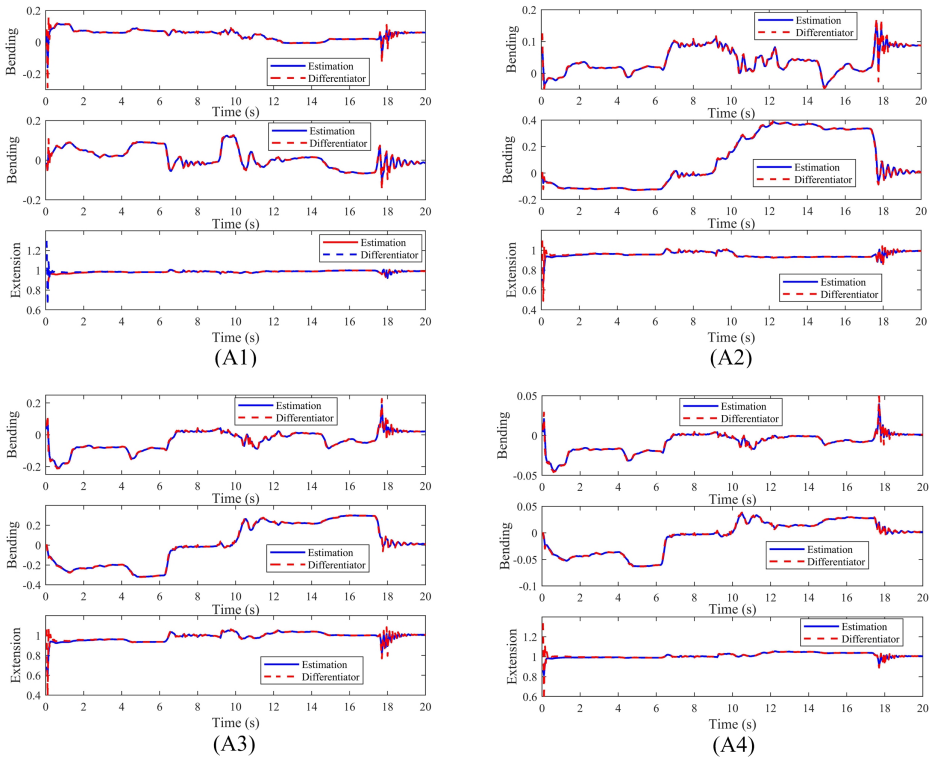


Figure 5.11: Accuracy comparison of each strain component from $\overline{\boldsymbol{\varkappa}}_1$ obtained from the estimation method and $\overline{\boldsymbol{\varkappa}}_1^*$ obtained by the differentiator, respectively.

The generalized strains extracted from the estimation scheme in Section 5.3.4 are compared with those of the differentiator. It is clearly seen from Fig. 5.11 that the estimated strain components of each interpolation node are highly matched with those obtained through the differentiator, showing that the differentiator is capable of rapid and accurate estimation of the system states.

5.6.2 Strain Following Test

To demonstrate the performance of the strain tracking controller in the experiment, we drive the manipulator in Fig. 3.8a to follow two fixed strain configurations. This requires to utilize the optimization algorithm in Section 5.3.4 to generate the reference configuration variables since we only have the measurement of the position.

In the design of soft manipulator, the axial beam stiffness is larger than that of bending direction, and the axial strain variation is small when soft manipulator bends. Thus, we do not intend to control the axial extension or compression strain component for the experiment, indicating that we will only implement curvature control. In this test, the soft prototype was expected to follow the fixed strain components. It can be seen from the left of Fig. 5.12 that, q_a and q_b converge to their desired strain values around 0.7 s and 1.6 s with only very small overshoot for q_a. From the absolute error displayed on the right of Fig. 5.12, it can be concluded that the strain tracking control scheme enables the soft manipulator to quickly converge to the reference strain components with a comparable precision.

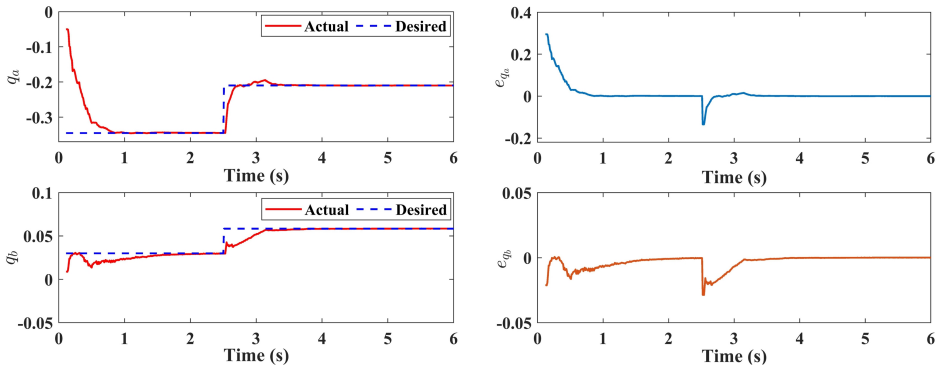


Figure 5.12: Strain trajectory tracking and absolute error between actual and desired trajectory for the studied soft manipulator by using the strain controller.

5.6.3 End-effector Position Tracking Tests

In this experiment, the soft manipulator was expected to track the rapidly star-shaped and circular trajectories using the proposed controller with state variables obtained from 5.12. The results are displayed in Fig. 5.13 and Fig. 5.14, which indicate that the end-effector of the manipulator was capable of fast reacting to the different rapidly time-varying trajectories. To put it another way, the proposed controller is of high feasibility and effectiveness for the manipulator moving around its whole workspace.

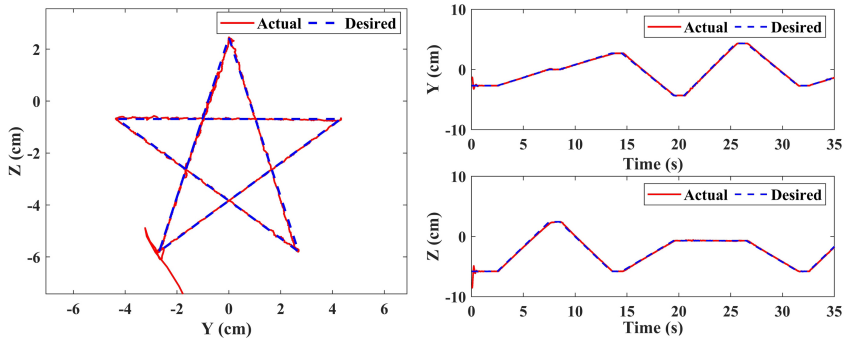


Figure 5.13: Rapidly time-varying star-shaped trajectory tracking by using the end-effector position controller.

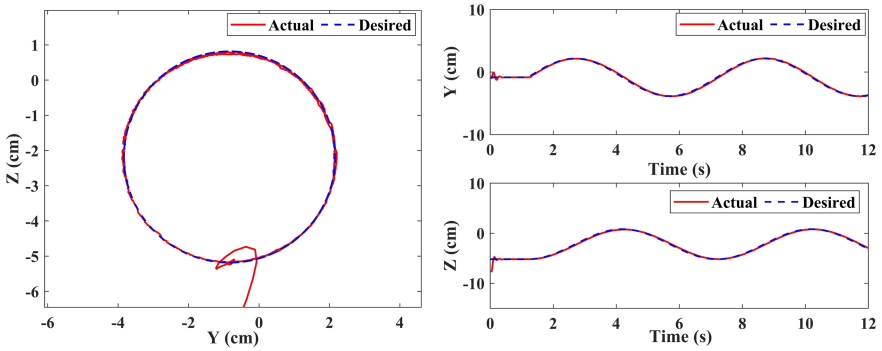


Figure 5.14: Rapidly time-varying circular trajectory tracking by applying the end-effector position controller.

5.6.4 Robustness Analysis

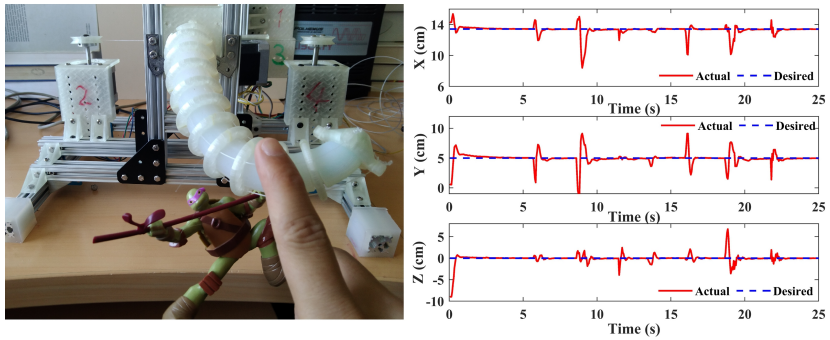


Figure 5.15: The results of soft manipulator for following the fixed point in the presence of temporary and permanent disturbances by using the end-effector position controller.

As we all know, the need for a closed-loop control strategy is more important in the presence of external disturbances. To prove that our proposed PLS Cosserat dynamic model-based controller is robust to external disturbances, we conducted the tracking test under external disturbances.

In the test, the manipulator carries an object (0.032 kg) which can be regarded as an externally permanent disturbance, and is simultaneously exerted a temporary perturbation on it by hand. As for the results from Fig. 5.15, the end-effector of the soft manipulator steadily tracks the trajectory, and this demonstrates that the closed-loop control scheme via the PLS Cosserat dynamics is the high robustness to the externally permanent and temporary disturbances.

5.7 Conclusion

In this chapter, we have presented a general control framework to achieve the control of the soft manipulator via the PLS Cosserat dynamics, and provided the theoretical convergence proof for the established control architecture, based on which the strain and end-effector position controllers were then designed. The simulation results demonstrated the developed controllers can always guarantee finite-time convergence of soft manipulator to a desired reference (strain and position trajectories) in 3D space, and enable to exploit the knowledge of the manipulator's dynamics to realize better and faster tracking performance compared to the static model-based controllers [132].

Despite this success, the designed controllers are dependent on the strain variable and its derivative which are in fact difficult to obtain in practice. Therefore, we introduced the state estimation approach and differentiator to estimate the system states in a finite time, and then presented the estimation-based control scheme. Several experiments were carried out to drive the soft manipulator to track the strain and end-effector position trajectories, and the experimental results shows the controllers can always rapidly track the desired reference (fixed strain, or different position trajectories). This work can be seen as a first step towards developing the strain and end-effector position controllers for the soft manipulator via the PLS Cosserat dynamics, and the proposed estimation-based control framework can be applied to the specific unstructured environments for the soft manipulators to implement the rapid positioning tasks with high accuracy.

Chapter 6

Conclusions and Perspectives

6.1 Conclusions

This thesis contributes to the modeling, strain control and end-effector position control for the slender soft manipulator based on the Cosserat rod theory. The proposed modeling approach can be employed to simulate a wide variety of slender soft robots with complex configurations, and facilitates the complete development of the model-based controllers from the theoretical proofs to the practical experimental validation. In the following, the conclusions for each chapter are given.

In Chapter 2, the systematic way to describe a rigid body's position and orientation which relies on attaching a reference frame to the body was introduced. After that, we reviewed the development of the continuous Cosserat models for the soft manipulator.

In Chapter 3, we have presented the derivation of the PLS Cosserat model, which provides the analytical formulas to simulate the dynamics of deformable objects and capture their motions including bending, elongation, shear and torsion. The accuracy of the PLS Cosserat models are in contrast to that of other discrete modeling methods by several simulations to corroborate the strengths of the proposed technique. Furthermore, through the strain mode choice scheme via PLS Cosserat, we can model the complex configurations with a reduced number of DoFs. The simulation results demonstrate that removing some modes is feasible in the special case, which is beneficial to the real-time simulation and control. Finally, three model parameter identification approaches are proposed. By the simulation and experimental validation, the proposed parameter identification methods for the PLS Cosserat model are of high feasibility and precision.

In Chapter 4, the PLS Cosserat static model-based control techniques have been explored. For the designed local and the global controllers, the rigorous stability proofs are provided by the established Lyapunov functions. The feasibility has been proved through the different tests, and the robustness of the proposed methods has been illustrated by rejecting the temporary and permanent external disturbances. The control schemes always enable the end-effector of the soft manipulator to reach the desired points or time-varying trajectories. However, as stated previously, the proposed controllers have some limitations. The main limitation of the these controllers is the quasi-static assumption, where we assume that the manipulator moves at low velocity. To handle fast motion control of the soft manipulator, it would be necessary to develop a control scheme based on PLS Cosserat dynamic model, which allows to compensate the vibrations of the robot and ensures stable and accurate control during high-speed movements.

In Chapter 5, we have developed two types of controllers to control the strain and

end-effector position of the soft manipulator within the framework of the estimation-based control, respectively. To validate the control performances, we have carried out extensive simulations and experiments, which indicate the proposed control approaches can drive the end-effector of the soft manipulator to track the desired rapid trajectories.

6.2 Perspectives

6.2.1 Recursive Articulated Body Algorithm for PLS Cosserat Dynamics

Forward dynamics is the problem of finding the acceleration of the system in response to given applied forces. There are two main approaches to solving the forward dynamics problem for a kinematic tree [149]:

- form an equation of motion for the whole system, and solve it for the acceleration variables;
- propagate constraints from one body to the next in such a way that the accelerations can be calculated one joint at a time.

In Section 3.3.3, the first approach has been used and involved with a calculation method (i.e., composite body algorithm) of the coefficient matrices of (3.20), whose basic idea is that each section of the soft manipulator contributes to the specific 6×6 element components of the coefficient matrices. This scheme is suitable to controller design although such an algorithm has higher complexity compared to the second one.

