Modeling and Sensing of Continuum and Parallel Robots for Physical Human-Robot Interaction

By

Andrew Leonidovich Orekhov

Dissertation

Submitted to the Faculty of the

Graduate School of Vanderbilt University

in partial fulfillment of the requirements

for the degree of

DOCTOR OF PHILOSOPHY

in

Mechanical Engineering

January 31, 2022

Nashville, Tennessee

Approved:

Nabil Simaan, Ph.D.

Howie Choset, Ph.D.

D. Caleb Rucker, Ph.D.

Eric J. Barth, Ph.D.

Michael Goldfarb, Ph.D.

ACKNOWLEDGMENTS

I would like to acknowledge and thank at least a few of the people, among many, without whom this dissertation would not be possible.

First, thank you to my advisor, Nabil Simaan. He is a dedicated and deeply knowledgable researcher who has pushed me to pursue excellence and integrity in all that I do. I have learned a lot from him, and I am blessed to have been a member of his lab and to have him as a mentor.

I would also like to thank the rest of my committee, Howie Choset, Caleb Rucker, Eric Barth, and Michael Goldfarb, for their helpful feedback on my work and for the unique ways in which each one of them has contributed to my training over the years. It's an honor to have them on my committee.

Many thanks to the ARMA Lab members for their friendship and for the many great memories both inside and outside the lab: Colette Abah, Garrison Johnston, Giuseppe Del Giudice, Elan Ahronovich, Neel Shihora, Jared Lawson, Rashid Yasin, Long Wang, and Nima Sarli. I'm also grateful to the undergraduate researchers who I had the privilege of working with on various projects: Jin Suh, Jeongwoo Seo, Zachary Taylor, Lauren Howard, and Jack Walton. Thank you as well to collaborators David Neiman and Yizhu Gu at Carnegie Mellon for their helpful insights and contributions.

Thank you to Myrtle Daniels for her diligence in helping me place purchase orders, and to John Fellenstein for his helpful design advice and excellent machining work.

I am grateful for financial support from the National Science Foundation through award #CMMI-1734461 and through the Graduate Research Fellowship Program under award #DGE-1445197.

I am also grateful to my parents, Leonid and Yelena, and to my siblings, Vitaliy, Viktor, and Galina, for their constant love and support in everything. I owe much of

my love for the pursuit of knowledge to the home environment I grew up in that my family cultivated, and I owe my interest in robotics to Viktor who first introduced me to this field.

Thank you to the members of Edgefield Church for supporting my spiritual life while I was in Nashville through prayer, Bible study, spiritual conversations, and regular worship together. Thank you for helping me hold fast to the gospel.

Finally, thank you to my Creator, because I know that “unless the Lord builds the house, those who build it labor in vain” (Psalm 127).

TABLE OF CONTENTS

LIST OF TABLES

Table Page

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 Background and Motivation

There is an increasing need for robots to operate in dynamically changing and uncertain environments, e.g. minimally invasive surgery, collaborative manufacturing, and inspection/maintenance in confined spaces. For robots to operate in these scenarios, they need to be able to not only maneuver in these uncertain environments but also detect unknown interaction forces along their body and be able to robustly control these physical interaction forces.

To address this need, there has been an increased interest in robots intentionally designed with significant passive compliance, e.g. rigid-link robots with joint compliance [6] and flexible continuum/soft robots [7, 8]. Compliant robots can provide improved force-control bandwidth and detection of unexpected contacts. In the case of continuum and soft robots, they also enable deep access into confined spaces due to their serpentine kinematic architectures.

Compliance in a robot, however, introduces modeling challenges due to deflections in the robot structure from external forces (whose location and magnitude may not be known). This uncertainty can be reduced with additional sensing, making the choice and placement of sensing modalities of particular importance for compliant robots. The additional compliance and need for sensing also has implications for the overall mechanical design of the robot, e.g. load-carrying capacity, dynamic behavior, safety, and sensor integration. This dissertation will address several research needs in each of these categories of modeling, sensing, and design, with a focus on the area of collaborative manufacturing, where a robot is operating in close proximity to a worker. We will first summarize the research needs in each of the areas that this dissertation will focus on, and then provide an overview of the specific contributions of each chapter in this dissertation.

1.1.1 Stiffness Modulation of Parallel Robots

One challenge in the area of robot design is how to design the passive compliance/stiffness of the robot when the environment is changing or a robot must carry out multiple tasks. The design of the passive stiffness matrix is important because it affects the system's force control bandwidth, modeling uncertainty, and robustness to unexpected disturbances. The end-effector stiffness matrix can to some extent be specified at the design stage, but the range of achievable stiffnesses is typically fixed for a given robot architecture. One way to expand the range of achievable stiffnesses is to change its stiffness online, for example by changing its internal joint force preload with antagonistic joints [9, 10, 11], changing its kinematic configuration [12, 13, 14], or by using variable stiffness actuators to change the joint-level stiffness [15, 16]. Each of these online stiffness modulation approaches has been studied individually for different robot architectures, but the combination of kinematic redundancy and variable stiffness actuation to expand the range of achievable stiffnesses has not been fully considered in prior work. We present a redundancy resolution scheme that allows kinematic redundancy and variable stiffness actuators to be used simultaneously for online stiffness modulation of parallel robots.

1.1.2 *In-Situ* Collaborative Robots for Confined Spaces

In addition to enabling stiffness modulation, kinematic redundancy can also expand the workspace and improve the dexterity of a robotic manipulator. One area where this is beneficial is collaborative manufacturing in confined spaces, examples of which include inspection/cleaning of tanks and maintenance/assembly of aircraft and ground vehicles. Working in a confined space is both a physiological burden for workers due to unergonomic postures and dangerous due to limited access. A collaborative robot deployed into the confined space, which we call here an *in-situ collaborative robot* (ISCR), could reduce these risks to the worker by either providing support via direct physical interaction, or by completely removing the worker from confined space via teleoperation.

In either case, the robot needs to have the necessary length and kinematic architecture to reach deep into a confined space, and it must deal with physical interaction forces both at its end effector and along its body. These forces may come from 1) forces at the end effector that occur while carrying out the task, 2) forces along the body for when the robot needs to brace against the environment to reduce joint torques [17], and 3) forces applied by the worker to guide the robot in carrying out the task.

Collaborative robots thus far have predominantly been rigid-link robots, which provide high precision but have the drawbacks of high inertia, potential for injury, and an unforgiving mechanical structure in the presence of misalignment with the environment. To mitigate some of these concerns, some have incorporated compliance at the joints via series-elastic actuation, but continuum robots are a promising design candidate for this specific application due to their improved reach into confined spaces, continuous compliance for accommodating contact anywhere along their body, and the potential to improve contact detection with shape sensing. In this work, we add an additional dimension to possible collaborative robot designs by investigating continuum robots as a candidate for collaborative manufacturing.

1.1.3 Shape Sensing of Continuum Robots

The passive compliance of a continuum robot introduces modeling uncertainties, especially when the location and/or magnitude of external loads are not accurately known. For this reason, sensing is needed to improve the estimate of the robot's shape. Sensing modalities in prior work for continuum robots include fiber Bragg grating optical fibers [18, 19], magnetic sensors [20, 21, 22], external cameras [23], and joint-level force sensors [24, 25]. Several works also measure the displacement of strings routed within the flexible structure by wrapping the string around a drum and measuring the drum angle with a potentiometer or an encoder [26, 27, 28, 29]. We will refer to this type of sensor as a *string encoder* since our implementation uses a magnetic encoder to measure the string displacement.

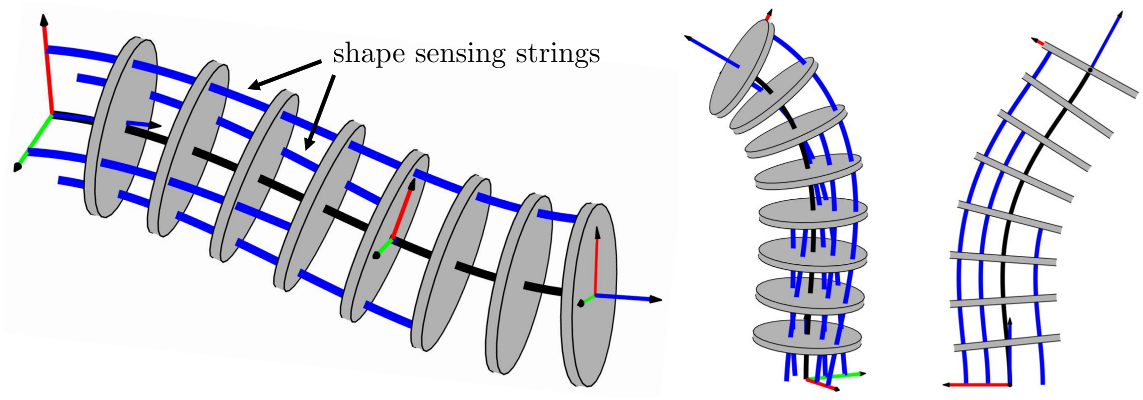


Figure 1.1: We propose general string encoder routing to sense the shape of a continuum segment.

The benefit of the string encoder approach over other methods include cost, ease of integration, and the fact that no external components like cameras or magnetic field generators are needed. However, existing works using string encoders either assume constant-curvature segments, are limited to planar cases, or assume constant pitch-radius string routings. In this dissertation, we provide a modeling formulation for shape sensing of variable curvature segments subject to spatial deflections with general string routing paths. The formulation uses a curvature-based modal shape basis in a

Lie group context. The modeling formulation allows us to describe general deflection shapes and provides analytic expressions for joint space, configuration space, and task space Jacobians. Using this formulation, we also provide guidance on how to design the string routing paths to improve the numerical conditioning of the shape sensing Jacobians. These kinematic equations also lay the groundwork for the mechanics modeling contributions of this dissertation.

1.1.4 Lie Group Methods for Statics and Compliance of Continuum Robots

As reviewed in [7, 8, 30, 31], a wide variety of modeling approaches have been presented in the literature for modeling continuum robots. Following [7], they can be categorized by their kinematic frameworks, e.g. rigid-link approximations, smooth constant-curvature approximations, or variable-curvature descriptions, and by their mechanics frameworks, e.g. lumped-parameter, energy-based, or classical elasticity models.

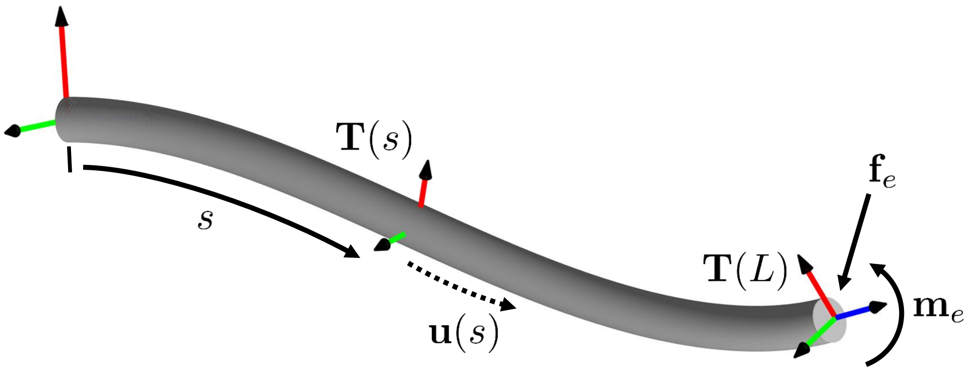


Figure 1.2: Cosserat rod models can describe variable curvature deflection shapes of continuum robots subject to external loading.

One popular variable-curvature approach based on classical elasticity models is the use of the Cosserat rod equations, which model both variable curvature and variable shear strain and can accommodate distributed and discrete loads. They require numerical integration of the governing equations as well as numerically solving a boundary value problem. This has been done in prior works using a shooting method

[32] and various collocation/weighted residuals methods [31, 33, 34, 35, 36], some of which describe the curvature and strains with polynomial shape functions of different orders [37, 38] and use Lie group formulations [39, 40, 41, 42].

Although Cosserat rod models have been successfully applied to several robot architectures, including concentric tube robots [43, 44], tendon-actuated robots [45], soft robots [39, 33], and parallel continuum robots [46], the computation cost of solving these equations is still a hindrance [47], especially when the robot is modeled as many coupled Cosserat rods, e.g. [48, 49]. Since the equations are typically solved numerically, there is also a lack of analytic expressions for these models. Analytic expressions are helpful for design and control where partial derivatives of the model equations are needed. For these reasons, there is still a need for new numerical and mathematical formulations for solving Cosserat rod models.

In this dissertation, we use global orthogonal polynomials on the curvature a the Lie group kinematic formulation to solve the Kirchhoff rod equations but neglecting the shear strains and extension. As described above, previous works have presented similar Lie group formulations and used collocation or modal shape descriptions of the curvature. We complement and extend these works by providing a unique combination of a Lie group method and global orthogonal collocation that provides computational benefits and useful analytical expressions.