The second approach involves making a fixed number of steps, and each step performs a fixed number of calculations. The articulated body algorithm, as the prime example of a propagation algorithms, is theoretically minimal computational complexity for calculating the forward dynamics problem. To improve the efficiency of dynamic simulation, this algorithm can be adopted and divided into following three steps:

1. In the first step, (3.3) and (3.7) are used to recursively compute, respectively, the configuration $\boldsymbol{g}(X)$ and velocity $\boldsymbol{\eta}(X)$ of each cross section X of the soft manipulator from base to tip.
2. In the second step, the articulated body inertia and bias force for all the sections of the soft manipulator should be calculated.
3. In the third step, we can use (3.9) and the second step to recursively compute $\dot{\boldsymbol{\eta}}(X)$ and $(\ddot{\bar{\boldsymbol{\xi}}}_{n-1}, \ddot{\bar{\boldsymbol{\xi}}}_{n})$ from the base of the soft manipulator to the tip.

6.2.2 From Cable-driven Soft Manipulators to Other Actuation Manners

As far as the actuation is concerned, the most common type of actuators used for soft manipulators by far are the tendon and fluidic actuation, with different applications [50,150]. In this thesis, we have focused on the soft manipulator actuated by cables in both the modeling and the controller design.

Since the PCS method was suitable to different methods of actuation (e.g., tendon and fluidic [76]), and the FEM approach was also applied to different actuation methods (e.g., pneumatic

and hydraulic in [151, 152]), we can therefore use different manners of actuation by adapting the actuation wrench $\boldsymbol{\mathcal{F}}_{ia}$ of the PLS Cosserat model in (3.14). For example, if the PLS Cosserat model is actuated by fluidic actuation, the actuation modeling is similar to the cable-driven actuation, which is given by [79]

$$\boldsymbol{\mathcal{F}}_{ia}(X) = \boldsymbol{\Lambda}(X)\boldsymbol{T}$$

where $\boldsymbol{T}$ is the vector consisted of the magnitude of the actuation force given by fluid pressure for fluidic actuation.

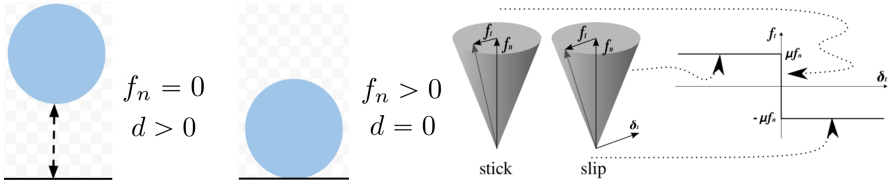
6.2.3 Considering Frictional Contacts between Soft Manipulator and External Environments

One of the focuses of research on soft robotics is the contact between the robot and the environment. The basic idea of contact detection in soft robots is to develop methods or mechanisms that enable the robot to sense and identify when it comes into contact with objects or surfaces in its environment. Even though robots can be modeled in a high accuracy, it is challenging to obtain an accurate model of the contact objects, particularly when the contact objects are changing in the dynamic environments.

The frictional contacts in soft robotics can be mathematically modeled using the concept of the linear complementarity problem (LCP) or the nonlinear complementarity problem (NCP), depending on the specific characteristics of the system being modeled. The LCP is suitable for modeling linear frictional behaviors, while the NCP is more general and can handle nonlinear frictional behaviors. Since the contacts may lead to high deformations of the soft robotic structure, a nonlinear model of the robot is necessary to achieve this task. Therefore, the PLS Cosserat dynamic model with frictional contacts can be written as the following NCP:

$$\begin{gathered}\underline{\boldsymbol{M}}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \underline{\boldsymbol{C}}(\boldsymbol{q},\dot{\boldsymbol{q}})\dot{\boldsymbol{q}} - \underline{\boldsymbol{K}}(\boldsymbol{q},\dot{\boldsymbol{q}}) + \boldsymbol{H}_n(\boldsymbol{q})\boldsymbol{\lambda}_n + \boldsymbol{H}_t(\boldsymbol{q})\boldsymbol{\lambda}_t = \overline{\boldsymbol{H}}(\boldsymbol{q})\boldsymbol{T} + \overline{\boldsymbol{\mathcal{H}}}\dot{\boldsymbol{T}}, \\ \boldsymbol{d} > \boldsymbol{0},\ \boldsymbol{\lambda}_n > \boldsymbol{0},\ \boldsymbol{d}^{\top}\cdot\boldsymbol{\lambda}_n = \boldsymbol{0},\ \mu f_n - f_t \geqslant 0.\end{gathered} \tag{6.1}$$

where $\boldsymbol{H}_n$ and $\boldsymbol{H}_t$ are the generalized contact matrices; $\boldsymbol{\lambda}_n$ and $\boldsymbol{\lambda}_t$ represent the normal and tangential contact forces, respectively; $\boldsymbol{d}$ stands for the vector consisted of all contact points distance (see Fig. 6.1a); μ is the frictional coefficient; the last inequality describes the Coulomb's friction law (see Fig. 6.1b).



(a) Signorini's law for contacts [153]. (b) Coulomb's friction law [154].

Figure 6.1: Schematic illustrations of Signorini's law and Coulomb's friction law.

This dynamic system can be solved by using several numerical methods. For more details, one can refer to [155, 156].

6.2.4 State Estimation of PLS Cosserat Model Using A Single Sensor

In this thesis, we have achieved the estimation of the joint position vector in a finite time by using four position sensors (see Section 4.5). To implement the state estimation with fewer sensors, we can adapt the original scheme to the one that only needs a single sensor placed at the end-effector of the soft manipulator, which can be formulated as the following NLP problem:

$$\begin{aligned} &\underset{(\boldsymbol{q},\ \boldsymbol{T})}{\arg\min}\ \|\boldsymbol{p}(\boldsymbol{q}) - \boldsymbol{p}_{\mathrm{d}}\|_2^2 + \beta(\mathcal{P}_E + \mathcal{P}_G) \\ &s.t. \quad \begin{cases} \text{PLS Cosserat static/dynamic model;} \\ \boldsymbol{T} \in \boldsymbol{\mathcal{T}} \end{cases} \end{aligned} \tag{6.2}$$

Compared to the estimation problem given in subsection 4.5.1, the equality and inequality constraints are added to the adaption scheme while the less sensors are required. To put it another way, the mechanical behavior of the model is taken into account in this scheme, which is of the same form as the problem (4.13), but solves it differently. To obtain the solution of the above NLP problem in real time, the sequential quadratic programming method will be used to achieve this goal.

6.2.5 End-effector Orientation and Position Control of the Soft Manipulator

In this thesis, we have proposed different methodologies only to control the end-effector position of the soft manipulator. However, we can extend it to control the orientation while controlling the position of the end-effector. First, we need to adapt the rotation matrix $\boldsymbol{R}(X)$ to the Euler angles (θ_x, θ_y, θ_z) in three axes for the purpose of investigating the orientation control. Next, we add the equation $\boldsymbol{\theta}(L) = f(\boldsymbol{q})$ to the system (4.2) to achieve the orientation and position control of end-effector for the soft manipulator based on the PLS Cosserat static model, or add this equation to (5.17) for implementing configuration control via the PLS Cosserat-based dynamics.

It is worth noting that the rank of the matrix $\boldsymbol{\Lambda}(X)$ is related to the cables' distribution. For our studied soft manipulator, 4 cables are attached at the end-effector such that the rank of the matrix $\boldsymbol{\Lambda}(X)$ equals to 3. In order to control 6 variables (i.e., 3 angle components and 3 position components), 8 cables are required to attach on the soft arm, and the cables' distribution is illustrated in Fig. 6.2.

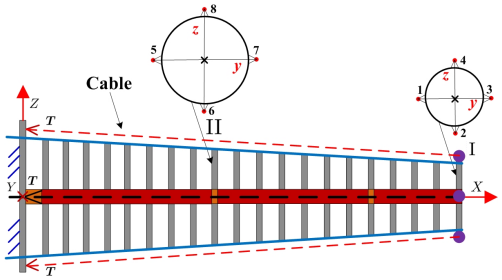


Figure 6.2: The side view of the soft manipulator actuated by 8 cables.

To determine the Jacobian matrix between the Euler angles $\boldsymbol{\theta}$ and the cables' tension $\boldsymbol{T}$, the gradient of $\boldsymbol{\theta}$ with respect to $\boldsymbol{T}$ is then calculated. Finally, a robust controller can be designed for the investigated soft manipulator.

Appendix A

Theoretical Background for Modeling

A.1 Rigid body Motions

To describe the configuration of the rigid body in 3D space, only the position and orientation of the body frame with respect to the fixed frame need to be specified. The configuration of the frame attached to the body with respect to a fixed reference frame can be expressed as a 4×4 matrix, which not only stands for the configuration of a frame, but can also be used to translate and rotate a vector or a frame, and change the representation of a vector or a frame from coordinates in one frame to ones in another frame.

The angular and linear velocities are combined together into a six-dimensional (6D) vector called a twist, while torques and forces are packaged together into a 6D vector called a wrench. The twist, wrench, and Newton-Euler formulation lay the foundation for the kinematic and dynamic analysis of manipulators, and they allow a global description of rigid-body motion without being affected by singularities due to the use of local coordinates. In the following, we will briefly illustrate these concepts.

A.1.1 Rotation Matrices

There are nine entries in the rotation matrix $\boldsymbol{R}$, however, only three can be chosen independently due to the unit norm and orthogonality conditions. The set of 3×3 rotation matrices forms the special orthogonal group $SO(3)$, which can be defined as follows.

Definition 2. *The special orthogonal group $SO(3)$, known as the group of rotation matrices, is the set of all 3×3 real matrices $\boldsymbol{R}$ that satisfy*

$$\boldsymbol{R}^\top \boldsymbol{R} = \mathbf{I}_3$$

and

$$\det \boldsymbol{R} = +1$$

where the additional constraint $\det \boldsymbol{R} = +1$ means that only right-handed frames are allowed.

A rotation matrix $\boldsymbol{R}$ aims at representing an orientation, changing the reference frame, and rotating a vector or a frame.

A.1.2 Angular Velocities

The rotation matrix $\boldsymbol{R}(t)$ describes the orientation of the body frame with respect to the fixed frame at time t. $\boldsymbol{\omega}_A$ and $\boldsymbol{\omega}_B$ are defined as the vector representations of the same angular velocity $\boldsymbol{\omega}$ in fixed frame and moving frame at time t, respectively. Thus, the time evolution of rotation matrix can be obtained

$$\dot{\boldsymbol{R}} = \boldsymbol{\omega}_A \times \boldsymbol{R} = \widetilde{\boldsymbol{\omega}}_A \boldsymbol{R} \tag{A.1}$$

where $\widetilde{\boldsymbol{\omega}}_A$ is a 3×3 skew-symmetric matrix representation of $\boldsymbol{\omega}_A \in \mathbb{R}^3$.

Definition 3. *Given a vector* $\boldsymbol{a} = \begin{bmatrix} a_1 & a_2 & a_3 \end{bmatrix}^\top$*, define*

$$\widetilde{\boldsymbol{a}} = \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}$$

as a 3×3 *skew-symmetric matrix representation of* $\boldsymbol{a}$*; that is,*

$$\widetilde{\boldsymbol{a}} = -\widetilde{\boldsymbol{a}}^\top$$

The set of all 3×3 real skew symmetric matrices is called $so(3)$ which is the Lie algebra of the special orthogonal group $SO(3)$. A useful property involving rotation and skew symmetric matrices is introduced as follows.

Proposition 1. *Given any* $\boldsymbol{\omega} \in \mathbb{R}^3$ *and* $\boldsymbol{R} \in SO(3)$*, the following equality always holds:*

$$\boldsymbol{R}\widetilde{\boldsymbol{\omega}}\boldsymbol{R}^\top = \widetilde{\boldsymbol{R}\boldsymbol{\omega}}$$

Post-multiplying both sides of (A.1) by $\boldsymbol{R}^\top$ to obtain

$$\widetilde{\boldsymbol{\omega}}_A = \dot{\boldsymbol{R}}\boldsymbol{R}^\top$$

To obtain the angular velocity $\boldsymbol{\omega}_B$ expressed in the body frame from $\boldsymbol{\omega}_A$, we have

$$\boldsymbol{\omega}_B = \boldsymbol{R}^\top \boldsymbol{\omega}_A,$$

which can also be expressed in skew-symmetric matrix representation using the Proposition 1 as follows

$$\widetilde{\boldsymbol{\omega}}_B = \boldsymbol{R}^\top \dot{\boldsymbol{R}}$$

Finally, we discovered that pre-or post-multiplying $\dot{\boldsymbol{R}}$ by $\boldsymbol{R}^\top$ leads to a skew-symmetric representation of the angular velocity vector, either in fixed- or body-frame.

A.1.3 Homogeneous Transformation Matrices

A natural choice of representations for the orientation and position of a rigid body is to use a rotation matrix $\boldsymbol{R} \in SO(3)$ to describe the orientation of the body frame $\{B\}$ in the fixed frame $\{A\}$ and a vector $\boldsymbol{p} \in \mathbb{R}^3$ to stand for the origin of the body frame in the fixed frame, as shown in Fig. A.1. They are packaged into a single matrix as follows.

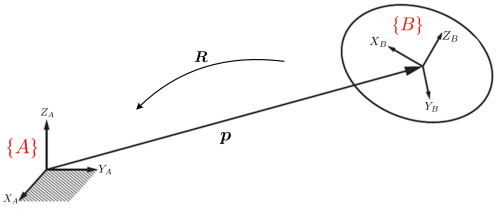


Figure A.1: Description of the configuration of rigid body.