In addition to computing the statics and deflection shape of a continuum robot, to achieve stiffness modulation and compliant motion control with continuum robots the stiffness/compliance matrix must be computed. An expression for the configuration space stiffness of constant curvature robots was given in [50], and methods to compute the task space compliance matrices of Cosserat rod models were given in [51, 25]. A gap in the literature is a method to compute the both the configuration space and task space compliance matrices of variable curvature continuum robots. We believe providing a way to compute these matrices will enable new studies of stiffness modulation methods and compliant motion controllers for continuum robots. In Chapter 6 we provide expressions for these matrices by leveraging the Lie group modal shape basis model that we used to compute the statics above.

1.2 Summary of Contributions and Overview

As described above, there are still a number of challenges associated with modeling and control of continuum and compliant parallel robots. As summarized by Fig. 1.3, this dissertation seeks to help address these challenges by making contributions in the area of online stiffness modulation of parallel robots and in the areas of design, modeling, and sensing of continuum robots for collaborative manufacturing. Here we will briefly restate the technical gaps in the literature, and summarize the contributions for each chapter in this dissertation.

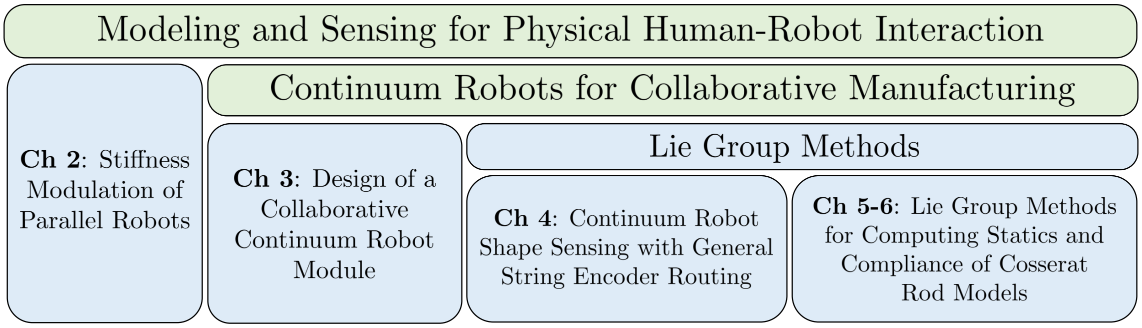

Modeling and Sensing for Physical Human-Robot Interaction
Continuum Robots for Collaborative Manufacturing
Lie Group Methods
Ch 2: Stiffness Modulation of Parallel Robots
Ch 3: Design of a Collaborative Continuum Robot Module
Ch 4: Continuum Robot Shape Sensing with General String Encoder Routing
Ch 5-6: Lie Group Methods for Computing Statics and Compliance of Cosserat Rod Models

Figure 1.3: Overview of the research areas addressed in this dissertation.

Redundancy Resolution for Stiffness Modulation of Parallel Robots: Prior works have studied stiffness modulation combining kinematic redundancy and variable stiffness actuators (VSAs), but these works are either not suitable for online stiffness modulation, are limited to planar cases, or neglect external loading in the stiffness model. In Chapter 2, we present an online redundancy resolution method that modulates the passive stiffness of a parallel robot by simultaneously changing the kinematic configuration and the joint-level stiffness. We show that combining these two approaches can expand the range of achievable stiffnesses and can generate desired directional stiffness values as well as desired spatial stiffness matrices.

Mechanical Design for In-Situ Collaborative Robots: Large-scale continuum robot design have been presented, but existing designs do not combine whole-body sensing, modularity, and high torsional stiffness, all of which are important for safe collaborative manufacturing in confined spaces. Chapter 3 presents the mechanical design of a tendon-actuated continuum module that is part of a new in-situ collaborative robot design that seeks to address these limitations. We describe both the actuation unit design and the mechanical components of the continuum structure. The design encapsulates all actuation components in the base of the segment to improve modularity of the system, a torsionally stiff compliant structure improves the load-carrying capacity of the segment, and multi-modal sensing disks and shape sensing strings provide proprioceptive sensing. These combined features, which are not present in other continuum robot designs, help address the unique challenges of collaborative manufacturing in confined spaces.

Shape Sensing via General String Routing: Although the use of string encoders has been previously demonstrated, all prior works assume the strings are routed in parallel paths and are limited to either constant-curvature shapes or to planar deflections. Chapter 4 presents a Lie group kinematic framework that models the robot curvature with modal shape functions, allows for shape sensing with general string encoder routing, and results in analytic expressions for Jacobians that can be computed efficiently. The Jacobians provide information about error sensitivity, and we discuss numerical methods for solving for the robot shape given a set of string encoder measurements. Our shape sensing approach by design avoids using a mechanics model, instead relying only on the kinematic equations to simplify implementation and reduce computation cost. Although we focus on shape sensing, our modeling approach also has relevance for kinematic modeling and control of tendon-actuated continuum robots and provides a compact formulation for kinematic calibration and identification as done in [52].

After presenting our kinematic formulation for shape sensing, we propose an algorithmic approach for designing the string routing to improve the kinematic conditioning of the shape sensing problem and avoid Jacobian singularities. We then validate the shape sensing and string routing optimization methods in both simulation and experimental studies, showing that the end disk position can be measured with errors below 5% of arc length.

Lie Group Methods for Statics and Compliance of Cosserat Rod Models: Cosserat rod models have been widely applied for continuum robots, and a variety of numerical methods for solving them have been presented, but a drawback of existing approaches is computational cost and a lack of analytical expressions for design and control. In Chapter 5, we present a numerical method for solving Cosserat rod models that combines the Lie group formulation used in Chapters 4 with global orthogonal polynomials on the Lie algebra (i.e. curvature) to solve the model.

Benefits of the approach are 1) the framework can be naturally applied to both low-order (e.g. constant-curvature models) and higher-order general deflection models and 2) the solution to the model is given as an analytic product of exponentials formula, 3) preliminary results indicate the computation cost is competitive with existing methods, and 4) the compliance of the robot in both configuration space and task space can be written analytically. Although some of the individual components of our approach have been presented in various prior works, these analytical and numerical benefits have not all been identified and fully explored in prior work, and the use of modal shape functions with Lie group methods to compute the compliance matrix has not been considered. In Chapter 5, we demonstrate our approach for solving the Kirchhoff rod model for the case of a single rod with a wrench applied at its tip in a simulation study.

In Chapter 6, we then build on this mechanics model formulation to present analytic expressions for the configuration space and task space compliance matrices for tendon-actuated continuum segments. We then discuss the different sensing information needed to compute these compliance matrices to guide future system designs for applications that require computation of the compliance matrix. We show that if a continuum robot is designed with high torsional stiffness and constant-pitch string paths (as we did for the design presented in Chapter 3) and the shape sensing formulation we present in Chapter 4 is used to determine the robot's configuration, computing the configuration space compliance requires no external sensing beyond the lengths of the actuation tendons and measurement strings.

To summarize, this dissertation addresses the following identified research needs in the context of collaborative manufacturing: 1) online stiffness modulation using kinematic redundancy and variable stiffness actuators, 2) mechanical design of collaborative continuum robots, 3) shape sensing of variable curvature continuum robot deflections with general string encoder routing, and 4) methods for computing statics and compliance of Cosserat rods that provide analytic expressions and computational efficiency.

CHAPTER 2

DIRECTIONAL STIFFNESS MODULATION OF PARALLEL ROBOTS WITH KINEMATIC REDUNDANCY AND VARIABLE STIFFNESS JOINTS

This chapter is adapted from [53] and has been reproduced with the permission of the publisher and the co-authors.

2.1 Introduction

Parallel manipulators are often used due to their potential for increasing precision and structural rigidity, but compared to serial architectures, their workspace is limited and they exhibit additional types of singularities characterized by loss of rigidity. For this reason, many works have studied the use of redundancy to improve the workspace, stiffness, and other performance characteristics of parallel manipulators [54, 55].

Parallel mechanisms admit two types of redundancy: *Actuation redundancy* refers to robots using more active joints than the minimal number required to satisfy wrench equilibrium of the end effector, while *kinematic redundancy* uses more active joints than the minimal number needed to satisfy a desired twist of the end effector. Although these types of redundancy can be further categorized (e.g. [56]), we will focus in this chapter on the use of kinematic redundancy to modulate the directional stiffness of parallel robots.

There are applications that would benefit from specific characteristics of compliance. One example is assembly of parts where there is some positional uncertainty in the system. While such tasks have historically been addressed using specific remote center of compliance mechanisms at the tip of industrial robots, the growing demand for human-robot collaboration in assembly lines demands robust and safe systems that can collaborate on assembly or manufacturing tasks. Parallel robots with stiffness modulation capabilities could have a wide range of applications for tasks such as deburring, grinding, and collaborative assembly. In addition, hybrid serial-parallel robots for human robot collaboration (e.g. [57]) could benefit from parallel modules capable of a wide range of selective stiffness variations.

To change stiffness, one may use closed-loop control approaches or open-loop methods. We will use the terminology *stiffness control* to refer to methods using dynamic motion control to achieve a desired interaction stiffness or motion impedance. Such methods require real-time measurements of the end effector forces and accurate robot dynamics (e.g. [58, 59]) and they are susceptible to control instabilities and bandwidth limitations. We will use the term *stiffness modulation* to designate methods not involving dynamic closed-loop control, but rather open loop design of stiffness. Stiffness modulation methods change the stiffness properties of the manipulator by changing its passive structural stiffness, e.g. its geometry or internal preload. These methods typically involve additional complexity in the form of extra actuators and transmission mechanisms but potentially allow for a wide range of stiffnesses to be achieved without closed-loop interaction control. This chapter will focus on stiffness modulation of parallel mechanisms while using variable stiffness actuators and kinematic redundancy.

Stiffness modulation can be achieved using a) antagonistic actuation or preload (e.g. [9, 10, 11]), b) kinematic redundancy for changing the stiffness via change of geometry (e.g. [12, 13, 14]), or c) using variable stiffness actuators to affect joint-space stiffness (e.g. [15, 16]). Details of these works and their limitations are presented in Section 2.2 of this chapter. Briefly, these limitations can be summarized as the lack of a framework for online stiffness modulation of spatial parallel robots using kinematic redundancy with or without the use of variable stiffness actuators.

To overcome these limitations, this chapter will explore the feasibility of combined use of real-time kinematic redundancy and variable stiffness actuators (VSAs) for achieving stiffness modulation of parallel robots. As opposed to attempting stiffness synthesis (e.g. [12]), which is known to be limited to the physical realization of the springs representing the robot structure ([60, 61]), we define the task of stiffness modulation to be based on the concept of directional stiffness. The concept of directional stiffness was presented in [62], but in this chapter we extend this concept to allow for translational-rotational stiffness coupling to be specified. In addition, we present a gradient-based redundancy resolution approach to solving both the kinematic redundancy and VSA stiffness with the aim of modulating the directional stiffness while satisfying additional tasks such as avoiding joint limits, collisions between legs, and kinematic singularities.

2.2 Relevant Works and Summary of Limitations

The following section reviews relevant works on stiffness modulation with the aim of elucidating the scientific gaps addressed by this chapter.

Actuation redundancy: Most works on stiffness modulation of parallel linkages and cable-driven platforms have focused on the use of actuation redundancy (e.g. [9, 63, 11, 64, 65, 66, 62]). There are three key limitations to using actuation redundancy to achieve a desired stiffness: 1) the attainable relative change in stiffness due to use of antagonistic actuation depends on the nominal stiffness of the actuators. High stiffness non-backdrivable actuators will correspond with a small attainable change in end-effector stiffness due to internal pre-load, and low stiffness actuators (e.g. pneumatic actuators) will allow a larger relative change in end-effector stiffness [67], 2) real-time control using actuation redundancy requires direct measurements of joint forces which can be difficult to obtain, 3) naïve implementation of actuation redundancy can possibly lead to reduced robustness in the neighbourhood of singularity, as first mentioned in [68]. Furthermore, geometric calibration errors have been shown in [69] to degrade motion tracking performance due to parasitic forces that cannot be compensated.

Kinematic redundancy: Kinematic redundancy has been recently studied for stiffness optimization in planar cable-driven robots [14, 13]. The stiffness synthesis of a spatial variable geometry parallel robot was studied in [12] by finding all solutions for a polynomial system. These aforementioned works are either limited to planar cases and/or to offline joint-space path planning.