Definition 4. *The special Euclidean group SE(3), regarded as the group of homogeneous transformation matrices, is the set of all* 4×4 *real matrices* $\boldsymbol{T}$ *of the form:*

$$\boldsymbol{T} = \begin{bmatrix} \boldsymbol{R} & \boldsymbol{p} \\ \boldsymbol{0} & 1 \end{bmatrix}$$

where $\boldsymbol{R} \in SO(3)$ *and* $\boldsymbol{p} \in \mathbb{R}^3$ *is a column vector.*

The same representation can also be used for a rigid body transformation between two coordinate frames.

A.1.4 Twists

The infinitesimal version of the screw motion is called a twist, which describes the instantaneous velocity of a rigid body in terms of linear and angular components, and plays an important role in formulation of the kinematics and dynamics of robotic mechanisms. The homogeneous transformation matrix $\boldsymbol{T}$ represents the configuration of body frame as seen from the inertial coordinate frame, as illustrated in Fig. A.1. To obtain the velocity twists in the moving frame and fixed frame, we will calculate $\boldsymbol{T}^{-1}\dot{\boldsymbol{T}}$ and $\dot{\boldsymbol{T}}\boldsymbol{T}^{-1}$, respectively. The calculation of $\boldsymbol{T}^{-1}\dot{\boldsymbol{T}}$ yields

$$\boldsymbol{T}^{-1}\dot{\boldsymbol{T}} = \begin{bmatrix} \boldsymbol{R}^\top & -\boldsymbol{R}^\top\boldsymbol{p} \\ \boldsymbol{0} & 1 \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{R}} & \dot{\boldsymbol{p}} \\ \boldsymbol{0} & 0 \end{bmatrix} = \begin{bmatrix} \boldsymbol{R}^\top\dot{\boldsymbol{R}} & \boldsymbol{R}^\top\dot{\boldsymbol{p}} \\ \boldsymbol{0} & 0 \end{bmatrix} = \begin{bmatrix} \widetilde{\boldsymbol{\omega}}_B & \boldsymbol{v}_B \\ \boldsymbol{0} & 0 \end{bmatrix}$$

where $\boldsymbol{R}^\top\dot{\boldsymbol{R}} = \widetilde{\boldsymbol{\omega}}_B$ is just the skew-symmetric matrix representation of the angular velocity expressed in the body frame, $\dot{\boldsymbol{p}}$ represents the linear velocity of the origin of body frame expressed in the fixed frame, while $\boldsymbol{v}_B$ is the linear velocity expressed in the body frame. The angular and linear velocities are merged into a single 6D vector called body velocity twist, and it is given by

$$\boldsymbol{\eta}_B = \begin{bmatrix} \boldsymbol{\omega}_B \\ \boldsymbol{v}_B \end{bmatrix} \in \mathbb{R}^6$$

Just as a skew-symmetric matrix representation of an angular velocity vector, a matrix representation of a twist can be formulated as

$$\widehat{\boldsymbol{\eta}}_B = \boldsymbol{T}^{-1}\dot{\boldsymbol{T}} = \begin{bmatrix} \widetilde{\boldsymbol{\omega}}_B & \boldsymbol{v}_B \\ \boldsymbol{0} & 0 \end{bmatrix} \in se(3)$$

where the symbol $\widehat{}$ represents the isomorphism between the vector space $\mathbb{R}^6$ and $se(3)$.

Next, the calculation of $\dot{\boldsymbol{T}}\boldsymbol{T}^{-1}$ is given by

$$\dot{\boldsymbol{T}}\boldsymbol{T}^{-1} = \begin{bmatrix} \dot{\boldsymbol{R}} & \dot{\boldsymbol{p}} \\ \boldsymbol{0} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{R}^\top & -\boldsymbol{R}^\top\boldsymbol{p} \\ \boldsymbol{0} & 1 \end{bmatrix} = \begin{bmatrix} \dot{\boldsymbol{R}}\boldsymbol{R}^\top & \dot{\boldsymbol{p}} - \dot{\boldsymbol{R}}\boldsymbol{R}^\top\boldsymbol{p} \\ \boldsymbol{0} & 0 \end{bmatrix} = \begin{bmatrix} \widetilde{\boldsymbol{\omega}}_A & \boldsymbol{v}_A \\ \boldsymbol{0} & 0 \end{bmatrix}$$

where $\dot{\boldsymbol{R}}\boldsymbol{R}^\top = \widetilde{\boldsymbol{\omega}}_A$ represents the angular velocity expressed in the fixed frame, and $\boldsymbol{v}_A$ is the instantaneous velocity of the point on this body expressed in the fixed frame. As we did above, $\boldsymbol{\omega}_A$ and $\boldsymbol{v}_A$ are assembled into a 6D spatial velocity twist, and its matrix representation can be expressed as

$$\widehat{\boldsymbol{\eta}}_A = \dot{\boldsymbol{T}}\boldsymbol{T}^{-1} = \begin{bmatrix} \widetilde{\boldsymbol{\omega}}_A & \boldsymbol{v}_A \\ \boldsymbol{0} & 0 \end{bmatrix} \in se(3)$$

Finally, we will find out the relationship of the velocity twists in the frames $\{A\}$ and $\{B\}$, $\boldsymbol{\eta}_A$ from $\boldsymbol{\eta}_B$ is then given by

$$\boldsymbol{\eta}_A = [\boldsymbol{T}\widehat{\boldsymbol{\eta}}_B\boldsymbol{T}^{-1}]^\vee = \begin{bmatrix} \boldsymbol{R}\widetilde{\boldsymbol{\omega}}_B\boldsymbol{R}^\top & -\boldsymbol{R}\widetilde{\boldsymbol{\omega}}_B\boldsymbol{R}^\top\boldsymbol{p} + \boldsymbol{R}\boldsymbol{v}_B \\ \boldsymbol{0} & 0 \end{bmatrix}^\vee = \underbrace{\begin{bmatrix} \boldsymbol{R} & \boldsymbol{0}_{3\times3} \\ \widetilde{\boldsymbol{p}}\boldsymbol{R} & \boldsymbol{R} \end{bmatrix}}_{\mathrm{Ad}_{\boldsymbol{T}}} \boldsymbol{\eta}_B \tag{A.2}$$

where the symbol $\vee$ is an operator about mapping a matrix into a vector, and Ad represents the Adjoint representation defined in Appendix C.

A.1.5 Exponential Coordinate Representation

Every rigid body motion can be realized by unifying a rotation about a fixed axis and a translation parallel to this axis [157]. Considering the above definition of the matrix representation of the body velocity twist $\boldsymbol{\eta}_B$, we have

$$\dot{\boldsymbol{T}} = \boldsymbol{T}\widehat{\boldsymbol{\eta}}_B$$

which is a differential equation on a Lie group. If $\widehat{\boldsymbol{\eta}}_B$ is independent of time t, the analytical solution of this equation can be formulated as

$$\boldsymbol{T}(t) = \boldsymbol{T}(0)e^{\widehat{\boldsymbol{\eta}}_B t} \tag{A.3}$$

where $\boldsymbol{T}(0)$ is the initial configuration of a rigid body, and $e^{\widehat{\boldsymbol{\eta}}_B t}$ maps an element of the Lie algebra $\widehat{\boldsymbol{\eta}}_B \in se(3)$ into an element of the Lie group $\boldsymbol{T} \in SE(3)$.

According to the Proposition 2.9 in [124], every rigid transformation $\boldsymbol{T}$ can be written as the exponential of some twist. In (A.3), the exponential map can be viewed as a local parameterization that provides solutions to a linear differential equation on a Lie group. The exponential of a twist as a mapping from initial to final configurations is especially important when we investigate the kinematics of robotic mechanisms in the following chapters.

A.1.6 Wrenches

A generalized force acting on a rigid body consists of a linear component (pure force) and an angular component (pure torque). Just as the linear and angular components of velocities merged as twists, we can also merge torques and forces into a single 6D vector called wrench as

$$\boldsymbol{\mathcal{F}} = \begin{bmatrix} \boldsymbol{m} \\ \boldsymbol{f} \end{bmatrix}$$

Recalling that the dot product of a force and a velocity is a power which is a coordinate-independent quantity, therefore, we have

$$\boldsymbol{\eta}_B^\top\boldsymbol{\mathcal{F}}_B = \boldsymbol{\eta}_A^\top\boldsymbol{\mathcal{F}}_A \tag{A.4}$$

where $\boldsymbol{\mathcal{F}}_A$ and $\boldsymbol{\mathcal{F}}_B$ represent the wrenches expressed in the frames $\{A\}$ and $\{B\}$, respectively. Substituting (A.2) into (A.4) yields

$$\boldsymbol{\eta}_B^\top\boldsymbol{\mathcal{F}}_B = (\mathrm{Ad}_{\boldsymbol{T}}\boldsymbol{\eta}_B)^\top\boldsymbol{\mathcal{F}}_A$$

which always holds for all $\boldsymbol{\eta}_B$, and simplifies to

$$\boldsymbol{\mathcal{F}}_A = \mathrm{Ad}_{\boldsymbol{T}}\boldsymbol{\mathcal{F}}_B \tag{A.5}$$

It can be seen from (A.5) that the relation of $\boldsymbol{\mathcal{F}}_A$ and $\boldsymbol{\mathcal{F}}_B$ is given by the Adjoint representation.

A.2 Newton–Euler Formulation

Typically, the dynamic equations for the robots are deduced by using one of two methods: by means of Newton's and Euler's dynamic equations, known as the Newton–Euler formulation for the rigid body, or by the Lagrangian dynamic formulation obtained from the kinetic and potential energy of the robot.

The Lagrangian dynamics is extremely effective for robots with fewer DoFs, however, the calculations will become cumbersome for the robots with more DoFs [158]. On the contrary, the Newton–Euler formulation yields efficient recursive algorithms for the forward or inverse dynamics which can also be assembled into closed-form analytical formulations.

It should be emphasized that the Newton–Euler formulation allows computationally efficient implementation, especially for the robots with more DoFs, without the need for differentiation. The resulting motion equations are the same as those derived by using the energy-based Lagrangian approach.

A.2.1 Dynamics of A Rigid Body

A rigid body can be regarded as composing of a number of rigidly connected particles, where the particle i has mass m_i. The vector $\boldsymbol{r}_i = [x_i, y_i, z_i]$ is the fixed location of i in the body frame, where the origin of this frame, called center of mass, is the point such that $\sum_i m_i \boldsymbol{r}_i = 0$. Assuming that the body is moving with a body twist $\boldsymbol{\eta}_B = (\boldsymbol{w}_B, \boldsymbol{v}_B)$, and the vector $\boldsymbol{p}_i$ is the time-varying position of m_i in the inertial frame, which is initially located at $\boldsymbol{r}_i$, we then have

$$\dot{\boldsymbol{p}}_i = \boldsymbol{v}_B + \boldsymbol{\omega}_B \times \boldsymbol{p}_i$$

$$\ddot{\boldsymbol{p}}_i = \dot{\boldsymbol{v}}_B + \dot{\boldsymbol{\omega}}_B \times \boldsymbol{p}_i + \boldsymbol{\omega}_B \times (\boldsymbol{v}_B + \boldsymbol{\omega}_B \times \boldsymbol{p}_i)$$

Replacing $\boldsymbol{p}_i$ on the right-hand side with $\boldsymbol{r}_i$ and adopting the skew-symmetric notation yield

$$\ddot{\boldsymbol{p}}_i = \dot{\boldsymbol{v}}_B + \widetilde{\dot{\boldsymbol{\omega}}}_B \boldsymbol{r}_i + \widetilde{\boldsymbol{\omega}}_B \boldsymbol{v}_B + \widetilde{\boldsymbol{\omega}}_B^2 \boldsymbol{r}_i$$

According to the statement of the Newton's second law, the force acting on the particle i is given by

$$\boldsymbol{f}_i = m_i(\dot{\boldsymbol{v}}_B + \widetilde{\dot{\boldsymbol{\omega}}}_B \boldsymbol{r}_i + \widetilde{\boldsymbol{\omega}}_B \boldsymbol{v}_B + \widetilde{\boldsymbol{\omega}}_B^2 \boldsymbol{r}_i)$$

and the torque is formulated as

$$\boldsymbol{m}_i = \widetilde{\boldsymbol{r}}_i \boldsymbol{f}_i$$

The total force and torque acting on the rigid body can be expressed as the wrench $\boldsymbol{\mathcal{F}}_B$:

$$\boldsymbol{\mathcal{F}}_B = \begin{bmatrix} \boldsymbol{m}_B \\ \boldsymbol{f}_B \end{bmatrix} = \begin{bmatrix} \sum_i \boldsymbol{m}_i \\ \sum_i \boldsymbol{f}_i \end{bmatrix}$$