Variable Stiffness Actuation: An additional method for modulating stiffness is to use variable stiffness actuators (VSAs), which typically consist of a variable-stiffness mechanism in series with the actuation input in order to allow for the joint-level stiffness to be modulated. This chapter does not consider the scenario of passive joints with stiffness modulation, but the approach presented herein can be easily applied to consider such designs. Many variable stiffness actuator/joint designs have been presented in the literature, a review of which is provided in [6]. Examples of methods for modulating the open-loop joint stiffness include changing the preload in a nonlinear spring [70], changing the transmission ratio between the load and the elastic element using various mechanisms [71], and changing the physical geometry of the elastic element [72]. The use of VSAs for stiffness control in serial manipulators in humanoid robots has been studied [73], and there are a small number of works using VSAs for stiffness modulation of planar parallel robots [15, 74, 75, 16].

A recent work [76] has presented a method combining kinematic redundancy and VSAs for online stiffness modulation of serial manipulators. Our method is similar to [76], but we investigate spatial stiffnesses with a parallel kinematic structure and we include the Jacobian derivative term in the stiffness matrix associated with external loading.

A case study utilizing all three of the above stiffness modulation approaches was recently done in [77] for a planar cable-driven leg-orthosis simulator, which considered various cost functions to optimize stiffness (e.g. maximize or generate an isotropic stiffness) while satisfying cable tension and joint position/velocity constraints. In this chapter, we consider similar problems (excluding actuation redundancy) but focus on spatial parallel robots with local, online redundancy resolution methods.

Summary of limitations: To summarize the limitations of stiffness modulation works, stiffness modulation via antagonistic actuation has been widely studied (including methods for real-time force redundancy resolution for stiffness modulation), however, this method requires either very high antagonistic loading or joints with low stiffness to achieve a significant change in the stiffness of the robot. We therefore exclude this approach from the scope of the current study. Works on the use of kinematic redundancy and VSAs for stiffness modulation either lack approaches for online stiffness modulation, are limited to planar cases, or neglect external loading in the stiffness matrix.

2.3 Problem Formulation and Nomenclature

Fig. 2.1 shows a 6-$\underline{\text{R}}$S$\underline{\text{P}}$S Gough/Stewart type parallel robot having n extensible legs that slide on circular tracks. Each kinematic chain uses an active revolute joint ($\underline{\text{R}}$), a passive spherical joint (S), an active prismatic joint ($\underline{\text{P}}$) and a passive spherical joint (S). The robot has six legs with two active joints per leg, resulting in a kinematic redundancy of degree 6. Similar parallel architectures with movable base anchor points have been studied [78, 79, 80, 81, 82], although none of these examples utilized kinematic redundancy.

A typical mechanical implementation of the extensible legs of this robot would be a linear slide or bearing actuated by a ball-screw drive. For actuating the base anchor points, several mechanical implementations are possible. Examples include a circular rack and pinion (e.g. [83, 84]), a circular guide rail with a wire-rope capstan drive,

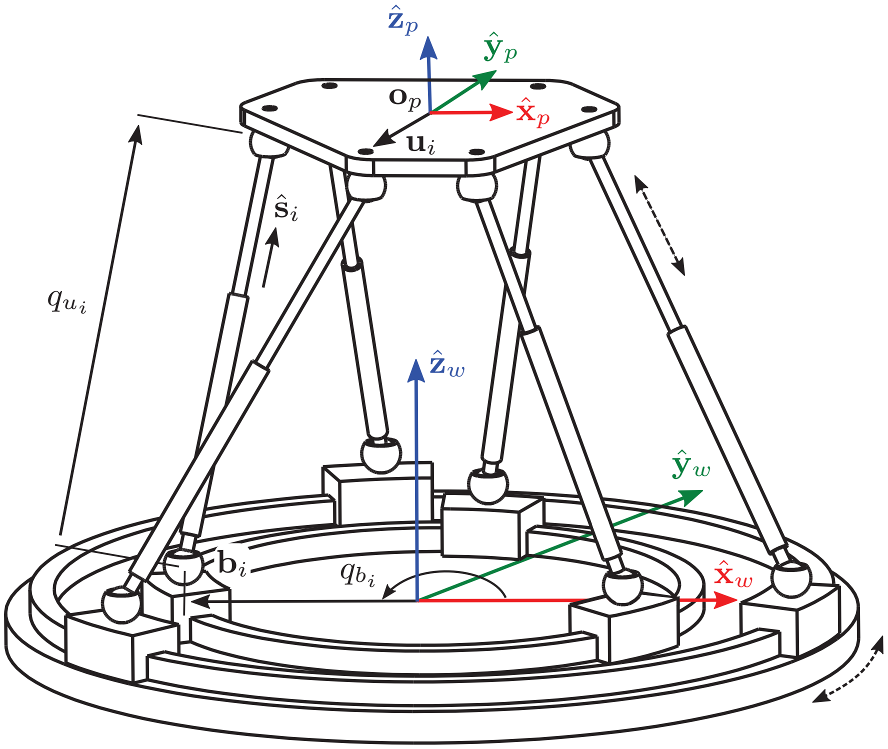


Figure 2.1: A Stewart-Gough type parallel manipulator with kinematic redundancy introduced through movable base anchor points.

and a piezoelectric or ceramic motor on a circular drive strip. Although we limit this work to circular motion at the base, the methods in this chapter could be used for a robot with planar stepper motors (e.g. Sawyer motors [85]) for actuating the base anchor points, as was done in [81].

Referring to Fig. 2.1, we will use $\mathbf{b}_i$, $i = 1 \ldots 6$ to designate the locations of the base anchor points in world frame {W}. Also, $\hat{\mathbf{s}}_i$ denotes the unit vector along the i^{th} leg, and $\mathbf{u}_i$ is the location of the i^{th} anchor point on the moving frame, expressed in the moving platform frame {P}[1]. The *pose* of the moving platform is denoted $\mathbf{x} \in \mathbb{R}^{6\times 1}$ such that $\mathbf{x} \triangleq [\mathbf{o}_p^{\mathrm{T}}, \boldsymbol{\sigma}^{\mathrm{T}}]^{\mathrm{T}}$ where $\mathbf{o}_p$ is the gripper position and $\boldsymbol{\sigma}$ is a 3-dimensional vector representing a parametrization of ${}^{w}\mathbf{R}_p$ (the moving platform's orientation).

[1] We will use {A} to denote a right-handed frame having $\hat{\mathbf{x}}_a, \hat{\mathbf{y}}_a, \hat{\mathbf{z}}_a$ as its unit vectors and $\mathbf{o}_a$ as its origin. Also, ${}^{b}\mathbf{R}_a$ will be used to denote a rotation transforming vector representation from frame {A} to frame {B}.

In addition, we will use $\mathbf{q} \in \mathrm{I\!R}^{2n}$ and $\boldsymbol{\tau} \in \mathrm{I\!R}^{2n}$ to designate the vectors of active joints and their corresponding joint forces/moments. It will be assumed that $\mathbf{q} \triangleq [\mathbf{q}_u^{\mathrm{T}}, \mathbf{q}_b^{\mathrm{T}}]^{\mathrm{T}}$ where $\mathbf{q}_u \in \mathrm{I\!R}^n$ designates the lengths of the leg pistons and $\mathbf{q}_b \in \mathrm{I\!R}^n$ denotes the angular coordinates of the sliding base anchor points. The angular joint values $\mathbf{q}_{b_i},\ i = 1 \ldots 6$ are measured from $\hat{\mathbf{x}}_w$ to $\mathbf{b}_i$ about $\hat{\mathbf{z}}_w$ (with the positive direction defined according to the right hand rule). We will also use $\boldsymbol{\tau}_u$ and $\boldsymbol{\tau}_b$ to denote the joint forces/moments corresponding with $\mathbf{q}_u$ and $\mathbf{q}_b$, respectively. We define joint-level stiffness as the stiffness of each of the actuators along the direction of travel. The joint-level stiffness is given by $\mathbf{k}_q \triangleq [\mathbf{k}_{q_u}^{\mathrm{T}}, \mathbf{k}_{q_b}^{\mathrm{T}}]^{\mathrm{T}}$ where $\mathbf{k}_{q_u} \in \mathrm{I\!R}^n$ designates the joint-level stiffness of the prismatic actuators and $\mathbf{k}_{q_b} \in \mathrm{I\!R}^n$ designates the joint-level stiffness of the rotary actuators.

Although the robot of Fig. 2.1 has 12 actuators, it does not possess actuation redundancy since each kinematic chain transmits a single force to the moving platform - therefore forming a 6-dimensional basis of zero-pitch wrenches[2] capable of resisting any general wrench on the moving platform. Each force is along the direction of its corresponding active prismatic joint. If the robot was designed with $n > 6$, one could theoretically use actuation redundancy in an antagonistic actuation mode. We do not consider actuation redundancy due to the large body of prior works on this method. We also assume that the prismatic joints are fitted with a variable stiffness mechanism to modulate the joint-level stiffness.

The problem we explore in this chapter is the modulation of the end effector's stiffness $\mathbf{k}_\beta$ in a set of task-specific directions. Specifically, we will explore the use of kinematic redundancy to specify joint speeds $\dot{\mathbf{q}}$ and VSAs to specify rates of change in joint-level stiffness $\dot{\mathbf{k}}_q$ to achieve a particular directional stiffness while executing a commanded end effector twist[3] $\boldsymbol{\xi}$ and respecting the instantaneous inverse kinematics

[2] We use a wrench representation following ray-coordinates of their screw axis, i.e. with force preceding moment.

[3] We define the twist in axis-coordinates, i.e. with linear velocity preceding angular velocity.

as specified by the Jacobian $\mathbf{J}$. This problem may be stated as:

$$\begin{aligned} \underset{\dot{\mathbf{q}},\,\dot{\mathbf{k}}_q}{\text{minimize}} \quad & f\left(\mathbf{k}_\beta(\mathbf{q},\mathbf{x},\mathbf{k}_q)\right) \\ \text{subject to} \quad & \mathbf{J}\boldsymbol{\xi} - \dot{\mathbf{q}} = \mathbf{0}. \end{aligned} \tag{2.1}$$

where the objective function f is defined to optimize open-loop stiffness for a particular task. Detailed descriptions of each of the variables in Eq. (2.1), as well as ways to define the objective function f, are given below. Since this work is motivated by applications in human-robot interaction, we consider the case where the prismatic legs are equipped with compliant VSAs, while the base anchor points are actuated using substantially stiffer transmissions. The assumption is that the user interacts with the end effector/moving platform (hence the use of VSAs) and that the base actuators are contained at the base in a manner that reduces risk to the user.

2.4 Parallel Kinematics and Stiffness Preliminaries

The instantaneous kinematics takes the form in Eq. (2.2). This equation can be derived either using a reciprocal screws formulation [86] or using loop closure differentiation [87, 88].

$$\mathbf{J}_x\boldsymbol{\xi} = \mathbf{J}_q\dot{\mathbf{q}} \tag{2.2}$$

where $\dot{\mathbf{q}} \triangleq [\dot{\mathbf{q}}_u^{\mathrm{T}}, \dot{\mathbf{q}}_b^{\mathrm{T}}]^{\mathrm{T}}$ is a vector of joint speeds and $\boldsymbol{\xi}$ is the end effector twist defined as the linear velocity followed by the angular velocity of the end effector, expressed in the spatial frame. $\mathbf{J}_x$ is often referred to as the "parallel Jacobian" and $\mathbf{J}_q$ is often referred to as the "serial Jacobian." The parallel Jacobian is known for a Stewart-

Gough platform:

$$\mathbf{J}_x = \begin{bmatrix} \$_1^{\mathrm{T}} \\ \vdots \\ \$_n^{\mathrm{T}} \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{s}}_1^{\mathrm{T}}, & ({}^w\mathbf{R}_p\mathbf{u}_1 \times \hat{\mathbf{s}}_1)^{\mathrm{T}} \\ & \vdots \\ \hat{\mathbf{s}}_n^{\mathrm{T}}, & ({}^w\mathbf{R}_p\mathbf{u}_n \times \hat{\mathbf{s}}_n)^{\mathrm{T}} \end{bmatrix} \in \mathrm{I\!R}^{6\times 6} \tag{2.3}$$

where $\$_i, i = 1 \ldots n$ are the Plücker line coordinates of the screw associated with the prismatic joints. For the robot of Fig. 2.1, the serial Jacobian is:

$$\mathbf{J}_q = \left[\begin{array}{c|c} \mathbf{I}_{6x6} & \begin{bmatrix} \|\mathbf{b}_1\|\hat{\mathbf{t}}_1^{\mathrm{T}}\hat{\mathbf{s}}_1 & & \mathbf{0} \\ & \ddots & \\ \mathbf{0} & & \|\mathbf{b}_6\|\hat{\mathbf{t}}_6^{\mathrm{T}}\hat{\mathbf{s}}_6 \end{bmatrix} \end{array} \right] \tag{2.4}$$

where $\hat{\mathbf{t}}_i^{\mathrm{T}}, i = 1 \ldots n$ designate the local tangent unit vectors to the curvilinear tracks at the base and $\|\mathbf{b}_i\|$ is the radial distance to the base anchor point. The positive directions of $\hat{\mathbf{t}}_i$ are defined according to the right-hand rule about $\hat{\mathbf{z}}_w$.

2.4.1 Stiffness Matrix

The stiffness of a parallel Stewart-Gough manipulator is formulated using methods as in [89, 11]. The following derivation retraces steps described in [90] and is included here for completeness. The stiffness matrix is defined as:

$$\delta\mathbf{w}_e = \mathbf{K}\delta\mathbf{x} \tag{2.5}$$

where $\delta\mathbf{x}$ is a perturbation of the moving platform pose and $\delta\mathbf{w}_e$ is a perturbation in the wrench $\mathbf{w}_e$ applied by the moving platform on the environment. Using virtual work, the statics of the robot is given by:

$$\boldsymbol{\tau} = \left(\mathbf{J}_x^{-1}\mathbf{J}_q\right)^{\mathrm{T}}\mathbf{w}_e \tag{2.6}$$

A perturbation of $\boldsymbol{\tau}$ around equilibrium is obtained as:

$$d\boldsymbol{\tau} = d\left(\left(\mathbf{J}_x^{-1}\mathbf{J}_q\right)^{\mathrm{T}}\right)\mathbf{w}_e + \left(\mathbf{J}_x^{-1}\mathbf{J}_q\right)^{\mathrm{T}} d\mathbf{w}_e \tag{2.7}$$

We now define the matrix $\mathbf{K}_q \triangleq \dfrac{d\boldsymbol{\tau}}{d\mathbf{q}}$, which is a $2n \times 2n$ diagonal matrix containing the joint-level stiffnesses as $\mathbf{K}_q = \mathrm{diag}(\mathbf{k}_{q_u}, \mathbf{k}_{q_b})$. Using the definition of $\mathbf{K}_q$ and rearranging Eq. (2.7) we obtain:

$$\left[\mathbf{K}_q - \mathbf{K}_g\right] d\mathbf{q} = \left(\mathbf{J}_x^{-1}\mathbf{J}_q\right)^{\mathrm{T}} d\mathbf{w}_e \tag{2.8}$$

where $\mathbf{K}_g$ is given by:

$$\mathbf{K}_g = \left[\frac{\partial(\mathbf{J}_x^{-1}\mathbf{J}_q)^{\mathrm{T}}}{\partial q_1}\mathbf{w}_e, \quad \cdots \quad, \frac{\partial(\mathbf{J}_x^{-1}\mathbf{J}_q)^{\mathrm{T}}}{\partial q_{2n}}\mathbf{w}_e\right] \tag{2.9}$$

Note that $\mathbf{K}_g$ contains second-order effects due to perturbation of the Jacobians as a pose perturbation $d\mathbf{x}$ occurs from the robot applying an external load $\mathbf{w}_e$ on the environment. Solving Eq. (2.8) for $d\mathbf{q}$ and substituting into the pose perturbation $d\mathbf{x} = \mathbf{J}_x^{-1}\mathbf{J}_q d\mathbf{q}$ given by Eq. (2.2), the compliance matrix $\mathbf{C} \triangleq \dfrac{d\mathbf{x}}{d\mathbf{w}_e}$ can be easily found. Inverting the compliance matrix [4] results in the stiffness matrix $\mathbf{K}$:

$$\mathbf{K} = \left[\left(\mathbf{J}_x^{-1}\mathbf{J}_q\right)\left(\mathbf{K}_q - \mathbf{K}_g\right)^{-1}\left(\mathbf{J}_x^{-1}\mathbf{J}_q\right)^{\mathrm{T}}\right]^{-1} \tag{2.10}$$

Since this robot has no actuation redundancy, Eq. (2.10) suggests that the stiffness can be modulated using either kinematic redundancy to change the Jacobians and their derivatives or VSAs to change the diagonal elements of $\mathbf{K}_q$.

As opposed to specifying the full 6 × 6 stiffness matrix, in some scenarios, it is more convenient to specify a *directional stiffness.* This is sometimes advantageous

[4] We exclude singular positions to obtain this inversion. The redundancy resolution is used in the following section to ensure singularity avoidance.

because specifying the entire stiffness may result in a physically unrealizable stiffness for a given robot architecture [91] or for a given set of kinematic redundancy joints [12]. We will follow a similar derivation of directional stiffness as in [62], except we allow for different directions in force and displacement to permit more general stiffness behaviors to be specified. Using the stiffness matrix definition in Eq. (2.5), the wrench produced by a twist deflection of a small magnitude δx_{β_x} about a unit screw[5] $\boldsymbol{\beta}_x \in \mathrm{I\!R}^{6\times 1}$ is:

$$\delta \mathbf{w}_e = \mathbf{K}\boldsymbol{\beta}_x \delta x_{\beta_x} \tag{2.11}$$

The work done by the wrench $\delta \mathbf{w}_e$ if the moving platform were to move along a unit twist $\boldsymbol{\beta}_w$ is:

$$\boldsymbol{\beta}_w^{\mathrm{T}} \delta \mathbf{w}_e = \boldsymbol{\beta}_w^{\mathrm{T}} \mathbf{K}\boldsymbol{\beta}_x \delta x_{\beta_x} \tag{2.12}$$

The expression $\boldsymbol{\beta}_w^{\mathrm{T}} \mathbf{K}\boldsymbol{\beta}_x$ is the work per unit deflection about the screw $\boldsymbol{\beta}_x$. This also is the stiffness of an imaginary screw spring constrained to deflect about $\boldsymbol{\beta}_w$. Therefore, we define the directional stiffness as:

$$k_{dir} = \frac{\boldsymbol{\beta}_w^{\mathrm{T}} \delta \mathbf{w}_e}{\delta x_{\beta_x}} = \boldsymbol{\beta}_w^{\mathrm{T}} \mathbf{K}\boldsymbol{\beta}_x \tag{2.13}$$

In the case where $\boldsymbol{\beta}_w$ represents an infinite pitch twist (linear velocity) the expression $\boldsymbol{\beta}_w^{\mathrm{T}} \mathbf{K}\boldsymbol{\beta}_x$ represents the translational stiffness along the screw axis unit vector of $\boldsymbol{\beta}_w$.

This generalization of the directional stiffness in [62] is useful for tasks where potential perturbations may not be along the same directions as the forceful interactions, and is particularly relevant for spatial stiffness matrices due to potential coupling between translational and rotational stiffness. An example is in assembly (e.g. peg-in-hole insertion), where it desirable to have a rotational deflection in reaction to a lateral reaction force so as to mimic the function of a remote center of compliance

[5] A unit twist in axis coordinates is a 6-vector that has been normalized to result in unit-vector angular velocity. In the case of infinite pitch, normalization is carried out such that the linear velocity is a unit vector.

mechanism (i.e. rotate the peg to realign it with the axis of the hole so as to avoid mechanical lock). The following sections describe our method for simultaneously modulating several directional stiffnesses for a given task.

2.5 Stiffness Modulation via Kinematic Redundancy

The general solution to the instantaneous inverse kinematics in Eq. (2.2) is given by:

$$\dot{\mathbf{q}} = \mathbf{J}_q^+ \mathbf{J}_x \boldsymbol{\xi} + (\mathbf{I} - \mathbf{J}_q^+ \mathbf{J}_q)\boldsymbol{\eta} \tag{2.14}$$

where $\mathbf{J}_q^+$ is the minimum norm pseudo-inverse, $\boldsymbol{\eta}$ is any vector of joint speeds, and $(\mathbf{I} - \mathbf{J}_q^+ \mathbf{J}_q)$ is a null-space projector that projects $\boldsymbol{\eta}$ into the null-space of $\mathbf{J}_q$ such that the primary task $\boldsymbol{\xi}$ is not affected. Gradient projection is a method for choosing $\boldsymbol{\eta}$ to locally optimize some objective function and is formulated as:

$$\dot{\mathbf{q}} = \underbrace{\mathbf{J}_q^+ \mathbf{J}_x \boldsymbol{\xi}}_{\dot{\mathbf{q}}_p} + \alpha \underbrace{(\mathbf{I} - \mathbf{J}_q^+ \mathbf{J}_q)\nabla g(\mathbf{q})}_{\dot{\mathbf{q}}_h} \tag{2.15}$$

where $g(\mathbf{q})$ is the objective function to be locally maximized and α is a user-defined gain that determines the step size taken along the gradient [92]. The problem of choosing α has been studied in several works [93, 94, 95]. Since the goal is to maximize the objective function $g(\mathbf{q})$, a maximal value of α is sought while respecting joint speed limits. The range of values for α that will not violate the joint speed limits of the i^{th} joint according to [96] are:

$$\alpha_{i,max} = \max \left\{ \frac{\dot{q}_{iu} - \dot{q}_{ip}}{\dot{q}_{ih}}, \frac{\dot{q}_{il} - \dot{q}_{ip}}{\dot{q}_{ih}} \right\} \tag{2.16}$$

where $\dot{q}_{iu}$ and $\dot{q}_{il}$ are the maximum and minimum joint speed of the i^{th} joint, respectively. The maximal allowable value for α is:

$$\alpha_{max} = \min\left\{\alpha_{1,max}, \cdots, \alpha_{2n,max}\right\} \tag{2.17}$$

When the manipulator approaches the optimal solution, large oscillatory joint velocities can occur. To prevent this, a function was proposed in [95] to damp these velocities:

$$f_a = 1 - e^{-\lambda \|\dot{\mathbf{q}}_h\|_\infty}, \quad \lambda > 0 \tag{2.18}$$

where λ is a user-defined scalar that determines the amount of damping and $\|\dot{\mathbf{q}}_h\|_\infty$ denotes the infinity norm of $\dot{\mathbf{q}}_h$. Finally, α is given by:

$$\alpha = f_a \alpha_{max} \tag{2.19}$$

2.6 Choosing the Function $g(\mathbf{q})$

In addition to expanding the range of achievable stiffnesses, a parallel robot with kinematic redundancy can also utilize its redundancy to optimize performance measures or satisfy other constraints. These additional tasks can be incorporated into the objective function of the gradient projection scheme with a weighted sum. In this work, three weighted metrics are used as follows:

$$g(\mathbf{q}) = -w_1 C_j - w_2 C_d - w_3 C_s + \gamma\left[\sum_{i=4}^{n} w_i C_{k,i}\right] \tag{2.20}$$

where C_j is a measure for avoiding joint limits, C_d is a measure to avoid prismatic leg collisions, C_s is a measure used to avoid singularities, $C_{k,i}$ is a measure used for either 1) maximizing stiffness or 2) generating a specific stiffness, and γ is an additional scaling gain described below. The following sections will define each of these measures and provide the analytical solution for their gradients.

2.6.1 Defining C_k: Optimizing Directional Stiffness

2.6.1.1 Maximizing Directional Stiffness

To maximize stiffness, we define C_k as:.

$$C_k = k_{dir} = \boldsymbol{\beta}_w^{\mathrm{T}} \mathbf{K} \boldsymbol{\beta}_x \tag{2.21}$$

The gradient for the i^{th} joint is given by:

$$\nabla k_{dir}(q_i) = \frac{\partial k_{dir}}{\partial q_i} = \boldsymbol{\beta}_w^{\mathrm{T}} \frac{\partial \mathbf{K}}{\partial q_i} \boldsymbol{\beta}_x \tag{2.22}$$

We use the analytical solution to the stiffness matrix gradient in the simulations below. Alternatively, the gradient can be estimated numerically via finite differences.

2.6.1.2 Generating a Specific Stiffness

Rather than maximize the directional stiffness, a specific directional stiffness can be commanded by redefining C_k as:

$$C_k = -\frac{1}{2}(k_{dir,des} - k_{dir})^2 \tag{2.23}$$

where $k_{dir,des}$ is the desired directional stiffness and k_{dir} is the directional stiffness at the current time step. The partial derivative with respect to the i^{th} joint is:

$$\nabla C_k(q_i) = (k_{dir,des} - k_{dir}) \nabla k_{dir}(q_i) \tag{2.24}$$

2.6.2 Defining C_s: Singularity Avoidance

Reconfiguring the base anchor points can lead to singular configurations. To avoid singularity, we use the Frobenius norm condition number of the parallel Jacobian due its ease of computation [97] :

$$C_s = \kappa = \|\mathbf{J}_x^{-1}\|_F \|\mathbf{J}_x\|_F \tag{2.25}$$

$$\|\mathbf{J}_x\|_F = \sqrt{\tfrac{1}{6}\mathrm{tr}(\mathbf{J}_x \mathbf{J}_x^{\mathrm{T}})} \tag{2.26}$$

The partial derivatives $\partial\kappa/\partial\mathbf{q}$ are symbolically computed, but are not included here for brevity.