For any vector $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^3$, $\widetilde{\boldsymbol{a}} = -\widetilde{\boldsymbol{a}}^\top$, $\widetilde{\boldsymbol{a}}\boldsymbol{b} = -\widetilde{\boldsymbol{b}}\boldsymbol{a}$, and $\widetilde{\boldsymbol{a}}\widetilde{\boldsymbol{b}} = (\widetilde{\boldsymbol{b}}\widetilde{\boldsymbol{a}})^\top$, the expressions for $\boldsymbol{f}_B$ and $\boldsymbol{m}_B$ can be simplified. Recalling that $\sum_i m_i \boldsymbol{r}_i = 0$, thus $\sum_i m_i \widetilde{\boldsymbol{r}}_i = 0$. In terms of the linear dynamics (Newton's equation), we have

$$\boldsymbol{f}_B = \sum_i m_i(\dot{\boldsymbol{v}}_B + \widetilde{\dot{\boldsymbol{\omega}}}_B \boldsymbol{r}_i + \widetilde{\boldsymbol{\omega}}_B \boldsymbol{v}_B + \widetilde{\boldsymbol{\omega}}_B^2 \boldsymbol{r}_i)$$

$$= \sum_i m_i(\dot{\boldsymbol{v}}_B + \widetilde{\boldsymbol{\omega}}_B \boldsymbol{v}_B) \underbrace{- \sum_i m_i \widetilde{\boldsymbol{r}}_i \dot{\boldsymbol{\omega}}_B - \sum_i m_i \widetilde{\boldsymbol{r}}_i \widetilde{\boldsymbol{\omega}}_B \boldsymbol{\omega}_B}_{0} = m(\dot{\boldsymbol{v}}_B + \widetilde{\boldsymbol{\omega}}_B \boldsymbol{v}_B)$$

After that, focusing on the rotational dynamics (Euler's equation), we can obtain

$$\begin{aligned}
\boldsymbol{m}_B &= \sum_i m_i \widetilde{\boldsymbol{r}}_i (\dot{\boldsymbol{v}}_B + \dot{\widetilde{\boldsymbol{\omega}}}_B \boldsymbol{r}_i + \widetilde{\boldsymbol{\omega}}_B \boldsymbol{v}_B + \widetilde{\boldsymbol{\omega}}_B^2 \boldsymbol{r}_i) \\
&= \underbrace{\sum_i m_i \widetilde{\boldsymbol{r}}_i \dot{\boldsymbol{v}}_B + \sum_i m_i \widetilde{\boldsymbol{r}}_i \widetilde{\boldsymbol{\omega}}_B \boldsymbol{v}_B}_{0} + \sum_i m_i \widetilde{\boldsymbol{r}}_i (\dot{\widetilde{\boldsymbol{\omega}}}_B \boldsymbol{r}_i + \widetilde{\boldsymbol{\omega}}_B^2 \boldsymbol{r}_i) \\
&= \sum_i m_i \widetilde{\boldsymbol{r}}_i (\dot{\widetilde{\boldsymbol{\omega}}}_B \boldsymbol{r}_i + \widetilde{\boldsymbol{\omega}}_B^2 \boldsymbol{r}_i) = \sum_i m_i (-\widetilde{\boldsymbol{r}}_i^2 \dot{\boldsymbol{\omega}}_B - \widetilde{\boldsymbol{r}}_i^{\mathrm{T}} \widetilde{\boldsymbol{\omega}}_B^{\mathrm{T}} \widetilde{\boldsymbol{r}}_i \boldsymbol{\omega}_B) \\
&= \sum_i m_i (-\widetilde{\boldsymbol{r}}_i^2 \dot{\boldsymbol{\omega}}_B - \widetilde{\boldsymbol{\omega}}_B \widetilde{\boldsymbol{r}}_i^2 \boldsymbol{\omega}_B) = \left(- \sum_i m_i \widetilde{\boldsymbol{r}}_i^2 \right) \dot{\boldsymbol{\omega}}_B + \widetilde{\boldsymbol{\omega}}_B \left(- \sum_i m_i \widetilde{\boldsymbol{r}}_i^2 \right) \boldsymbol{\omega}_B \\
&= \boldsymbol{\mathcal{I}}_B \dot{\boldsymbol{\omega}}_B + \widetilde{\boldsymbol{\omega}}_B \boldsymbol{\mathcal{I}}_B \boldsymbol{\omega}_B
\end{aligned}$$

where $\boldsymbol{\mathcal{I}}_B = -\sum_i m_i \widetilde{\boldsymbol{r}}_i^2 \in \mathbb{R}^{3\times 3}$ is the rotational inertia matrix of the rigid body, and it is symmetric and positive definite matrix.

A.2.2 Twist-Wrench Formulation of Rigid-body Dynamics

The above linear and the rotational dynamics can be written as the following combined form:

$$\begin{bmatrix} \boldsymbol{m}_B \\ \boldsymbol{f}_B \end{bmatrix} = \begin{bmatrix} \boldsymbol{\mathcal{I}}_B & \mathbf{0} \\ \mathbf{0} & m\mathbf{I} \end{bmatrix} \begin{bmatrix} \dot{\boldsymbol{\omega}}_B \\ \dot{\boldsymbol{v}}_B \end{bmatrix} - \begin{bmatrix} \widetilde{\boldsymbol{\omega}}_B & \mathbf{0} \\ \widetilde{\boldsymbol{v}}_B & \widetilde{\boldsymbol{\omega}}_B \end{bmatrix}^{\top} \begin{bmatrix} \boldsymbol{\mathcal{I}}_B & \mathbf{0} \\ \mathbf{0} & m\mathbf{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{\omega}_B \\ \boldsymbol{v}_B \end{bmatrix}$$

And then, each term of the above formulation can be specified by using the following definition and notation:

$$\boldsymbol{\mathcal{M}}_B = \begin{bmatrix} \boldsymbol{\mathcal{I}}_B & \mathbf{0} \\ \mathbf{0} & m\mathbf{I} \end{bmatrix}, \quad \mathrm{ad}_{\boldsymbol{\eta}_B} \triangleq \begin{bmatrix} \widetilde{\boldsymbol{\omega}}_B & \mathbf{0} \\ \widetilde{\boldsymbol{v}}_B & \widetilde{\boldsymbol{\omega}}_B \end{bmatrix} \in \mathbb{R}^{6\times 6}$$

Therefore, the rigid-body dynamics in the body frame can be written as

$$\boldsymbol{\mathcal{F}}_B = \boldsymbol{\mathcal{M}}_B \dot{\boldsymbol{\eta}}_B - \mathrm{ad}_{\boldsymbol{\eta}_B}^{\top} \boldsymbol{\mathcal{M}}_B \boldsymbol{\eta}_B \tag{A.6}$$

This equation gives a global description of the motion equations for a rigid body subject to the external wrench.

Appendix B

Simplification of Internal Wrench

In the light of the definition of the internal wrench, the detailed derivation of $\boldsymbol{F}_{\mathrm{i}}(\boldsymbol{q},\dot{\boldsymbol{q}})$ can be summarized as follows

$$
\begin{aligned}
&\boldsymbol{F}_{\mathrm{i}}(\boldsymbol{q},\dot{\boldsymbol{q}}) = \boldsymbol{\mathcal{P}}^{\top}\int_{0}^{L_N}\boldsymbol{J}^{\top}\left(\boldsymbol{\mathcal{F}}_i' - \mathrm{ad}_{\boldsymbol{\xi}}^{\top}\boldsymbol{\mathcal{F}}_i\right)\mathrm{d}X\\
&= \boldsymbol{\mathcal{P}}^{\top}\int_{0}^{L_N}\boldsymbol{J}^{\top}\Big[\boldsymbol{\Sigma}'\Big(\boldsymbol{\xi}(X)-\boldsymbol{\xi}_0\Big)+\boldsymbol{\Sigma}\Big(\boldsymbol{\xi}(X)-\boldsymbol{\xi}_0\Big)'+\boldsymbol{\gamma}\dot{\boldsymbol{\xi}}'(X)+\boldsymbol{\gamma}'\dot{\boldsymbol{\xi}}(X)-\mathrm{ad}_{\boldsymbol{\xi}}^{\top}\Big(\boldsymbol{\Sigma}(\boldsymbol{\xi}(X)-\boldsymbol{\xi}_0)+\boldsymbol{\gamma}\dot{\boldsymbol{\xi}}(X)\Big)\Big]\mathrm{d}X\\
&= \boldsymbol{\mathcal{P}}^{\top}\int_{0}^{L_N}\boldsymbol{J}^{\top}\Big[\Big(\boldsymbol{\Sigma}'-\mathrm{ad}_{\boldsymbol{\xi}}^{\top}\boldsymbol{\Sigma}\Big)\Big(\boldsymbol{\xi}(X)-\boldsymbol{\xi}_0\Big)+\boldsymbol{\Sigma}\Big(\boldsymbol{\xi}(X)-\boldsymbol{\xi}_0\Big)'+\Big(\boldsymbol{\gamma}'-\mathrm{ad}_{\boldsymbol{\xi}}^{\top}\boldsymbol{\gamma}\Big)\dot{\boldsymbol{\xi}}(X)+\boldsymbol{\gamma}\dot{\boldsymbol{\xi}}'(X)\Big]\mathrm{d}X\\
&= \boldsymbol{\mathcal{P}}^{\top}\sum_{n=1}^{N}\int_{L_{n-1}}^{L_n}\boldsymbol{J}^{\top}\Big[\Big(\boldsymbol{\Sigma}'-\mathrm{ad}_{\boldsymbol{\xi}_n}^{\top}\boldsymbol{\Sigma}\Big)\Big((\overline{\boldsymbol{\xi}}_{n-1}-\boldsymbol{\xi}_{(n-1)0})a_n(X)+(\overline{\boldsymbol{\xi}}_n-\boldsymbol{\xi}_{n0})b_n(X)\Big)\\
&\quad+\boldsymbol{\Sigma}(\overline{\boldsymbol{\xi}}_{n-1}-\boldsymbol{\xi}_{(n-1)0})a_n(X)'+(\overline{\boldsymbol{\xi}}_n-\boldsymbol{\xi}_{n0})b_n(X)'\Big]\mathrm{d}X\\
&\quad+\boldsymbol{\mathcal{P}}^{\top}\sum_{n=1}^{N}\int_{L_{n-1}}^{L_n}\boldsymbol{J}^{\top}\Big[\Big(\boldsymbol{\gamma}'-\mathrm{ad}_{\boldsymbol{\xi}_n}^{\top}\boldsymbol{\gamma}\Big)\Big(\dot{\overline{\boldsymbol{\xi}}}_{n-1}a_n(X)+\dot{\overline{\boldsymbol{\xi}}}_n b_n(X)\Big)+\boldsymbol{\gamma}\Big(\dot{\overline{\boldsymbol{\xi}}}_{n-1}a_n(X)'+\dot{\overline{\boldsymbol{\xi}}}_n b_n(X)'\Big)\Big]\mathrm{d}X\\
&= \boldsymbol{\mathcal{P}}^{\top}\sum_{n=1}^{N}\int_{L_{n-1}}^{L_n}\boldsymbol{J}^{\top}\Bigg\{\Big(\boldsymbol{\Sigma}'-\mathrm{ad}_{\boldsymbol{\xi}_n}^{\top}\boldsymbol{\Sigma}\Big)\Big[a_n(X)\mathbf{I}_6\quad b_n(X)\mathbf{I}_6\Big]\mathrm{d}X\begin{bmatrix}\overline{\boldsymbol{\xi}}_{n-1}-\boldsymbol{\xi}_{(n-1)0}\\ \overline{\boldsymbol{\xi}}_n-\boldsymbol{\xi}_{n0}\end{bmatrix}\\
&\quad+\boldsymbol{\Sigma}\Big[a_n(X)'\mathbf{I}_6\quad b_n(X)'\mathbf{I}_6\Big]\mathrm{d}X\begin{bmatrix}\overline{\boldsymbol{\xi}}_{n-1}-\boldsymbol{\xi}_{(n-1)0}\\ \overline{\boldsymbol{\xi}}_n-\boldsymbol{\xi}_{n0}\end{bmatrix}\Bigg\}\\
&\quad+\boldsymbol{\mathcal{P}}^{\top}\sum_{n=1}^{N}\int_{L_{n-1}}^{L_n}\boldsymbol{J}^{\top}\Bigg\{\Big(\boldsymbol{\gamma}'-\mathrm{ad}_{\boldsymbol{\xi}_n}^{\top}\boldsymbol{\gamma}\Big)\Big[a_n(X)\mathbf{I}_6\quad b_n(X)\mathbf{I}_6\Big]\mathrm{d}X\begin{bmatrix}\dot{\overline{\boldsymbol{\xi}}}_{n-1}\\ \dot{\overline{\boldsymbol{\xi}}}_n\end{bmatrix}+\boldsymbol{\gamma}\Big[a_n(X)'\mathbf{I}_6\quad b_n(X)'\mathbf{I}_6\Big]\mathrm{d}X\begin{bmatrix}\dot{\overline{\boldsymbol{\xi}}}_{n-1}\\ \dot{\overline{\boldsymbol{\xi}}}_n\end{bmatrix}\Bigg\}\\
&= \boldsymbol{\mathcal{P}}^{\top}\sum_{n=1}^{N}\int_{L_{n-1}}^{L_n}\Big(\boldsymbol{A}_n+\boldsymbol{B}_n\Big)\mathrm{d}X\begin{bmatrix}\overline{\boldsymbol{\xi}}_{n-1}-\boldsymbol{\xi}_{(n-1)0}\\ \overline{\boldsymbol{\xi}}_n-\boldsymbol{\xi}_{n0}\end{bmatrix}+\boldsymbol{\mathcal{P}}^{\top}\sum_{n=1}^{N}\int_{L_{n-1}}^{L_n}\Big(\boldsymbol{C}_n+\boldsymbol{D}_n\Big)\mathrm{d}X\begin{bmatrix}\dot{\overline{\boldsymbol{\xi}}}_{n-1}\\ \dot{\overline{\boldsymbol{\xi}}}_n\end{bmatrix}\\
&= \boldsymbol{\mathcal{P}}^{\top}\Big[\int_{L_0}^{L_1}\Big(\boldsymbol{A}_1+\boldsymbol{B}_1\Big)\mathrm{d}X\quad\int_{L_1}^{L_2}\Big(\boldsymbol{A}_2+\boldsymbol{B}_2\Big)\mathrm{d}X\quad\cdots\quad\int_{L_{N-1}}^{L_N}\Big(\boldsymbol{A}_N+\boldsymbol{B}_N\Big)\mathrm{d}X\Big]\begin{bmatrix}\overline{\boldsymbol{\xi}}_0-\boldsymbol{\xi}_{00}\\ \overline{\boldsymbol{\xi}}_1-\boldsymbol{\xi}_{10}\\ \overline{\boldsymbol{\xi}}_1-\boldsymbol{\xi}_{10}\\ \overline{\boldsymbol{\xi}}_2-\boldsymbol{\xi}_{20}\\ \vdots\\ \overline{\boldsymbol{\xi}}_{N-1}-\boldsymbol{\xi}_{(N-1)0}\\ \overline{\boldsymbol{\xi}}_N-\boldsymbol{\xi}_{N0}\end{bmatrix}
\end{aligned}
$$

$$
\begin{aligned}
&+\boldsymbol{\mathcal{P}}^{\top}\Big[\int_{L_0}^{L_1}\Big(\boldsymbol{C}_1+\boldsymbol{D}_1\Big)\mathrm{d}X \quad \int_{L_1}^{L_2}\Big(\boldsymbol{C}_2+\boldsymbol{D}_2\Big)\mathrm{d}X \quad \cdots \quad \int_{L_{N-1}}^{L_N}\Big(\boldsymbol{C}_N+\boldsymbol{D}_N\Big)\mathrm{d}X\Big]
\begin{bmatrix}
\dot{\bar{\boldsymbol{\xi}}}_0\\ \dot{\bar{\boldsymbol{\xi}}}_1\\ \dot{\bar{\boldsymbol{\xi}}}_1\\ \dot{\bar{\boldsymbol{\xi}}}_2\\ \vdots\\ \dot{\bar{\boldsymbol{\xi}}}_{N-1}\\ \dot{\bar{\boldsymbol{\xi}}}_N
\end{bmatrix}\\
&=\boldsymbol{\mathcal{P}}^{\top}\Big[\int_{L_0}^{L_1}\Big(\boldsymbol{A}_1+\boldsymbol{B}_1\Big)\mathrm{d}X \quad \int_{L_1}^{L_2}\Big(\boldsymbol{A}_2+\boldsymbol{B}_2\Big)\mathrm{d}X \quad \cdots \quad \int_{L_{N-1}}^{L_N}\Big(\boldsymbol{A}_N+\boldsymbol{B}_N\Big)\mathrm{d}X\Big]\boldsymbol{\mathcal{I}}(\boldsymbol{q}-\boldsymbol{q}_0)\\
&\quad+\boldsymbol{\mathcal{P}}^{\top}\Big[\int_{L_0}^{L_1}\Big(\boldsymbol{C}_1+\boldsymbol{D}_1\Big)\mathrm{d}X \quad \int_{L_1}^{L_2}\Big(\boldsymbol{C}_2+\boldsymbol{D}_2\Big)\mathrm{d}X \quad \cdots \quad \int_{L_{N-1}}^{L_N}\Big(\boldsymbol{C}_N+\boldsymbol{D}_N\Big)\mathrm{d}X\Big]\boldsymbol{\mathcal{I}}\dot{\boldsymbol{q}}\\
&=\boldsymbol{K}(\boldsymbol{q})\Big(\boldsymbol{q}-\boldsymbol{q}_0\Big)+\boldsymbol{D}(\boldsymbol{q})\dot{\boldsymbol{q}}
\end{aligned}
$$

with

$$
\boldsymbol{\mathcal{I}}=\begin{bmatrix}
\mathbf{I}_6 & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0}\\
\mathbf{0} & \mathbf{I}_6 & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0}\\
\mathbf{0} & \mathbf{I}_6 & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{0}\\
\mathbf{0} & \mathbf{0}_6 & \mathbf{I}_6 & \cdots & \mathbf{0} & \mathbf{0}\\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots\\
\mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{I}_6 & \mathbf{0}\\
\mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{I}_6 & \mathbf{0}\\
\mathbf{0} & \mathbf{0} & \mathbf{0} & \cdots & \mathbf{0} & \mathbf{I}_6
\end{bmatrix}\in\mathbb{R}^{12N\times 6(N+1)}
$$

$$
\begin{aligned}
&\boldsymbol{A}_n=\boldsymbol{J}^{\top}(\boldsymbol{\Sigma}'-\mathrm{ad}_{\boldsymbol{\xi}_n}^{\top}\boldsymbol{\Sigma})\begin{bmatrix}a_n(X)\mathbf{I}_6 & b_n(X)\mathbf{I}_6\end{bmatrix},\boldsymbol{B}_n=\boldsymbol{J}^{\top}\boldsymbol{\Sigma}\begin{bmatrix}a_n(X)'\mathbf{I}_6 & b_n(X)'\mathbf{I}_6\end{bmatrix};\\
&\boldsymbol{C}_n=\boldsymbol{J}^{\top}(\boldsymbol{\gamma}'-\mathrm{ad}_{\boldsymbol{\xi}_n}^{\top}\boldsymbol{\gamma})\begin{bmatrix}a_n(X)\mathbf{I}_6 & b_n(X)\mathbf{I}_6\end{bmatrix},\boldsymbol{D}_n=\boldsymbol{J}^{\top}\boldsymbol{\gamma}\begin{bmatrix}a_n(X)'\mathbf{I}_6 & b_n(X)'\mathbf{I}_6\end{bmatrix}.
\end{aligned}
$$

where $\boldsymbol{\mathcal{I}}$ represents transformation matrix for reducing the dimension of the strain twists.

Appendix C

Lie Group Framework

Definition 5. *A group G is a set of elements g with a composition operation which satisfies the group axioms involving associativity, the neutral element, the inverse element, and closure. A Lie group is a continuous group for which the composition rule and the inverse are smooth.*

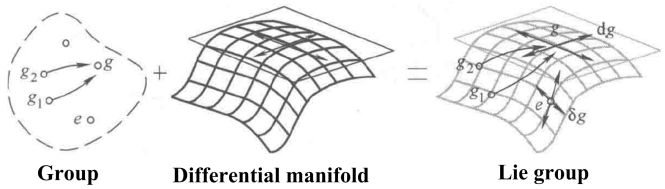


Figure C.1: The relationship among group, differential manifold and Lie group.

Therefore, mathematically speaking, a Lie group is a smooth and differentiable manifold equipped with a group structure such that the operations of group multiplication and inversion are smooth. Not every Lie group is isomorphic to a matrix Lie group. In this context, however, we have restricted our attention to matrix Lie groups for the purpose of minimizing prerequisites and keeping the discussion as concrete as possible.

Definition 6. *The Lie algebra $\mathfrak{g}$ is the tangent space at the identity of a Lie group. A finite-dimensional real or complex Lie algebra is a finite dimensional real or complex vector space $\mathfrak{g}$, together with a map $[\cdot,\cdot]$ from $\mathfrak{g}\times\mathfrak{g}$ into $\mathfrak{g}$ with the following properties:*

1. *$[\cdot,\cdot]$ is bilinear, which is referred to as the bracket operation on $\mathfrak{g}$.*
2. *$[\cdot,\cdot]$ is skew symmetric: $[X,Y]=-[Y,X]$ for all $X,Y\in\mathfrak{g}$.*

$\mathfrak{g}$ is a Lie algebra with bracket operation given by

$$[X,Y]=XY-YX$$

Definition 7. *If $\mathfrak{g}$ is a Lie algebra and X is an element of $\mathfrak{g}$, define a linear map $\mathrm{ad}_X:\mathfrak{g}\to\mathfrak{g}$ by*

$$\mathrm{ad}_X Y=[X,Y]$$

The map $X \mapsto \mathrm{ad}_X$ is the adjoint map or adjoint representation.

The adjoint representation (or adjoint action) of a Lie group G is a way of representing the elements of the group as linear transformations of the group's Lie algebra, considered as a vector space. Let $\psi : G \to \mathrm{Aut}(G)$ be the mapping $g \mapsto \psi_g$, with $\mathrm{Aut}(G)$ the automorphism group of G and $\psi_g : G \to G$ given by the inner automorphism (conjugation) $\psi_g(h) = ghg^{-1}$. This ψ is a Lie group homomorphism.

Since $g \mapsto \psi_g$ is a Lie group homomorphism, $g \mapsto \mathrm{Ad}_g$ is also a group homomorphism. Hence, the map $\mathrm{Ad} : G \to \mathrm{Aut}(\mathfrak{g}), g \mapsto \mathrm{Ad}_g$ is a group representation called the adjoint representation of G. Some properties of the adjoint representation is given by

$$(\mathrm{Ad}_g)^{-1} = \mathrm{Ad}_{g^{-1}}$$

$$\mathrm{Ad}_{g_1 g_2} = \mathrm{Ad}_{g_1} \mathrm{Ad}_{g_2}$$

It is a typical feature of Lie theory of continuous groups to associate Lie algebra and Lie groups by employing the matrix exponential map.

Bibliography

[1] Andrea Cherubini, Robin Passama, André Crosnier, Antoine Lasnier, and Philippe Fraisse. Collaborative manufacturing with physical human–robot interaction. *Robotics and Computer-Integrated Manufacturing*, 40:1–13, 2016.

[2] Jessica Burgner-Kahrs, D Caleb Rucker, and Howie Choset. Continuum robots for medical applications: A survey. *IEEE Transactions on Robotics*, 31(6):1261–1280, 2015.

[3] Deepak Trivedi, Christopher D Rahn, William M Kier, and Ian D Walker. Soft robotics: Biological inspiration, state of the art, and future research. *Applied bionics and biomechanics*, 5(3):99–117, 2008.

[4] Thomas George Thuruthel, Yasmin Ansari, Egidio Falotico, and Cecilia Laschi. Control strategies for soft robotic manipulators: A survey. *Soft robotics*, 5(2):149–163, 2018.

[5] Daniela Rus and Michael T Tolley. Design, fabrication and control of soft robots. *Nature*, 521(7553):467–475, 2015.

[6] Riby Abraham Boby. Identification of elasto-static parameters of an industrial robot using monocular camera. *Robotics and Computer-Integrated Manufacturing*, 74:102276, 2022.

[7] Filippo Sanfilippo, Lars Ivar Hatledal, Houxiang Zhang, Massimiliano Fago, and Kristin Y Pettersen. Controlling kuka industrial robots: Flexible communication interface jopenshowvar. *IEEE robotics & automation magazine*, 22(4):96–109, 2015.

[8] Lei Tang, Jungang Wang, Yang Zheng, Guoying Gu, Limin Zhu, and Xiangyang Zhu. Design of a cable-driven hyper-redundant robot with experimental validation. *International Journal of Advanced Robotic Systems*, 14(5):1729881417734458, 2017.

[9] Robert J Webster III and Bryan A Jones. Design and kinematic modeling of constant curvature continuum robots: A review. *The International Journal of Robotics Research*, 29(13):1661–1683, 2010.

[10] Michele Giorelli, Federico Renda, Marcello Calisti, Andrea Arienti, Gabriele Ferri, and Cecilia Laschi. Neural network and jacobian method for solving the inverse statics of a cable-driven soft arm with nonconstant curvature. *IEEE Transactions on Robotics*, 31(4):823–834, 2015.

[11] Yong Zhong, Luohua Hu, and Yinsheng Xu. Recent advances in design and actuation of continuum robots for medical applications. In *Actuators*, volume 9, page 142. Multidisciplinary Digital Publishing Institute, 2020.

[12] Amehri Walid, Gang Zheng, Alexandre Kruszewski, and Federico Renda. Discrete cosserat method for soft manipulators workspace estimation: An optimization-based approach. *Journal of Mechanisms and Robotics*, 14(1), 2022.

[13] Carmel Majidi. Soft robotics: a perspective—current trends and prospects for the future. *Soft robotics*, 1(1):5–11, 2014.

[14] Olesya Ogorodnikova. Methodology of safety for a human robot interaction designing stage. In *2008 Conference on Human System Interactions*, pages 452–457. IEEE, 2008.

[15] Thor Morales Bieze. *Contribution to the kinematic modeling and control of soft manipulators using computational mechanics*. PhD thesis, Lille 1, 2017.

[16] Gang Zheng. Control of a silicone soft tripod robot via uncertainty compensation. *IEEE Robotics and Automation Letters*, 5(2):2801–2807, 2020.

[17] Apurba Das and M Nabi. A review on soft robotics: modeling, control and applications in human-robot interaction. In *2019 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, pages 306–311. IEEE, 2019.

[18] Zhongkai Zhang. *Vision-based calibration, position control and force sensing for soft robots*. PhD thesis, Université de Lille, 2019.

[19] Andrea Arienti, Marcello Calisti, Francesco Giorgio-Serchi, and Cecilia Laschi. Poseidrone: design of a soft-bodied rov with crawling, swimming and manipulation ability. In *2013 OCEANS-San Diego*, pages 1–7. IEEE, 2013.

[20] Faheem Ahmed, Muhammad Waqas, Bushra Shaikh, Umair Khan, Afaque Manzoor Soomro, Suresh Kumar, Hina Ashraf, Fida Hussain Memon, and Kyung Hyun Choi. Multi-material bio-inspired soft octopus robot for underwater synchronous swimming. *Journal of Bionic Engineering*, pages 1–13, 2022.

[21] Yuwang Liu, Zhuang Ge, Shangkui Yang, Ian D Walker, and Zhaojie Ju. Elephant's trunk robot: An extremely versatile under-actuated continuum robot driven by a single motor. *Journal of Mechanisms and Robotics*, 11(5):051008, 2019.