2.6.3 Defining C_j: Joint Limit Avoidance

Joint limit avoidance can be incorporated by defining a function that approaches infinity as the joint limits are approached [98]:

$$C_j = \sum_{i=1}^{12} \frac{q_{i,max} - q_{i,min}}{(q_{i,max} - q_i)(q_i - q_{i,min})} \tag{2.27}$$

$$\nabla C_j(q_i) = \frac{-(q_{i,max} - q_{i,min})(q_{i,max} - 2q_i + q_{i,min})}{\left[(q_{i,max} - q_i)(q_i - q_{i,min})\right]^2} \tag{2.28}$$

For the base anchor points, the joint limits must be actively updated at each time step to prevent collisions among points on a common track. The prismatic joint limits are constant.

2.6.4 Defining C_d: Leg Collision Avoidance

The reconfiguration of the base anchor points can result in collisions between the prismatic legs of the robot. Since manipulator links can be modeled as cylindrical volumes, some works have studied the problem of detecting collisions between cylindrical bodies [99, 100], including collision detection specifically for parallel robots [101, 102]. Here, rather than detect collisions, the goal is to prevent collisions by enforcing a minimum distance between each of the legs. A cost function that can be used to enforce a minimum value for the distance d of all 15 leg pairs is:

$$C_d = \sum_{i=1}^{15} \frac{d_{min}}{d_i - d_{min}} \tag{2.29}$$

This function will approach infinity as the minimum distance between any two legs is approached during robot motion. The partial derivative can be written:

$$\frac{\partial C_d}{\partial q_i} = \frac{\partial C_d}{\partial d}\frac{\partial d}{\partial q_i} \tag{2.30}$$

A method for calculating d is provided in Appendix A. The gradient of d_i is discontinuous in the instances where the location of the minimum distance is at the endpoint of a line segment and when the legs are parallel. The effect of these discontinuities is expected to be small if the minimum distance between the base anchor points in Eq. (2.27) is set to a value larger than d_{min}.

2.6.5 Choosing the directional stiffness scaling gain γ

When using variable stiffness actuators with a large range of achievable stiffnesses, the gradient of C_k can vary significantly across the range of stiffnesses. We therefore use a variable gain γ in order to counteract the effect of wide variations of joint level stiffness on the last element in Eq. (2.20), which in turn can aggressively affect the joint speeds in Eq. (2.15). The choice of scaling used here is $\gamma = 10^{-s}$, where $s = \log_{10} \overline{\mathbf{k_q}}$ and $\overline{\mathbf{k_q}}$ is the mean of all elements in $\mathbf{k_q}$. It is unnecessary to scale C_j, C_d, and C_s since they are not a function of the joint stiffnesses.

This completes the formulation for real-time stiffness modulation utilizing the robot's kinematic redundancy. We now present a method for commanding the variable stiffness actuators for directional stiffness modulation.

2.7 Stiffness Modulation via Variable Stiffness Actuators

Stiffness modulation can also be achieved using variable stiffness actuators to modulate the joint-level stiffnesses. Changes in the joint level stiffness $\mathbf{k}_q$ can be generated using one of a variety of different VSA mechanisms, as reviewed in [6]. Assuming the goal is to modulate several directional stiffnesses as defined in Eq. (2.13), we define a directional stiffness vector $\mathbf{k}_{dir} \triangleq [k_{dir_1} \ldots k_{dir_j}]^{\mathrm{T}}$ and we use the Jacobian relating changes in joint-level stiffness to changes in $\mathbf{k}_{dir}$:

$$\dot{\mathbf{k}}_{dir} = \mathbf{J}_k \dot{\mathbf{k}}_q \tag{2.31}$$

$$\mathbf{J}_k = \begin{bmatrix} \boldsymbol{\beta}_{w_1}^{\mathrm{T}} \frac{\partial \mathbf{K}}{\partial k_{q_1}} \boldsymbol{\beta}_{x_1} & \cdots & \boldsymbol{\beta}_{w_1}^{\mathrm{T}} \frac{\partial \mathbf{K}}{\partial k_{qn}} \boldsymbol{\beta}_{x_1} \\ \vdots & & \vdots \\ \boldsymbol{\beta}_{w_j}^{\mathrm{T}} \frac{\partial \mathbf{K}}{\partial k_{qn}} \boldsymbol{\beta}_{x_j} & \cdots & \boldsymbol{\beta}_{w_j}^{\mathrm{T}} \frac{\partial \mathbf{K}}{\partial k_{qn}} \boldsymbol{\beta}_{x_j} \end{bmatrix} \in \mathrm{I\!R}^{j \times n}, \tag{2.32}$$

Noting that the $\mathbf{J}_x$ and $\mathbf{J}_q$ are independent of the joint-level stiffness, the gradient $\frac{\partial \mathbf{K}}{\partial k_{qn}}$ is simple to find in closed-form. The general solution to Eq. (2.31) while avoiding the joint-level stiffness limits and maximum joint-level stiffness rate limits is found in a similar fashion as above:

$$\dot{\mathbf{k}}_q = \mathbf{J}_k^{+} \dot{\mathbf{k}}_{dir} + \alpha_k (\mathbf{I} - \mathbf{J}_k^{+} \mathbf{J}_k) \nabla g_k(\mathbf{k}_q) \tag{2.33}$$

Similar to before, the objective function $g_k(\mathbf{k}_q)$ is defined to maximize stiffness in j different desired directions and avoid the joint-level stiffness limits:

$$g_k(k_{q_i}) = w_{k_0} C_{h,i} + \sum_{m=1}^{j} w_{k_m} C_{k,m} \tag{2.34}$$

where w_{k_0} designates a weight for avoiding joint stiffness limits, $C_{h,i}$ is defined below in Eq. (2.35), and $C_{k,m}$ is defined as in Eq. (2.21) or Eq. (2.23) depending on wether we wish to maximize or specify directional stiffness.

$$C_{h,i} = \frac{k_{q_i,max} - k_{q_i,min}}{(k_{q_i,max} - k_{q_i})(k_{q_i} - k_{q_i,min})} \tag{2.35}$$

The gradient of $C_{h,i}$ is the same as Eq. (2.28), and the gradient of $C_{k,i}$ is trivial. The step size α_k is determined as with α above using Eq. (2.19) to avoid violating the maximum and minimum joint-level stiffness rates of the VSA's.

2.8 Kinematic Simulations

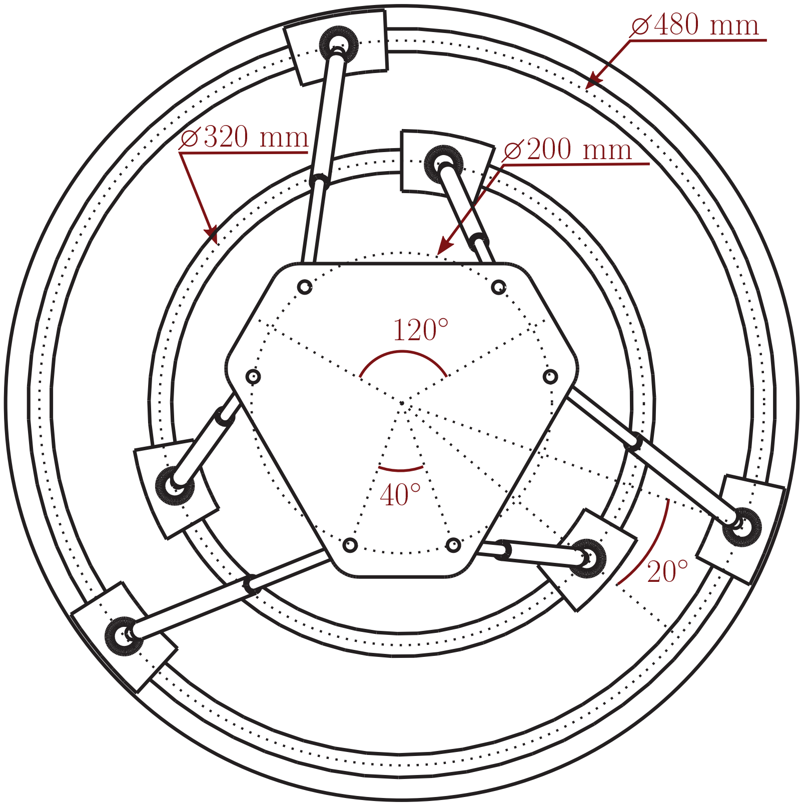


Figure 2.2: Geometry of robot used in kinematic simulations.

The redundancy resolution method described above was validated in a MATLAB kinematic simulation using the exact robot geometry and architecture described above and shown in Fig. 2.2. Additional simulation parameters used were are as follows. The position limits of the prismatic joints were 100-1000 mm, while the joint limits of the base anchor points were updated each time step to keep the arc-length distance between neighboring carriages on the same track greater than 50 mm. The maximum joint speed was 200 mm/s for the prismatic joints and 45°/s for the revolute joints. The VSA stiffness range[6] was 1000-3000 N/m and the maximum $\dot{\mathbf{k}}_{dir}$ was 500 N/m/s. The distance d_{min} was set to 20 mm. The time step used was 0.001 seconds. In all simulations we assumed the moving platform was loaded by a 2.0 kg weight, i.e. $\mathbf{w}_e = [0, 0, 19.62, 0, 0, 0]^{\mathrm{T}}$. The revolute joint-level stiffness was a constant 10^9 Nm/rad.

For all simulations, we confirmed that the base anchor points did not collide, leg collisions did not occur, and the limits on joint position, speed, stiffness, and stiffness rates were not violated.

2.8.1 Example 1: Specified Stiffness & Path Following

This example is illustrated in Parts 1, 2, 3, and 4 of the multimedia extension. The robot follows a spiral trajectory (shown in Fig. 2.3a) while keeping its moving platform parallel its base. The spiral path was described by:

$$\mathbf{o}_p = 5\theta \left[\cos(\theta), \sin(\theta), \frac{250}{5\theta} \right]^{\mathrm{T}}, \ \theta \in [0, 6\pi] \tag{2.36}$$

where $\mathbf{o}_p$ is in units of mm. First, the spiral trajectory was followed while keeping the base anchor points fixed (like a typical Stewart-Gough platform). Fig. 2.3(a) shows the directional stiffness in translation along the axes of the moving frame for this simulation. Note the directional stiffness values in the x and y directions vary widely throughout the motion.

[6] The stiffness range of VSA's in the literature vary widely depending on the application, but a low passive stiffness such as this is typical [103, 104].

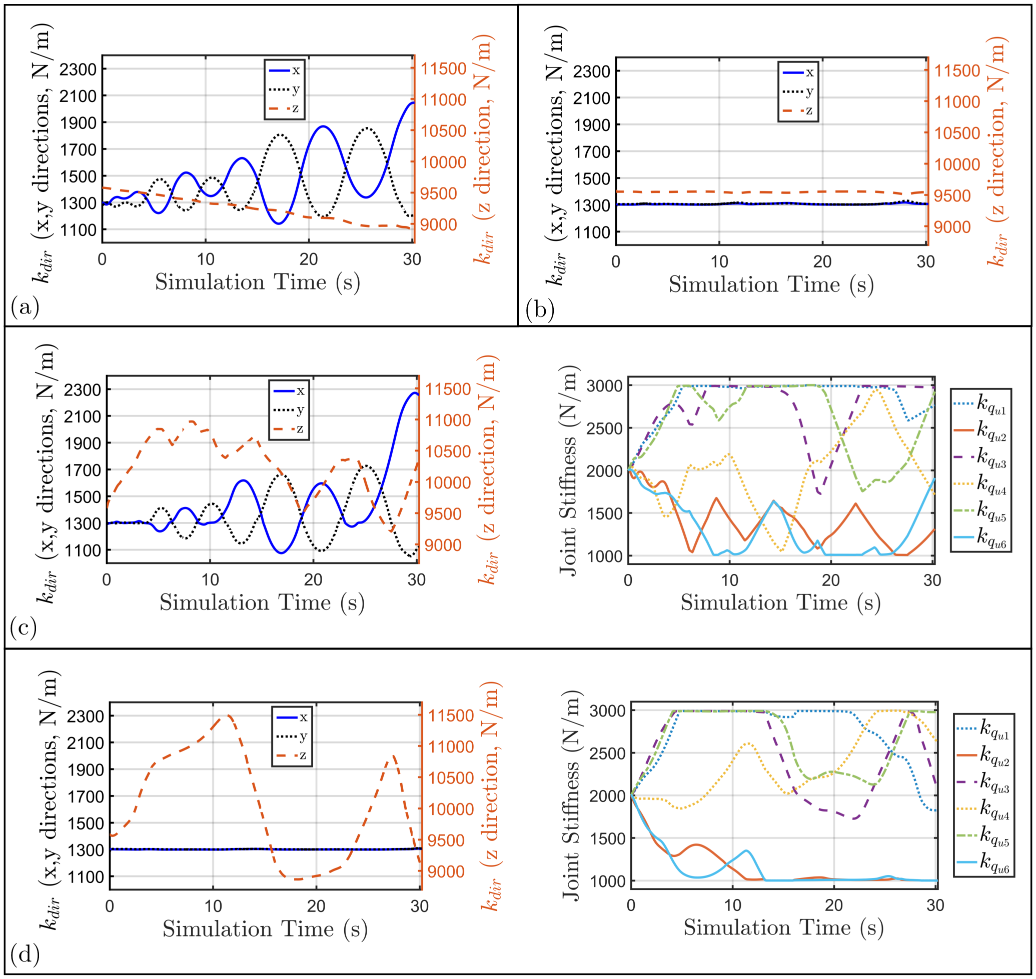


Figure 2.3: Directional stiffness along x, y, and z directions while following the spiral in Eq. (2.36). (a) fixed anchor points with constant joint stiffness, (b) using kinematic redundancy with constant joint stiffness, (c) fixed anchor points with variable joint stiffness, and (d) using both kinematic redundancy and variable joint stiffness.