[22] T Umedachi, V Vikas, and BA Trimmer. Softworms: the design and control of non-pneumatic, 3d-printed, deformable robots. *Bioinspiration & biomimetics*, 11(2):025001, 2016.

[23] Raphael Deimel and Oliver Brock. A novel type of compliant and underactuated robotic hand for dexterous grasping. *The International Journal of Robotics Research*, 35(1-3):161–185, 2016.

[24] Laura Yu Matloff. *Birds of a feather: Designing feathered morphing wings for soft aerial robots*. Stanford University, 2020.

[25] Robert K Katzschmann, Joseph DelPreto, Robert MacCurdy, and Daniela Rus. Exploration of underwater life with an acoustically controlled soft robotic fish. *Science Robotics*, 3(16):eaar3449, 2018.

[26] Alireza Ramezani, Soon-Jo Chung, and Seth Hutchinson. A biomimetic robotic platform to study flight specializations of bats. *Science Robotics*, 2(3):eaal2505, 2017.

[27] Shirong Zheng, Tongil Park, Manh Cuong Hoang, Gwangjun Go, Chang-sei Kim, Jong-Oh Park, Eunpyo Choi, and Ayoung Hong. Ascidian-inspired soft robots that can crawl, tumble, and pick-and-place objects. *IEEE Robotics and Automation Letters*, 6(2):1722–1728, 2021.

[28] Saeed Hashemi, Darrin Bentivegna, and William Durfee. Bone-inspired bending soft robot. *Soft Robotics*, 8(4):387–396, 2021.

[29] Jianglong Guo, Khaled Elgeneidy, Chaoqun Xiang, Niels Lohse, Laura Justham, and Jonathan Rossiter. Soft pneumatic grippers embedded with stretchable electroadhesion. *Smart Materials and Structures*, 27(5):055006, 2018.

[30] Tianhao Zhang, Runyu Tian, Hongqi Yang, Chen Wang, Jinan Sun, Shikun Zhang, and Guangming Xie. From simulation to reality: A learning framework for fish-like robots to perform control tasks. *IEEE Transactions on Robotics*, 2022.

[31] Michele Giorelli, Federico Renda, Gabriele Ferri, and Cecilia Laschi. A feed-forward neural network learning the inverse kinetics of a soft cable-driven manipulator moving in three-dimensional space. In *2013 IEEE/RSJ international conference on intelligent robots and systems*, pages 5033–5039. IEEE, 2013.

[32] Gang Zheng, Yuan Zhou, and Mingda Ju. Robust control of a silicone soft robot using neural networks. *ISA transactions*, 100:38–45, 2020.

[33] James M Bern, Yannick Schnider, Pol Banzet, Nitish Kumar, and Stelian Coros. Soft robot control with a learned differentiable model. In *2020 3rd IEEE International Conference on Soft Robotics (RoboSoft)*, pages 417–423. IEEE, 2020.

[34] Michael W Hannan and Ian D Walker. Analysis and experiments with an elephant's trunk robot. *Advanced Robotics*, 15(8):847–858, 2001.

[35] Wenhui Zeng, Junyan Yan, Kim Yan, Xu Huang, Xuefeng Wang, and Shing Shin Cheng. Modeling a symmetrically-notched continuum neurosurgical robot with non-constant curvature and superelastic property. *IEEE Robotics and Automation Letters*, 6(4):6489–6496, 2021.

[36] Srinivas Neppalli, Matthew A Csencsits, Bryan A Jones, and Ian D Walker. Closed-form inverse kinematics for continuum manipulators. *Advanced Robotics*, 23(15):2077–2091, 2009.

[37] Nabil Simaan, Kai Xu, Wei Wei, Ankur Kapoor, Peter Kazanzides, Russell Taylor, and Paul Flint. Design and integration of a telerobotic system for minimally invasive surgery of the throat. *The International journal of robotics research*, 28(9):1134–1153, 2009.

[38] Bryan A Jones and Ian D Walker. Kinematics for multisection continuum robots. *IEEE Transactions on Robotics*, 22(1):43–55, 2006.

[39] Federico Renda, Costanza Armanini, Anup Mathew, and Frederic Boyer. Geometrically-exact inverse kinematic control of soft manipulators with general threadlike actuators' routing. *IEEE Robotics and Automation Letters*, 7(3):7311–7318, 2022.

[40] Stanislao Grazioso, Giuseppe Di Gironimo, and Bruno Siciliano. A geometrically exact model for soft continuum robots: The finite element deformation space formulation. *Soft robotics*, 6(6):790–811, 2019.

[41] Gregory S Chirikjian and Joel W Burdick. A modal approach to hyper-redundant manipulator kinematics. *IEEE Transactions on Robotics and Automation*, 10(3):343–354, 1994.

[42] Gregory S Chirikjian. Hyper-redundant manipulator dynamics: A continuum approximation. *Advanced Robotics*, 9(3):217–243, 1994.

[43] Frédéric Boyer, Mathieu Porez, and Wisama Khalil. Macro-continuous computed torque algorithm for a three-dimensional eel-like robot. *IEEE transactions on robotics*, 22(4):763–775, 2006.

[44] Christian Duriez. Control of elastic soft robots based on real-time finite element method. In *2013 IEEE international conference on robotics and automation*, pages 3982–3987. IEEE, 2013.

[45] Gina Olson, Ross L Hatton, Julie A Adams, and Yiğit Mengüç. An euler–bernoulli beam model for soft robot arms bent through self-stress and external loads. *International Journal of Solids and Structures*, 207:113–131, 2020.

[46] Lukas Lindenroth, Junghwan Back, Adrian Schoisengeier, Yohan Noh, Helge Würdemann, Kaspar Althoefer, and Hongbin Liu. Stiffness-based modelling of a hydraulically-actuated soft robotics manipulator. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2458–2463. IEEE, 2016.

[47] Federico Renda, Matteo Cianchetti, Michele Giorelli, Andrea Arienti, and Cecilia Laschi. A 3d steady-state model of a tendon-driven continuum soft manipulator inspired by the octopus arm. *Bioinspiration & biomimetics*, 7(2):025006, 2012.

[48] Mattia Gazzola, LH Dudte, AG McCormick, and L Mahadevan. Forward and inverse problems in the mechanics of soft filaments. *Royal Society open science*, 5(6):171628, 2018.

[49] Xiaotian Zhang, Fan Kiat Chan, Tejaswin Parthasarathy, and Mattia Gazzola. Modeling and simulation of complex dynamic musculoskeletal architectures. *Nature communications*, 10(1):1–12, 2019.

[50] Federico Renda, Michele Giorelli, Marcello Calisti, Matteo Cianchetti, and Cecilia Laschi. Dynamic model of a multibending soft robot arm driven by cables. *IEEE Transactions on Robotics*, 30(5):1109–1122, 2014.

[51] John Daniel Till. On the statics, dynamics, and stability of continuum robots: Model formulations and efficient computational schemes. 2019.

[52] John Till, Vincent Aloi, and Caleb Rucker. Real-time dynamics of soft and continuum robots based on cosserat rod models. *The International Journal of Robotics Research*, 38(6):723–746, 2019.

[53] Federico Renda, Frédéric Boyer, Jorge Dias, and Lakmal Seneviratne. Discrete cosserat approach for multisection soft manipulator dynamics. *IEEE Transactions on Robotics*, 34(6):1518–1533, 2018.

[54] Frédéric Boyer, Vincent Lebastard, Fabien Candelier, and Federico Renda. Dynamics of continuum and soft robots: A strain parameterization based approach. *IEEE Transactions on Robotics*, 37(3):847–863, 2020.

[55] Gundula Runge, Mats Wiese, and Annika Raatz. Fem-based training of artificial neural networks for modular soft robots. In *2017 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 385–392, 2017.

[56] René Felix Reinhart, Zeeshan Shareef, and Jochen Jakob Steil. Hybrid analytical and data-driven modeling for feed-forward robot control. *Sensors*, 17(2):311, 2017.

[57] Kristin M De Payrebrune and Oliver M O'Reilly. On the development of rod-based models for pneumatically actuated soft robot arms: a five-parameter constitutive relation. *International Journal of Solids and Structures*, 120:226–235, 2017.

[58] Zhongkai Zhang, Jeremie Dequidt, Alexandre Kruszewski, Frederick Largilliere, and Christian Duriez. Kinematic modeling and observer based control of soft robot using real-time finite element method. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5509–5514. IEEE, 2016.

[59] Zhongkai Zhang, Jérémie Dequidt, and Christian Duriez. Vision-based sensing of external forces acting on soft robots using finite element method. *IEEE Robotics and Automation Letters*, 3(3):1529–1536, 2018.

[60] Maxime Thieffry, Alexandre Kruszewski, Olivier Goury, Thierry-Marie Guerra, and Christian Duriez. Dynamic control of soft robots. In *IFAC World congress*, 2017.

[61] Zu-Qing Qu. *Model Order Reduction Techniques with Applications in Finite Element Analysis: With Applications in Finite Element Analysis*. Springer Science & Business Media, 2004.

[62] Olivier Goury and Christian Duriez. Fast, generic, and reliable control and simulation of soft robots using model order reduction. *IEEE Transactions on Robotics*, 34(6):1565–1576, 2018.

[63] Robert K Katzschmann, Maxime Thieffry, Olivier Goury, Alexandre Kruszewski, Thierry-Marie Guerra, Christian Duriez, and Daniela Rus. Dynamically closed-loop controlled soft robotic arm using a reduced order finite element model with state observer. In *2019 2nd IEEE International Conference on Soft Robotics (RoboSoft)*, pages 717–724. IEEE, 2019.

[64] SMH Sadati, S Elnaz Naghibi, Lyndon Da Cruz, and Christos Bergeles. Reduced-order modeling and model order reduction for soft robots, 2021.

[65] Cosimo Della Santina, Christian Duriez, and Daniela Rus. Model based control of soft robots: A survey of the state of the art and open challenges. *arXiv preprint arXiv:2110.01358*, 2021.

[66] Eulalie Coevoet, Thor Morales-Bieze, Frederick Largilliere, Zhongkai Zhang, Maxime Thieffry, Mario Sanz-Lopez, Bruno Carrez, Damien Marchal, Olivier Goury, Jeremie Dequidt, et al. Software toolkit for modeling, simulation, and control of soft robots. *Advanced Robotics*, 31(22):1208–1224, 2017.

[67] G Runge, M Wiese, L Günther, and A Raatz. A framework for the kinematic modeling of soft material robots combining finite element analysis and piecewise constant curvature kinematics. In *2017 3rd International Conference on Control, Automation and Robotics (ICCAR)*, pages 7–14. IEEE, 2017.

[68] Isuru S Godage, Gustavo A Medrano-Cerda, David T Branson, Emanuele Guglielmino, and Darwin G Caldwell. Dynamics for variable length multisection continuum arms. *The International Journal of Robotics Research*, 35(6):695–722, 2016.

[69] Hesheng Wang, Bohan Yang, Yuting Liu, Weidong Chen, Xinwu Liang, and Rolf Pfeifer. Visual servoing of soft robot manipulator in constrained environments with an adaptive controller. *IEEE/ASME Transactions on Mechatronics*, 22(1):41–50, 2016.

[70] Cosimo Della Santina, Antonio Bicchi, and Daniela Rus. On an improved state parametrization for soft robots with piecewise constant curvature and its use in model based control. *IEEE Robotics and Automation Letters*, 5(2):1001–1008, 2020.

[71] Valentin Falkenhahn, Alexander Hildebrandt, Rüdiger Neumann, and Oliver Sawodny. Model-based feedforward position control of constant curvature continuum robots using feedback linearization. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 762–767. IEEE, 2015.

[72] Cosimo Della Santina, Robert K Katzschmann, Antonio Biechi, and Daniela Rus. Dynamic control of soft robots interacting with the environment. In *2018 IEEE International Conference on Soft Robotics (RoboSoft)*, pages 46–53. IEEE, 2018.

[73] Robert K Katzschmann, Cosimo Della Santina, Yasunori Toshimitsu, Antonio Bicchi, and Daniela Rus. Dynamic motion control of multi-segment soft robots using piecewise constant curvature matched with an augmented rigid body model. In *2019 2nd IEEE International Conference on Soft Robotics (RoboSoft)*, pages 454–461. IEEE, 2019.

[74] Brandon Caasenbrood, Alexander Pogromsky, and Henk Nijmeijer. Control-oriented models for hyperelastic soft robots through differential geometry of curves. *Soft Robotics*, 2022.

[75] Federico Renda, Vito Cacucciolo, Jorge Dias, and Lakmal Seneviratne. Discrete cosserat approach for soft robot dynamics: A new piece-wise constant strain model with torsion and shears. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5495–5502. IEEE, 2016.

[76] Federico Renda, Matteo Cianchetti, Haider Abidi, Jorge Dias, and Lakmal Seneviratne. Screw-based modeling of soft manipulators with tendon and fluidic actuation. *Journal of Mechanisms and Robotics*, 9(4), 2017.

[77] Irfan Hussain, Monica Malvezzi, Dongming Gan, Zubair Iqbal, Lakmal Seneviratne, Domenico Prattichizzo, and Federico Renda. Compliant gripper design, prototyping, and modeling using screw theory formulation. *The International Journal of Robotics Research*, 40(1):55–71, 2021.

[78] Costanza Armanini, Irfan Hussain, Muhammad Zubair Iqbal, Dongming Gan, Domenico Prattichizzo, and Federico Renda. Discrete cosserat approach for closed-chain soft robots: Application to the fin-ray finger. *IEEE Transactions on Robotics*, 2021.

[79] Federico Renda, Costanza Armanini, Vincent Lebastard, Fabien Candelier, and Frederic Boyer. A geometric variable-strain approach for static modeling of soft manipulators with tendon and fluidic actuation. *IEEE Robotics and Automation Letters*, 5(3):4006–4013, 2020.