2.8.1.1 Results Using Variable Stiffness Actuators Only:

The robot was then simulated following the same trajectory with fixed base anchor points but with the joint-level stiffnesses of the prismatic legs actively updated according to Eq. (2.33). The maximum allowable $\dot{\mathbf{k}}_q$ was 500 N/m/s. One $C_{k,m}$ term was included to maximize translational stiffness in the z direction. The weights used were $w_{k_0} = 1$ and $w_{k_1} = 100$. Fig. 2.3(b) shows the directional stiffness values for this simulation and the resulting joint-level stiffness values. The RMS and maximum stiffness errors are given in Table 2.1. Although the directional stiffnesses were closer to their desired values for parts of the trajectory, the ability of these particular simulated VSAs to achieve the desired stiffness for this particular example was limited by the joint-stiffness range and maximum joint-stiffness rate $\dot{\mathbf{k}}_q$.

Table 2.1: Stiffness modulation errors while following the spiral of Eq. (2.36) utilizing VSAs with fixed base anchor points.

	x direction	y direction
Error [RMS, max.]	[295.53, 972.13]	[183.67, 426.21]
% Error [RMS, max.]	[22.73%, 74.78%]	[14.13%, 32.79%]

Table 2.2: Stiffness modulation errors while following the spiral of Eq. (2.36) utilizing kinematic redundancy with constant joint stiffness

	x direction	y direction
Error [RMS, max.]	[6.49, 17.24]	[9.19, 29.70]
% Error [RMS, max.]	[0.50%, 1.33%]	[0.70%, 2.29%]

Table 2.3: Stiffness modulation errors while following the spiral of Eq. (2.36) utilizing both kinematic redundancy and VSAs

	x direction	y direction
Error [RMS, max.]	[1.54, 7.06]	[1.49, 19.37]
% Error [RMS, max.]	[0.12%, 0.54%]	[0.16%, 1.49%]

2.8.1.2 Results Using Kinematic Redundancy Only:

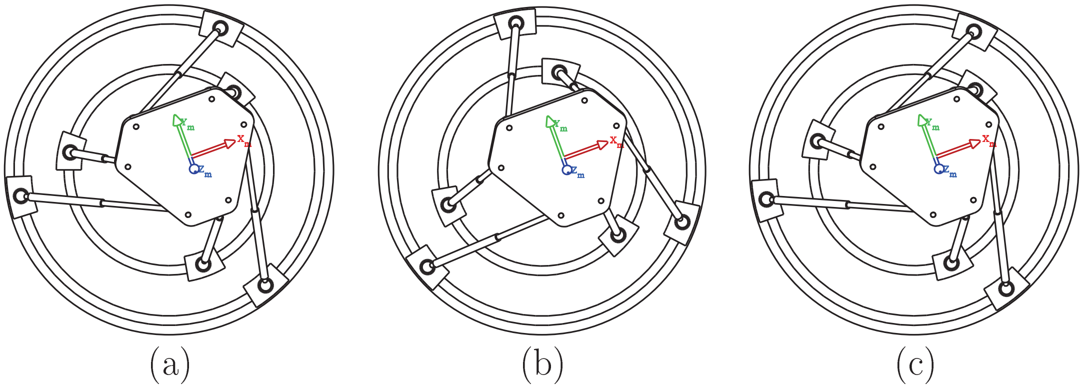
Next, the robot was simulated following the same trajectory with movable base anchor points. The objective function in Eq. (2.20) was used for redundancy resolution, with two C_k terms to keep the x and y directional stiffness constant at 1300 N/m, and a third C_k term to maximize stiffness in the z direction. The weights used in Eq. (2.20) were $w_1 = 1$, $w_2 = 5$, $w_3 = 10$, $w_4 = w_5 = 10$, and $w_6 = 5$. The VSA's in the prismatic legs were kept at a constant value of 2000 N/m. Fig. 2.3(b) shows the directional stiffness values for this simulation. Note that the x and y directional stiffness values are close to their desired values (1300 N/m). The z-direction stiffness stayed fairly constant, in contrast to the first simulation where the stiffness decreased. The RMS and maximum stiffness errors are given in Table 2.2. For this particular example, kinematic redundancy alone was more effective than VSAs alone in achieving the desired directional stiffnesses.

2.8.1.3 Results Using VSAs and Kinematic Redundancy:

The same simulation was run utilizing both kinematic redundancy and variable joint-level stiffness using the same parameters as the two examples above. Fig. 2.3(d) shows the directional stiffness values and the resulting joint-level stiffness values for this simulation. The RMS and maximum stiffness errors are given in Table 2.3. Compared to the kinematic-redundancy-only and the VSA-only cases, using combined kinematic redundancy and VSA's allowed for the desired directional stiffness to be achieved with less error throughout the trajectory while also increasing the z-direction stiffness in some parts of the trajectory. Note that many of the VSA's were saturated for at their extreme stiffness values throughout the workspace, which is primarily due to the $C_{k,m}$ term that attempts to maximize the z-direction stiffness.

2.8.2 Example 2: Satisfying a Spatial Stiffness

In this example, we show how our proposed redundancy resolution scheme can be used to specify a spatial stiffness matrix for a particular moving platform pose. Part 5 of the multimedia extension shows one such simulation. We first found a physically realizable spatial stiffness by finding the robot's stiffness matrix after perturbing all base anchor points by 30° (as shown in Fig. 2.4a) and the joint-level stiffnesses by 200 N/m from their nominal values, i.e. $\mathbf{q}_b = [50°, 70°, 170°, 190°, 290°, 310°]^{\mathrm{T}}$, $\mathbf{k}_{q_u} = [1800, 2200, 1800, 2200, 1800, 2200]^{\mathrm{T}}$ N/m. An arbitrary pose was chosen at $\mathbf{o}_p = [30, 30, 250]^{\mathrm{T}}$ mm, ${}^{w}\mathbf{R}_p = \mathbf{R}_z(20°)\mathbf{R}_x(20°)$, where $\mathbf{R}_z$ and $\mathbf{R}_x$ are elementary rotations about the z and x axes. In addition to a 2 kg load, 5 Nm moments were



(a) (b) (c)

Figure 2.4: (a) Configuration in which the desired spatial stiffness was computed, (b) Initial configuration, (c) Final configuration after optimizing the base anchor locations to satisfy the spatial stiffness. Note that the configuration in (c) matches well with its counterpart in (a).

included in the external wrench, i.e. $\mathbf{w}_e = [0, 0, 19.62, 5, 5, 0]^{\mathrm{T}}$. The resulting desired stiffness matrix was:

$$\mathbf{K}_{desired} = \begin{bmatrix} 1975.67 & 235.49 & 770.22 & -200.34 & -26.59 & 49.63 \\ 235.49 & 1942.09 & 858.27 & -49.92 & -207.40 & 48.62 \\ 770.22 & 858.27 & 8144.98 & 107.68 & -29.60 & 407.74 \\ -200.34 & -49.92 & 107.68 & 27.37 & 1.32 & -7.44 \\ -26.59 & -207.40 & -29.60 & 1.32 & 42.41 & -2.16 \\ 49.63 & 48.62 & 407.74 & -2.44 & -7.16 & 31.34 \end{bmatrix}$$

Note that the stiffness matrix has asymmetries due to the external load. The objective function in Eq. (2.20) was defined with 36 C_{k_i} terms corresponding to the 36 elements of $\mathbf{K}_{desired}$, where $\boldsymbol{\beta}_w$, $\boldsymbol{\beta}_x$, and k_{dir} for each C_{k_i} term were chosen to extract each of the 36 elements from $\mathbf{K}$ so that all elements of $\mathbf{K}$ are modulated in a least-squares sense. For improved numerical scaling, the C_{k_i} weights were chosen so that each $k_{dir,des}$ was scaled to a value between 10 and 100 such that the maximum $k_{dir,des}$ was assigned a weight of 10, and the minimum $k_{dir,des}$ was assigned a weight of 100.

The joint-level stiffness rates were updated according to Eq. (2.33). The VSAs were used satisfy the first three diagonal elements $\mathbf{K}_{desired}$, i.e. each element of $\dot{\mathbf{k}}_{dir}$ was associated with the x, y, and z translational stiffnesses.

The robot was first initialized at the pose where the desired stiffness matrix was found, but with the base anchor points at their nominal values (see Fig. 2.4b) and $\mathbf{k}_{q_u} = [2000, 2000, 2000, 2000, 2000, 2000]^{\mathrm{T}}$ Nm. In this configuration, the stiffness error was:

$$\mathbf{K}_{error} = \mathbf{K}_{desired} - \mathbf{K} = \begin{bmatrix} 565.11 & 114.62 & -76.73 & -127.60 & -108.80 & 47.05 \\ 114.62 & 461.50 & -168.20 & 78.29 & -72.55 & 18.54 \\ -76.73 & -168.20 & -1034.71 & -3.01 & -17.98 & 200.15 \\ -127.60 & 78.29 & -3.01 & -6.89 & -0.31 & -0.45 \\ -108.80 & -72.55 & -17.98 & -0.31 & -3.23 & -0.73 \\ 47.05 & 18.54 & 200.15 & -0.45 & -0.73 & 10.55 \end{bmatrix}$$

The pose was then kept constant while the base anchor points and joint-level stiffnesses were allowed to vary according to the redundancy resolution scheme. The final stiffness matrix error after allowing all velocities to settle was:

$$\mathbf{K}_{error} = \begin{bmatrix} 0.50 & 1.19 & -0.80 & -6.41 & 2.41 & -2.38 \\ 1.19 & 0.06 & -0.76 & -2.44 & 5.43 & -6.62 \\ -0.80 & -0.76 & -0.15 & 9.50 & 5.74 & 0.99 \\ -6.41 & -2.44 & 9.50 & 0.19 & -0.07 & 0.57 \\ 2.41 & 5.43 & 5.74 & -0.07 & 0.30 & -0.73 \\ -2.38 & -6.62 & 0.99 & 0.57 & -0.73 & 0.31 \end{bmatrix}$$

The Euclidian norm of the stiffness error vector comprised of the 36 controlled elements was reduced from 1094.0 at the initial configuration to 13.7 for the final configuration. A plot of the Euclidian norm during the simulation is shown in the multimedia extension. Comparing Fig. 2.4a and Fig. 2.4c, we note the redundancy resolution scheme caused the base anchor points to move close to, but not exactly return to, the predefined configuration used for choosing $\mathbf{K}_{des}$.

In general, our redundancy resolution method cannot guarantee that desired stiffnesses will be satisfied due to 1) the possibility of local minima of $g(\mathbf{q})$, which was given in Eq. (2.15), 2) contributions from the singularity avoidance and joint collision avoidance terms in $g(\mathbf{q})$, and 3) potential conflicts with the primary kinematic task. However, as we have demonstrated in the simulation examples, online local redundancy resolution can be used to move the kinematic structure and VSA stiffnesses closer to the desired passive stiffness without having to solve the complete stiffness synthesis problem.

2.9 Conclusions

The primary result of this chapter is a real-time redundancy resolution method for modulating the directional stiffness of a spatial parallel robot that is equipped with both VSAs and kinematic redundancy. Kinematic simulations demonstrated the ability of the method to modulate directional stiffness while avoiding joint limits, singularities, and self-collisions, and that using both VSAs and kinematic redundancy can improve the system's ability to modulate structural stiffness. While it is not guaranteed that the desired stiffness will be exactly satisfied, this redundancy resolution method allows the kinematic redundancy and VSAs to be used to approach the desired stiffness values without finding exact solutions to the stiffness synthesis problem. A topic for future work is incorporating this redundancy resolution scheme into a higher-level planner that generates desired stiffness profiles for a given task. We believe that the results provided in this chapter will support future exploration of applications of parallel robots in manufacturing and human-robot interaction.