[80] Jun Wu, Jinsong Wang, and Zheng You. An overview of dynamic parameter identification of robots. *Robotics and computer-integrated manufacturing*, 26(5):414–419, 2010.

[81] Jovana Jovic, Adrien Escande, Ko Ayusawa, Eiichi Yoshida, Abderrahmane Kheddar, and Gentiane Venture. Humanoid and human inertia parameter identification using hierarchical optimization. *IEEE Transactions on Robotics*, 32(3):726–735, 2016.

[82] Yinyan Zhang, Siyuan Chen, Shuai Li, and Zhijun Zhang. Adaptive projection neural network for kinematic control of redundant manipulators with unknown physical parameters. *IEEE Transactions on Industrial Electronics*, 65(6):4909–4920, 2017.

[83] Dechao Chen, Yunong Zhang, and Shuai Li. Tracking control of robot manipulators with unknown models: A jacobian-matrix-adaption method. *IEEE Transactions on Industrial Informatics*, 14(7):3044–3053, 2017.

[84] Naveen Kumar Uppalapati, Gaurav Singh, and Girish Krishnan. Parameter estimation and modeling of a pneumatic continuum manipulator with asymmetric building blocks. In *2018 IEEE International Conference on Soft Robotics (RoboSoft)*, pages 528–533. IEEE, 2018.

[85] Michele Di Lecce, Onaizah Onaizah, Peter Lloyd, James H Chandler, and Pietro Valdastri. Evolutionary inverse material identification: Bespoke characterization of soft materials using a metaheuristic algorithm. *Frontiers in Robotics and AI*, 8, 2021.

[86] Yong Han, Jianhua Wu, Chao Liu, and Zhenhua Xiong. Static model analysis and identification for serial articulated manipulators. *Robotics and Computer-Integrated Manufacturing*, 57:155–165, 2019.

[87] Matthias Rolf and Jochen J Steil. Efficient exploratory learning of inverse kinematics on a bionic elephant trunk. *IEEE transactions on neural networks and learning systems*, 25(6):1147–1160, Jun. 2014.

[88] Thomas Thuruthel, Egidio Falotico, Matteo Cianchetti, Federico Renda, and Cecilia Laschi. Learning global inverse statics solution for a redundant soft robot. In *Proceedings of the 13th International Conference on Informatics in Control, Automation and Robotics*, volume 2, pages 303–310, 2016.

[89] Thomas George Thuruthel, Egidio Falotico, Mariangela Manti, Andrea Pratesi, Matteo Cianchetti, and Cecilia Laschi. Learning closed loop kinematic controllers for continuum manipulators in unstructured environments. *Soft robotics*, 4(3):285–296, 2017.

[90] Minhan Li, Rongjie Kang, David T Branson, and Jian S Dai. Model-free control for continuum robots based on an adaptive kalman filter. *IEEE/ASME Transactions on Mechatronics*, 23(1):286–297, 2017.

[91] Guoxin Fang, Yingjun Tian, Zhi-Xin Yang, Jo MP Geraedts, and Charlie CL Wang. Efficient jacobian-based inverse kinematics with sim-to-real transfer of soft robots by learning. *IEEE/ASME Transactions on Mechatronics*, 2022.

[92] Xuanke You, Yixiao Zhang, Xiaotong Chen, Xinghua Liu, Zhanchi Wang, Hao Jiang, and Xiaoping Chen. Model-free control for soft manipulators based on reinforcement learning. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2909–2915. IEEE, 2017.

[93] Guanda Li, Jun Shintake, and Mitsuhiro Hayashibe. Deep reinforcement learning framework for underwater locomotion of soft robot. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 12033–12039, 2021.

[94] Michael C Yip and David B Camarillo. Model-less feedback control of continuum manipulators in constrained environments. *IEEE Transactions on Robotics*, 30(4):880–889, 2014.

[95] Michael C Yip and David B Camarillo. Model-less hybrid position/force control: a minimalist approach for continuum manipulators in unknown, constrained environments. *IEEE Robotics and Automation Letters*, 1(2):844–851, 2016.

[96] Yu-Yang Wu and Ning Tan. Model-less feedback control for soft manipulators with jacobian adaptation. In *2020 International Symposium on Autonomous Systems (ISAS)*, pages 217–222, 2020.

[97] Thomas George Thuruthel, Egidio Falotico, Federico Renda, and Cecilia Laschi. Learning dynamic models for open loop predictive control of soft robotic manipulators. *Bioinspiration & biomimetics*, 12(6):066003, 2017.

[98] Thomas George Thuruthel, Egidio Falotico, Federico Renda, and Cecilia Laschi. Model-based reinforcement learning for closed-loop dynamic control of soft robotic manipulators. *IEEE Transactions on Robotics*, 35(1):124–134, 2018.

[99] Morgan T Gillespie, Charles M Best, Eric C Townsend, David Wingate, and Marc D Killpack. Learning nonlinear dynamic models of soft robots for model predictive control with neural networks. In *2018 IEEE International Conference on Soft Robotics (RoboSoft)*, pages 39–45, 2018.

[100] Andrew D Marchese, Russ Tedrake, and Daniela Rus. Dynamics and trajectory optimization for a soft spatial fluidic elastomer manipulator. *The International Journal of Robotics Research*, 35(8):1000–1019, 2016.

[101] Andrea Centurelli, Alessandro Rizzo, Silvia Tolu, and Egidio Falotico. Open-loop model-free dynamic control of a soft manipulator for tracking tasks. In *2021 20th International Conference on Advanced Robotics (ICAR)*, pages 128–133. IEEE, 2021.

[102] Hesheng Wang, Weidong Chen, Xiaojin Yu, Tao Deng, Xiaozhou Wang, and Rolf Pfeifer. Visual servo control of cable-driven soft robotic manipulator. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 57–62. IEEE, 2013.

[103] Andrew D Marchese and Daniela Rus. Design, kinematics, and control of a soft spatial fluidic elastomer manipulator. *The International Journal of Robotics Research*, 35(7):840–869, 2016.

[104] Apoorva D Kapadia, Katelyn E Fry, and Ian D Walker. Empirical investigation of closed-loop control of extensible continuum manipulators. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 329–335, 2014.

[105] Cosimo Della Santina, Robert K Katzschmann, Antonio Bicchi, and Daniela Rus. Model-based dynamic feedback control of a planar soft robot: trajectory tracking and interaction with the environment. *The International Journal of Robotics Research*, 39(4):490–513, 2020.

[106] Frederick Largilliere, Valerian Verona, Eulalie Coevoet, Mario Sanz-Lopez, Jeremie Dequidt, and Christian Duriez. Real-time control of soft-robots using asynchronous finite element modeling. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2550–2555. IEEE, 2015.

[107] Maxime Thieffry, Alexandre Kruszewski, Christian Duriez, and Thierry-Marie Guerra. Control design for soft robots based on reduced-order model. *IEEE Robotics and Automation Letters*, 4(1):25–32, 2018.

[108] Maxime Thieffry, Alexandre Kruszewski, Thierry-Marie Guerra, and Christian Duriez. Lpv framework for non-linear dynamic control of soft robots using finite element model. *IFAC-PapersOnLine*, 53(2):7312–7318, 2020.

[109] Sander Tonkens, Joseph Lorenzetti, and Marco Pavone. Soft robot optimal control via reduced order finite element models. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 12010–12016. IEEE, 2021.

[110] Ahmad Abu Alqumsan, Suiyang Khoo, and Michael Norton. Robust control of continuum robots using cosserat rod theory. *Mechanism and Machine Theory*, 131:48–61, 2019.

[111] Mohsen Mahvash and Pierre E Dupont. Stiffness control of surgical continuum manipulators. *IEEE Transactions on Robotics*, 27(2):334–345, 2011.

[112] John Till, Caroline E Bryson, Scotty Chung, Andrew Orekhov, and D Caleb Rucker. Efficient computation of multiple coupled cosserat rod models for real-time simulation and control of parallel continuum manipulators. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5067–5074, 2015.

[113] Federico Campisano, Simone Caló, Andria A Remirez, James H Chandler, Keith L Obstein, Robert J Webster III, and Pietro Valdastri. Closed-loop control of soft continuum manipulators under tip follower actuation. *The International Journal of Robotics Research*, 40(6-7):923–938, 2021.

[114] Thomas George Thuruthel, Federico Renda, and Fumiya Iida. First-order dynamic modeling and control of soft robots. *Frontiers in Robotics and AI*, 7:95, 2020.

[115] Curtis C Johnson, Tyler Quackenbush, Taylor Sorensen, David Wingate, and Marc D Killpack. Using first principles for deep learning and model-based control of soft robots. *Frontiers in Robotics and AI*, 8:654398, 2021.

[116] Mats Wiese, Gundula Runge-Borchert, Benjamin-Hieu Cao, and Annika Raatz. Transfer learning for accurate modeling and control of soft actuators. In *Proc. IEEE Int. Conf. Soft Robot. (RoboSoft)*, pages 51–57, 2021.

[117] Zhi Qiang Tang, Ho Lam Heung, Kai Yu Tong, and Zheng Li. Model-based online learning and adaptive control for a "human-wearable soft robot" integrated system. *The International Journal of Robotics Research*, 40(1):256–276, 2021.

[118] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. 1960.

[119] Hannah Michalska and David Q Mayne. Moving horizon observers and observer-based control. *IEEE Transactions on Automatic Control*, 40(6):995–1006, 1995.

[120] Ross Hartley, Maani Ghaffari, Ryan M Eustice, and Jessy W Grizzle. Contact-aided invariant extended kalman filtering for robot state estimation. *The International Journal of Robotics Research*, 39(4):402–430, 2020.

[121] Stephen Tully, George Kantor, Marco A Zenati, and Howie Choset. Shape estimation for image-guided surgery with a highly articulated snake robot. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1353–1358, 2011.

[122] Ahmad Ataka, Peng Qi, Ali Shiva, Ali Shafti, Helge Wurdemann, Hongbin Liu, and Kaspar Althoefer. Real-time pose estimation and obstacle avoidance for multi-segment continuum manipulator in dynamic environments. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2827–2832, 2016.

[123] Stefan Escaida Navarro, Steven Nagels, Hosam Alagi, Lisa-Marie Faller, Olivier Goury, Thor Morales-Bieze, Hubert Zangl, Björn Hein, Raf Ramakers, Wim Deferme, et al. A model-based sensor fusion approach for force and shape estimation in soft robotics. *IEEE Robotics and Automation Letters*, 5(4):5621–5628, 2020.

[124] Richard M Murray, Zexiang Li, and S Shankar Sastry. *A mathematical introduction to robotic manipulation*. CRC press, 2017.

[125] Deng-Qing Cao and Robin W Tucker. Nonlinear dynamics of elastic rods using the cosserat theory: Modelling and simulation. *International Journal of Solids and Structures*, 45(2):460–477, 2008.

[126] Juan C Simo. A finite strain beam formulation. the three-dimensional dynamic problem. part i. *Computer methods in applied mechanics and engineering*, 49(1):55–70, 1985.

[127] F Renda, M Giorelli, M Calisti, M Cianchetti, and C Laschi. Dynamic model of a multibending soft robot arm driven by cables. *IEEE Transactions on Robotics*, PP(5):1–14, 2014.

[128] Stuart S Antman. The theory of rods. In *Linear theories of elasticity and thermoelasticity*, pages 641–703. Springer, 1973.

[129] Joachim Linn, Holger Lang, and Andrey Tuganov. Geometrically exact cosserat rods with kelvin–voigt type viscous damping. *Mechanical Sciences*, 4(1):79–96, 2013.

[130] Haihong Li, Lingxiao Xun, and Gang Zheng. Piecewise linear strain cosserat model for soft slender manipulator. *IEEE Transactions on Robotics*, 39(3):2342–2359, 2023.

[131] Brian C Hall et al. *Lie groups, Lie algebras, and representations: an elementary introduction*, volume 10. Springer, 2003.

[132] Haihong Li, Lingxiao Xun, Gang Zheng, and Federico Renda. Discrete cosserat static model-based control of soft manipulator. *IEEE Robotics and Automation Letters*, 8(3):1739–1746, 2023.

[133] Ke Wu and Gang Zheng. Fem-based nonlinear controller for a soft trunk robot. *IEEE Robotics and Automation Letters*, 7(2):5735–5740, 2022.

[134] Amehri Walid, Gang Zheng, Alexandre Kruszewski, and Federico Renda. Discrete cosserat method for soft manipulators workspace estimation: An optimization-based approach. *Journal of Mechanisms and Robotics*, 14(1):011012, 2021.

[135] AS El-Bakry, Richard A Tapia, T Tsuchiya, and Yin Zhang. On the formulation and theory of the newton interior-point method for nonlinear programming. *Journal of Optimization theory and Applications*, 89(3):507–541, 1996.

[136] Robert Hecht-Nielsen. Theory of the backpropagation neural network. In *Neural networks for perception*, pages 65–93. 1992.

[137] Ke Wu and Gang Zheng. Fem-based gain-scheduling control of a soft trunk robot. *IEEE Robotics and Automation Letters*, 6(2):3081–3088, 2021.

[138] Moritz Bächer, Espen Knoop, and Christian Schumacher. Design and control of soft robots using differentiable simulation. *Current Robotics Reports*, 2(2):211–221, 2021.

[139] Kunal Garg and Dimitra Panagou. Fixed-time stable gradient flows: Applications to continuous-time optimization. *IEEE Transactions on Automatic Control*, 66(5):2002–2015, 2020.