CHAPTER 3

DESIGN OF A CONTINUUM ROBOT MODULE FOR AN IN-SITU COLLABORATIVE ROBOT

In this chapter, we describe the design of a tendon-actuated continuum robot module as a first step towards achieving reconfigurable collaborative continuum robots with whole-body sensing for collaborative manufacturing in confined spaces. We describe the mechanical design of the actuation unit as well as the electronics and control software for the module. We begin with a literature overview to motivate the need for this new robot design, and then provide details on the mechanical design and integration into a larger robotic system for the target application of collaborative manufacturing confined spaces.

3.1 Motivation and Prior Work

Collaborative robots are robots intended to operate in close proximity to a human. Traditionally, industrial robots have been completely isolated behind safety barriers, but, as reviewed in [105], recent technological progress and new safety standards have enabled robots to safely operate in close proximity with workers in shared workspaces. The number of collaborative robot installations has grown in recent years, but collaborative robots still accounted for only 4.8% of all industrial robot installations in 2019, indicating that this category is still in its infancy [106]. Potential benefits of collaborative robots include reduced physiological burdens for workers and efficiency through an expanded range of tasks that can be automated, increased flexibility, and reduced setup costs. Reducing long-term physiological burden on workers in particular has the potential to make an impact. In 2018, musculoskeletal disorders due to physiological strain contributed to 30% of lost work days in the United States [107].

In this dissertation, we focus on manufacturing and assembly tasks in confined spaces, e.g. inspection/cleaning of tanks and maintenance/assembly of aircraft, which put a physiological burden on workers due to unergonomic postures and are particularly dangerous because of difficulty in accessing the confined space. A collaborative robot could reduce these risks for workers by either working collaboratively with the worker in the confined space to reduce the burden on the worker or by completely removing the worker from the confined space via teleoperation.

There are several properties of continuum robots that make them promising candidates for collaborative manufacturing, especially in the context of manufacturing in confined spaces. First, their flexible structures allow them to passively and locally comply to loads anywhere along their body, making them potentially more robust to unexpected contact. With the appropriate sensing modalities, this continuous compliance could also provide improved contact detection and isolation, possibly using methods from the literature on rigid-link robots [108].

Second, continuum robots, due to their continuously smooth, compliant structures, can brace anywhere along their body. Bracing against the environment could allow a robot to reduce the joint torques needed to support is own weight [109], which would allow it to reach deeper into confined spaces and improve its ability to provide forces at the end effector. A robot could also use bracing to modulate kinematic conditioning [109], natural frequency [110], and passive compliance [17] online to improve performance for a given task. Although these behaviors could be implemented on a rigid-link robot, continuum robots provide more flexibility in choosing bracing locations anywhere along the body. Bracing forces generated via interaction with the environment could also be used by a continuum robot to change its shape [111, 112], which could provide an even better ability to reach into a confined space and modify its properties to improve performance. Rigid-link robots, even if they are bracing against the environment, have a limited ability change their shapes in this way.

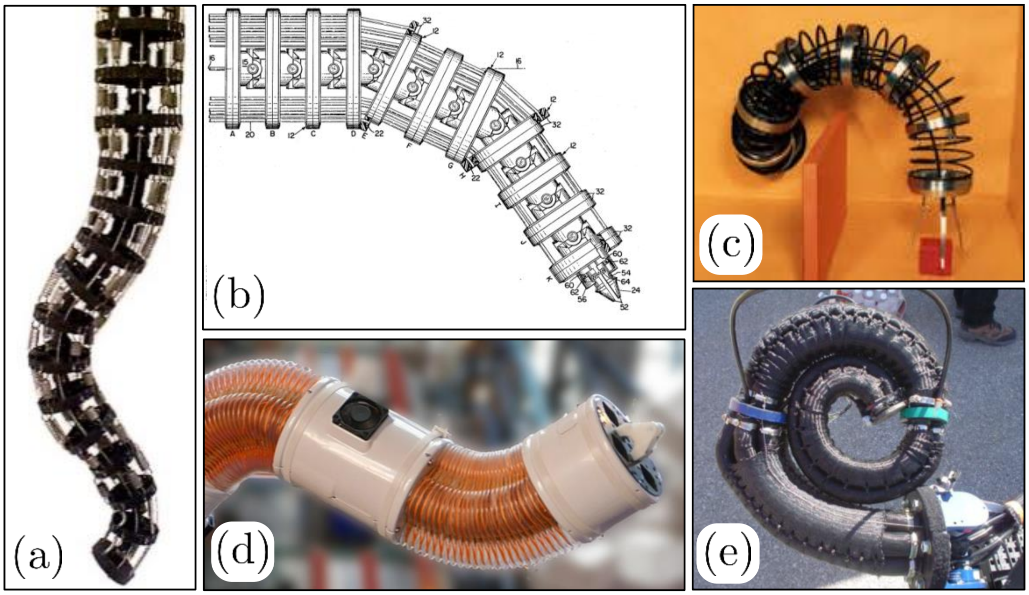


Figure 3.1: Example continuum robot designs with (a-b) discrete backbones [1, 2], (c) spring backbones [3], (d) pneumatic bellows [4] (reprinted with permission from Springer Nature, © 2019 Springer Nature), and (e) McKibben actuators [5] © 2006 IEEE.

These potential benefits make continuum robots a promising design option for collaborative manufacturing confined spaces. Several large-scale continuum and soft robots have been presented in prior work. These designs include tendon-actuated robots with discrete joints [2, 113, 1], tendon-actuated origami-structure manipulators [114], pneumatic bellows robots, [115, 116, 117, 4], robots utilizing McKibben muscle actuators [5, 118], tendon-actuated bellows [119], tendon-actuated springs [3], and inflatable robots [120, 121]. Some of these design incorporate modular actuation unit designs [122, 115] and utilize torsionally stiff backbones [2, 1, 114, 123], but there is still a challenge in designing continuum robots that combine whole-body sensing, assembly modularity, and high torsional stiffness to improve their load-carrying capacity.

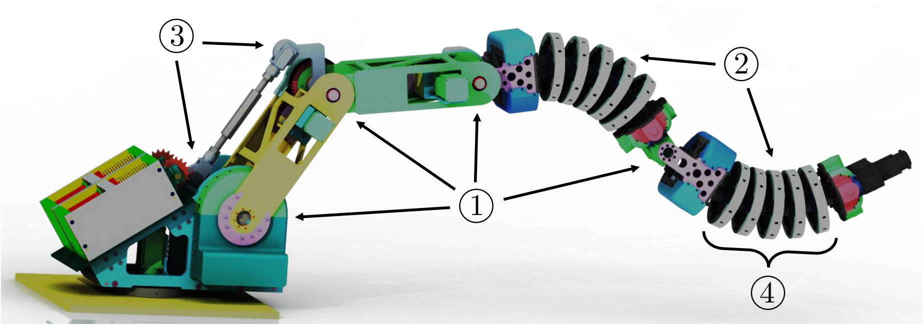


Figure 3.2: Overview of our ISCR concept, showing ① revolute joints, ② continuum segment modules, ③ the static-balancing mechanism, and ④ the multi-modal sensing disks on each continuum segment.

Figures 3.2 and 3.3 show our concept for an *in-situ collaborative robot* (ISCR) for collaborative manufacturing in confined spaces, which we believe will improve on these limitations. The manipulator consists of a traditional serial structure at its base, which is statically balanced with springs to reduce the required torques. The distal portion of the manipulator consists of both traditional revolute joints and tendon-actuated continuum segments. This kinematic structure combines the load-carrying capacity of a rigid-link robot with the potential benefits of a continuum robot described above. The actuation units for the continuum modules are integrated into the base of the segment for improved modularity, and sensing disks in the continuum manipulator provide multi-modal sensing along its body.

The contribution of this chapter is the mechanical design of the actuation unit and compliant structure of the continuum segments in the distal portion of our unique ISCR concept. We present a modular actuation unit design where the tendon actuation components are integrated into the base of each segment as a first step towards achieving a reconfigurable ISCR system. We do not provide a fully modular electrical interface as done in other work on modular robots [124, 125, 126], but a future design iteration could include these kinds of embedded electronics to provide additional modularity. We also present the mechanical/electronic design, integration, and calibration of four custom string encoders. This continuum robot and its shape

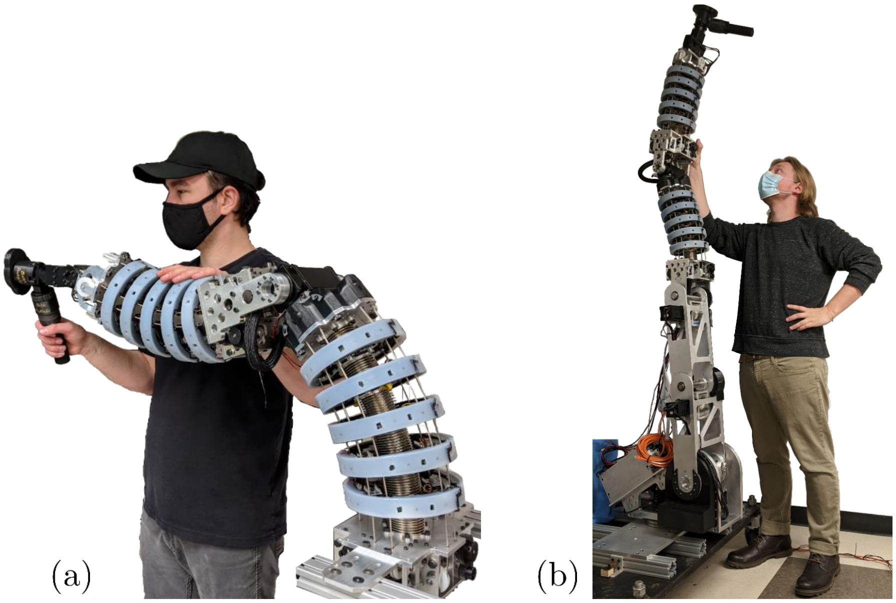


Figure 3.3: (a) Assembly of the distal arm of the robot showing the envisioned collaborative operation of the robot, and (b) assembly of the full robotic system.

sensing capabilities will be used as an experimental test bed for the shape sensing work presented in Chapter 4.

3.2 Continuum Segment Mechanical Design

An overview of the continuum segment assemblies is shown in Fig. 3.4. We will describe each of the four subassemblies, which are the continuum structure, the sensor disk assembly, the distal endplate assembly, and the actuation unit.

The continuum structure consists of five aluminum intermediate disks, which are connected to each other with off-the-shelf electrodeposited nickel bellows that are designed for use as flexible shaft couplings. These bellows are lightweight (33 g each) and flexible in bending but provide a high torsional stiffness (∼63 Nm/°), making the bellows approximately 1950 times stiffer in torsion than in bending. For the target application, the segments are expected to experience static torsional loads of up to 64 Nm, resulting in approximately 1° of torsional deflection per bellow.

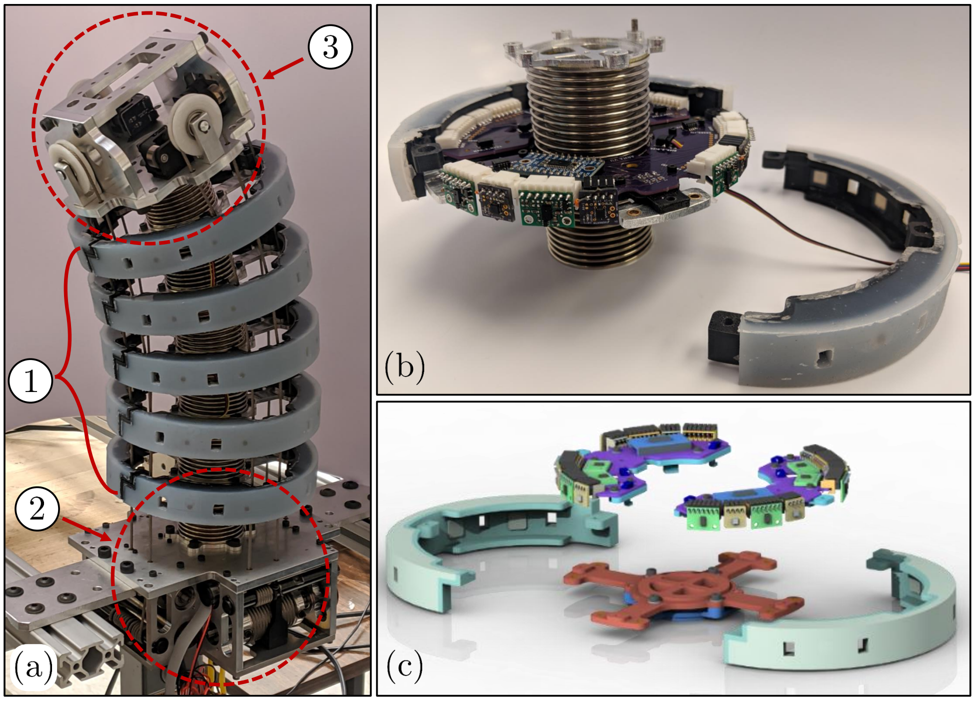


Figure 3.4: Shown here is (a) the continuum segment module with ① the continuum structure with assembled sensor disk, ② the actuation unit, and ③ the distal endplate assembly, (b) a sensor disk with the outer cover removed, and (c) an exploded view of the sensor disk.