[140] Orlando Romero and Mouhacine Benosman. Finite-time convergence in continuous-time optimization. In *International Conference on Machine Learning*, pages 8200–8209, 2020.

[141] Jie Huang and Zhiyong Chen. A general framework for tackling the output regulation problem. *IEEE Transactions on Automatic Control*, 49(12):2203–2218, 2004.

[142] Alberto Isidori. *Nonlinear control systems*. Springer, 1995.

[143] Thomas Kailath. *Linear systems*, volume 156. Prentice-Hall Englewood Cliffs, NJ, 1980.

[144] John C Strikwerda. *Finite difference schemes and partial differential equations*. SIAM, 2004.

[145] Hassan K Khalil. High-gain observers in nonlinear feedback control. In *2008 International conference on control, automation and systems*, pages xlvii–lvii, 2008.

[146] Yi Xiong and Mehrdad Saif. Sliding mode observer for nonlinear uncertain systems. *IEEE transactions on automatic control*, 46(12):2012–2017, 2001.

[147] Stefan Koch, Markus Reichhartinger, Martin Horn, and Leonid Fridman. Discrete-time implementation of homogeneous differentiators. *IEEE Transactions on Automatic Control*, 65(2):757–762, 2019.

[148] Arie Levant. Higher-order sliding modes, differentiation and output-feedback control. *International journal of Control*, 76(9-10):924–941, 2003.

[149] Roy Featherstone. *Rigid body dynamics algorithms*. Springer, 2014.

[150] Nikolaos Vasios, Andrew J Gross, Scott Soifer, Johannes TB Overvelde, and Katia Bertoldi. Harnessing viscous flow to simplify the actuation of fluidic soft robots. *Soft robotics*, 7(1):1–9, 2020.

[151] Alejandro Rodríguez, Eulalie Coevoet, and Christian Duriez. Real-time simulation of hydraulic components for interactive control of soft robots. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4953–4958. IEEE, 2017.

[152] Thor Morales Bieze, Frederick Largilliere, Alexandre Kruszewski, Zhongkai Zhang, Rochdi Merzouki, and Christian Duriez. Finite element method-based kinematics and closed-loop control of soft, continuum manipulators. *Soft robotics*, 5(3):348–364, 2018.

[153] Maxime Thieffry. *Model-Based Dynamic Control of Soft Robots*. PhD thesis, Université Polytechnique des Hauts-de-France, 2019.

[154] Christophe Guébert, Christian Duriez, and Laurent Grisoni. Unified processing of constraints for interactive simulation. In *Workshop in Virtual Reality Interactions and Physical Simulation VRIPHYS'2008*. Eurographics association, 2008.

[155] Vincent Acary and Bernard Brogliato. *Numerical methods for nonsmooth dynamical systems: applications in mechanics and electronics*. Springer Science & Business Media, 2008.

[156] Peter Wriggers and Tod A Laursen. *Computational contact mechanics*, volume 2. Springer, 2006.

[157] S Grazioso. *Geometric soft robotics: a finite element approach*. PhD thesis, Ph. D. thesis, University of Naples Federico II, 2018.

[158] C Frank. *Modern Robotics-Mechanics, Planning, and Control*. Cambridge University Press, 2017.

Résumé

La robotique souple est devenue un domaine de recherche émergent en raison de ses caractéristiques distinctes par rapport à ses homologues rigides conventionnels. Fabriqués à partir de matériaux flexibles et conformes, les robots souples présentent de nombreux avantages, notamment une grande dextérité, des interactions sûres et une grande adaptabilité. Ces propriétés inhérentes sont utiles pour les applications robotiques actuelles, notamment pour saisir des objets fragiles, explorer des environnements, etc. Cependant, les robots souples sont caractérisés par un grand nombre de degrés de liberté (DDL) et une déformation fortement non linéaire, ce qui rend leur modélisation et leur contrôle précis difficiles.

Le développement d'outils de modélisation généraux et précis pour les applications de robotique souple est une tâche difficile que de nombreux chercheurs ont tenté de résoudre. Malgré le grand nombre de solutions proposées, de nombreux objectifs souhaités restent à atteindre, notamment en ce qui concerne les applications des méthodes proposées dans la prospective de conception de commande. Cela est principalement dû à la cinématique et à la dynamique complexes et fortement non linéaires associées à ce type de robots mous.

Cette thèse contribue à la modélisation, au contrôle de la déformation et au contrôle de la position de l'effecteur terminal pour le manipulateur souple élancé basé sur la théorie de la tige de Cosserat. L'approche de modélisation proposée peut être utilisée pour simuler une grande variété de robots souple élancé avec des configurations complexes, et facilite le développement complet des contrôleurs basés sur des modèles, des preuves théoriques à la validation expérimentale pratique. Dans ce qui suit, le contenu de chaque chapitre est résumé.

Le chapitre 2 commence par la manière systématique de décrire la position et l'orientation d'un corps rigide qui repose sur l'attachement d'un cadre de référence au corps, suivi par la dérivation standard des équations du mouvement d'un corps rigide à l'aide de la formulation dynamique de Newton-Euler. Des modèles continus de Cosserat comprenant la cinématique, la cinématique différentielle et la statique pour le manipulateur souple sont établis et enfin, le modèle dynamique continu de Cosserat est déduit, ce qui ouvre la voie à l'approche de modélisation discrète.

Pour contribuer à ce domaine de recherche, le chapitre 3 développe une technique de modélisation discrète appelée «déformation linéaire par morceaux» (PLS) pour résoudre les équations aux dérivées partielles (EDP) des modèles basés sur Cosserat pour les manipulateurs souples élancés, sur la base desquels les modèles analytiques associés sont déduits. Pour valider l'exactitude du modèle Cosserat proposé, le modèle statique de la tige en porte-à-faux conique sous gravité en tant qu'exemple simple est simulé en utilisant différentes méthodes de discrétisation. Les résultats indiquent que le modèle Cosserat PLS est comparable au comporte-

ment de déformation mécanique du manipulateur souple réel dans le monde réel. Enfin, trois types de schémas d'identification de paramètre sont établis, et la validation expérimentale ainsi que la simulation démontrent que l'utilisation de ces approches peut identifier les paramètres physiques du modèle avec une grande précision.

Au chapitre 4, nous concevons d'abord un contrôleur localement robuste via le modèle statique PLS Cosserat, qui n'est valable que dans un certain sous-espace de travail en raison d'une matrice jacobienne constante utilisée. Pour mieux contrôler le manipulateur dans tout son espace de travail, nous combinons le réseau neuronal de la fonction de base radiale et PLS Cosserat pour approximer globalement (hors ligne) la matrice jacobienne. Les preuves théoriques et les résultats expérimentaux sur le manipulateur souple étudié démontrent que les contrôleurs proposés peuvent toujours conduire l'effecteur à la position souhaitée, et ont une grande robustesse aux perturbations externes. Enfin, la déformation généralisée peut être obtenue (en ligne) en utilisant la méthode du gradient en temps fini, et les expériences ont été exécutées pour valider la précision de l'estimation de la déformation. Grâce au vecteur de déformation estimé, nous pouvons obtenir la matrice jacobienne en temps réel pour réaliser le contrôle global de la position de l'effecteur terminal.

Cependant, les contrôleurs basés sur un modèle statique présentent également certaines limitations. La principale limitation de ces contrôleurs est l'hypothèse quasi-statique, où nous supposons que le manipulateur se déplace à faible vitesse. Pour gérer le contrôle des mouvements rapides du manipulateur souple, il serait nécessaire de développer un schéma de contrôle basé sur le modèle dynamique PLS Cosserat, qui permette de compenser les vibrations du robot et assure un contrôle stable et précis lors des mouvements à grande vitesse.

Dans le chapitre 5, nous avons présenté un cadre de contrôle général permettant de contrôler le manipulateur souple via la dynamique PLS Cosserat, et nous avons fourni une preuve de convergence théorique pour l'architecture de contrôle établie. Sur cette base, nous avons ensuite conçu des contrôleurs de déformation et de position de l'effecteur terminal. Les résultats de simulation ont démontré que les contrôleurs développés garantissent toujours la convergence du manipulateur souple vers une référence souhaitée (trajectoires de déformation et de position) dans l'espace 3D, et permettent d'exploiter la connaissance de la dynamique du manipulateur pour obtenir des performances de suivi meilleures et plus rapides par rapport aux contrôleurs basés sur un modèle statique.

Malgré ce succès, les contrôleurs conçus dépendent de la variable de déformation et de sa dérivée, qui sont en réalité difficiles à obtenir en pratique. Par conséquent, nous avons introduit l'approche d'estimation d'état et le différenciateur pour estimer les états du système en temps fini, puis nous avons présenté le schéma de contrôle basé sur l'estimation. Plusieurs expériences ont été réalisées pour guider le manipulateur souple afin de suivre les trajectoires de déformation et de position, et les résultats expérimentaux montrent que les contrôleurs peuvent toujours suivre rapidement la référence souhaitée (déformation fixe ou trajectoires de position différentes). Ce travail peut être considéré comme une première étape vers le développement des contrôleurs de déformation et de position de l'effecteur terminal pour le manipulateur souple via la dynamique PLS Cosserat, et le cadre de contrôle basé sur l'estimation proposé peut être appliqué aux environnements non structurés spécifiques des manipulateurs souples pour réaliser des tâches de positionnement rapide avec une grande précision.

Le chapitre 6 présente les conclusions de la présente thèse et expose les perspectives des

travaux futurs, notamment le mode d'actionnement, l'estimation des paramètres de l'offline à l'online, les contacts de frottement entre le manipulateur souple et les environnements externes, ainsi que le contrôle de l'orientation et de la position du manipulateur souple via PLS Cosserat. Les details sont les suivants.

En ce qui concerne l'actionnement, les types d'actionneurs les plus couramment utilisés pour les manipulateurs souples sont de loin l'actionnement tendineux et fluidique, avec différentes applications. Dans cette thèse, nous nous sommes concentrés sur le manipulateur souple actionné par des câbles à la fois dans la modélisation et la conception du contrôleur. Étant donné que la méthode PCS était adaptée à différentes méthodes d'actionnement (par exemple, tendon et fluidique), et que l'approche FEM a également été appliquée à différentes méthodes d'actionnement (par exemple, pneumatique et hydraulique), nous pouvons donc utiliser différentes manières d'actionnement en adaptant la force d'actionnement et couple du modèle PLS Cosserat.

Dans cette thèse, nous avons réalisé l'estimation du vecteur position articulaire en temps fini en utilisant quatre capteurs de position (voir Section 4.5). Pour implémenter l'estimation d'état avec moins de capteurs, nous pouvons adapter le schéma original à celui qui ne nécessite qu'un seul capteur placé à l'effecteur du manipulateur souple. Autrement dit, le comportement mécanique du modèle sera pris en compte dans ce schéma, qui est de la même forme que le problème (4.13), mais le résout différemment.

L'un des points clés de la robotique souple concerne le contact entre le robot et l'environnement, qui peut être modélisé mathématiquement à l'aide du concept de problème de complémentarité linéaire ou non linéaire. Étant donné que les contacts peuvent entraîner de fortes déformations des robots souples, le modèle dynamique PLS Cosserat avec des contacts de frottement peut être formulé comme un problème de complémentarité non linéaire, puis nous pouvons utiliser des méthodes numériques pour le résoudre.

Dans cette thèse, nous avons proposé de contrôler la position du manipulateur souple. À l'avenir, nous pourrons l'étendre pour contrôler à la fois l'orientation et la position de l'effecteur terminal. Tout d'abord, nous devons adapter la matrice de rotation $\boldsymbol{R}$ aux angles d'Euler, puis ajouter cette équation au système statique ou dynamique pour atteindre l'objectif de contrôle.

Modélisation et Commande de Robots Souples Élancés Basées sur la Théorie de Cosserat

Résumé en français

Les robots mous sont caractérisés par un grand nombre de degrés de liberté (DDL) et une déformation fortement non linéaire, ce qui rend difficile leur modélisation et leur contrôle précis. Par conséquent, des défis scientifiques tels que le développement d'un cadre théorique exhaustif pour la modélisation et le contrôle dynamique se sont posés dans le domaine de la robotique souple. Dans cette thèse, nous nous concentrons principalement sur les robots souples élancés, ce qui introduit deux défis, notamment la modélisation et la conception de contrôleurs basés sur des modèles. Pour relever le premier défi, nous développons une technique de modélisation discrète appelée «déformation linéaire par morceaux» (PLS) pour résoudre les équations aux dérivées partielles (EDP) des modèles basés sur Cosserat pour les manipulateurs souples élancés, à partir desquels les modèles analytiques associés sont déduits. Pour résoudre le deuxième défi, les différents contrôleurs basés sur les modèles Cosserat proposés ont été conçus.

Mots-clefs: Robots mous élancés, déformation linéaire par morceaux Cosserat, modélisation dynamique, contrôleur basées sur le modèle

Cosserat-Based Modeling and Control of Slender Soft Robots

Summary in English

Soft robots are characterized by a large number of degrees of freedom (DoFs) and highly nonlinear deformation, which makes their modeling and accurate control difficult. Therefore, scientific challenges such as the development of an exhaustive theoretical framework for dynamic modeling and control have arisen in the field of soft robotics. In this thesis, we mainly focus on the slender soft robots, which introduces two challenges including the modeling and the model-based controller design. To address the first challenge, we develop a discrete modeling technique named piecewise linear strain (PLS) to solve the partial differential equations (PDEs) of Cosserat-based models for slender soft manipulators, based on which the associated analytic models are deduced. To solve the second challenge, the different controllers based on the proposed Cosserat models have been designed.

Keywords: Slender soft robots, piecewise linear strain Cosserat, dynamic modeling, model-based controller