The high torsional stiffness of the bellows increases the load-carrying capacity of the robot. Torsional stiffness is particularly important for tendon-actuated continuum robots since most are designed with constant pitch-radius tendons which cannot easily counteract torsional deflections. For comparison, to achieve the same torsional stiffness as the metal bellows using a solid Nitinol rod, the rod would need to be ∼34 mm in diameter. Compared to the more common approaches to constructing continuum robot backbones, these metal bellows provide substantially higher torsional rigidity while still having low bending stiffness. These off-the-shelf metal bellows also have a higher torsional-to-bending stiffness ratio (1950:1) than the 3D printed torsionally stiff origami-inspired segments in [123], where the maximum reported torsional-to-bending stiffness ratio was 293:1, as well as the 3D printed bellows in [127], where a maximum reported torsional-to-bending stiffness ratio of 63:1.

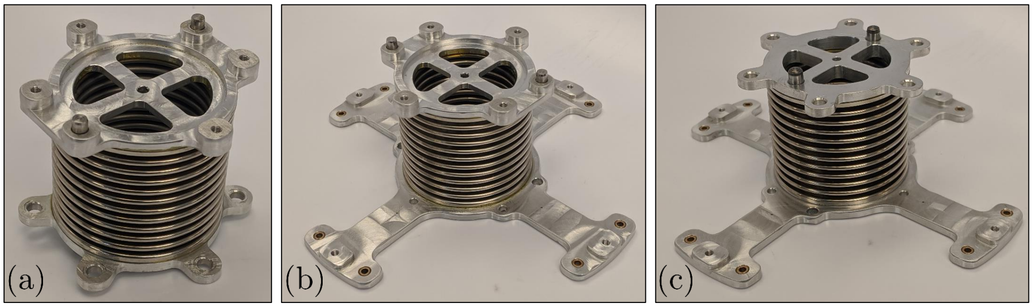


Figure 3.5: Counting from the base of the segment, the continuum structure is built from one subassembly (a), four subassemblies (b), and one subassembly (c).

We use an acrylic structural adhesive (Loctite 326) to assemble the bellows to each intermediate disk, as shown in Fig. 3.5. Each continuum structure consists of one of the bellow subassemblies shown in Fig. 3.5), four of the bellow subassemblies shown in Fig. 3.5b, and one of the bellow subassemblies shown in Fig. 3.5c. These bellow subassemblies are assembled to the base plate, each other, and the endplate using screws. A superelastic nickel-titanium (Nitinol) rod is also passed through the center of the structure. For the proximal segment of the ISCR, a 4 mm diameter rod is used for increased stiffness, while a 3 mm diameter rod is used in the distal segment. The length of the segment, measured from the top of the actuation unit to the bottom of the end disk, is 300.65 mm.

The segment is actuated by actuation tendons that pass through bronze bushings in the intermediate disks and are made from ∅2.38 mm steel wire rope. The actuation tendons are actuated in a differential manner by a capstan mounted on a linear ball spline, and this ball spline is actuated by a gearmotor with a 111:1 gear ratio (maxon DCX22L/GPX22HP) through a single spur gear stage with a gear ratio of 1.851. In the distal endplate assembly of the segment, idler pulleys route the tendons back towards the base of the segment, and the tendons are anchored to a manual pretensioning mechanism within the actuation unit. These idler pulleys provide a 2:1 reduction in the tendon force, which reduces the required sizing of the mechanical components in the actuation unit.

Each intermediate disk in the continuum structure also contains the sensor disk assembly, which has eight time-of-flight sensors for proximity detection and eight Hall-effect sensors for contact detection. The sensing electronics are split between two PCBs and mounted on 3D-printed holders to allow the sensor electronics to be disassembled more easily, as shown in Fig. 3.4d. Finally, two 3D printed covers with over-molded silicone encase the intermediate disks and the sensors to protect the sensors and provide a soft outer surface for the robot. Additional details about the sensor disk assembly and design can be found in [128].

The distal endplate assembly, denoted as ③ in Fig. 3.4, contains four idler pulleys that route the actuation tendons back to the base of the robot. There are also four string encoders housed within this assembly. More details on the string encoders are provided below.

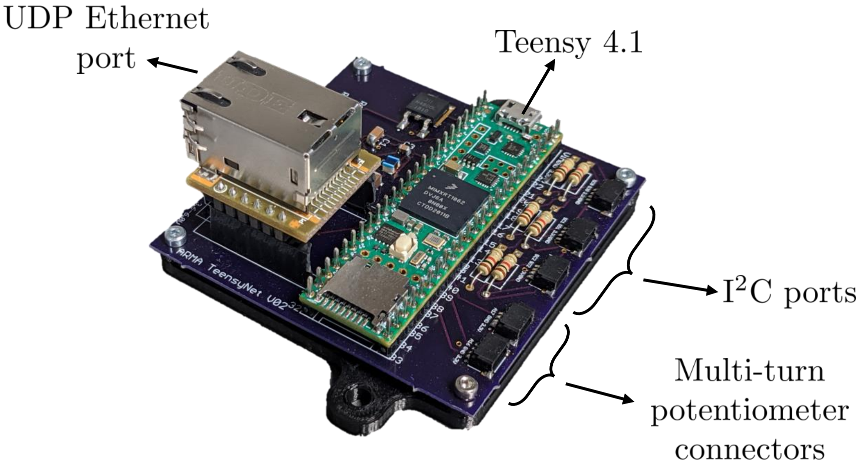


Figure 3.6: A microcontroller mounted at the base of the segment reads the sensor data the I^2C buses and the analog multi-turn potentiometer values and provides the data to the higher-level ROS system via UDP.

All of the sensors in the module, including the time-of-flight sensors, the Hall-effect sensors, and the string encoders, provide an I^2C communication interface. The wires for power and I^2C communication are passed through the center of the continuum segment to a Teensy 4.1 microcontroller at the base of the segment that has three I^{2}C ports. This microcontroller reads the sensor data and provides it to other software components in the system via UDP communication on a local area network. Additional detail on the setup of the I^2C buses and communication can be found in [129].

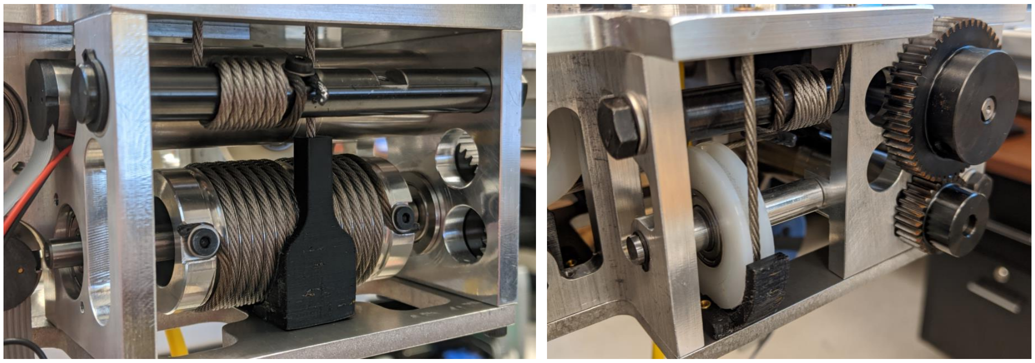


Figure 3.7: Tendon termination points for one of the two DoF in the continuum segment.

The actuation unit, denoted as ② in Fig. 3.4, has two DoF. Each DoF controls the lengths of two tendons that are terminated on the actuating capstan, as shown in Fig. 3.7. We use a 7x49 steel wire rope with a 2.38 mm diameter (Sava Cable) for the actuation tendons. Each tendon has a tensioning shaft that allows the pretension of the tendon to be adjusted using a torque wrench. The tensioning shafts are held in place with two set screws. The path of a tendon begins at one tensioning shaft, passes through a series of oil-embedded sleeve bearings in the intermediate disks up to the distal endplate assembly, wraps over the idler pulley, then passes back through all the intermediate disks. The tendon then wraps around the actuating capstan and terminates at one end of the actuating capstan with a tie-down screw. This tendon path is replicated on the other side of the continuum structure, except an additional idler pulley is need in the base of the segment to route the tendon to the actuating capstan. When the length of one tendon is increased by turning the actuating capstan, the length of the second tendon is reduced by the same amount using this actuation scheme. This entire actuation mechanism and tendon routing scheme is repeated for the other DoF of the robot.

Typical tendon-actuated continuum robots in prior work terminate actuation tendons at the endplate. Here we use an idler pulley in the endplate instead to provide a 2:1 reduction in the tendon force. This allows for a smaller diameter tendon, a smaller actuating capstan diameter, and a smaller gear ratio for the motor gearhead, but comes at a cost of reduced stiffness and increased friction.

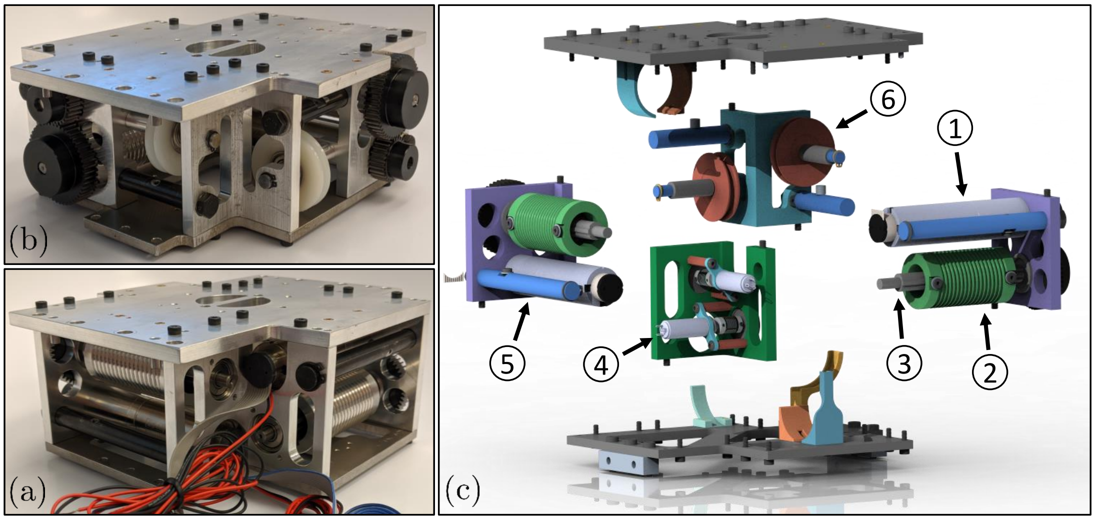


Figure 3.8: The continuum segment actuation unit (a) front view (multi-turn potentiometers not shown), (b) back view, and (c) exploded front view showing the ① gearmotor, ② actuating capstans, ③ ball splines, ④ multi-turn potentiometers, ⑤ tensioning shafts, and ⑥ idler pulleys.

Within the actuation unit, the actuating capstan is driven by a single-stage 1.851:1 spur gear pair, which is then driven by an 18V 20W brushed DC gearmotor (maxon DCX22L/GPX22HP) with a 3-stage planetary gearhead. The proximal segment of the ISCR is assembled with a 111:1 planetary gearhead to provide more torque at lower speed, while the distal segment is assembled with a 62:1 planetary gearhead. The actuating capstan is mounted on a ball spline to allow the capstan to translate to compensate for capstan walk as the tendons wrap and unwrap. The ball spline is supported by two radial ball bearings.

There is also a 10-turn potentiometer mounted at the end of each of the two ball splines to measure rotation of the spline. These multi-turn potentiometers are used only for homing the continuum segment on power-up, and are measured using the microcontroller mounted at the base of the segment through connectors on the board as shown in Fig. 3.6. Once the segment is homed, the incremental encoders on the gearmotors (1024 counts/turn) are used to measure the motor positions. The total mass of one segment, with all sensing electronic and wiring included, was measured to be 4.8 kg. The mass of the actuation unit alone was $\sim$ 2.2 kg.

3.3 Motor Control Hardware and Software

The motor power and encoder cables are routed through the center of each continuum segment and along the rest of the robot to a control box. The control box, shown in Fig. 3.9, contains six motor drivers (maxon ESCON 50/5) that control the motor current, and a PC/104 stack with a 1.91 GHz computer (Diamond Systems Aries) and two DAQ cards with encoder counting chips and digital-to-analog converters (DACs) (Sensoray 526). The box also contains a power supply for the PC/104 stack and for the microcontrollers at the base of each segment.

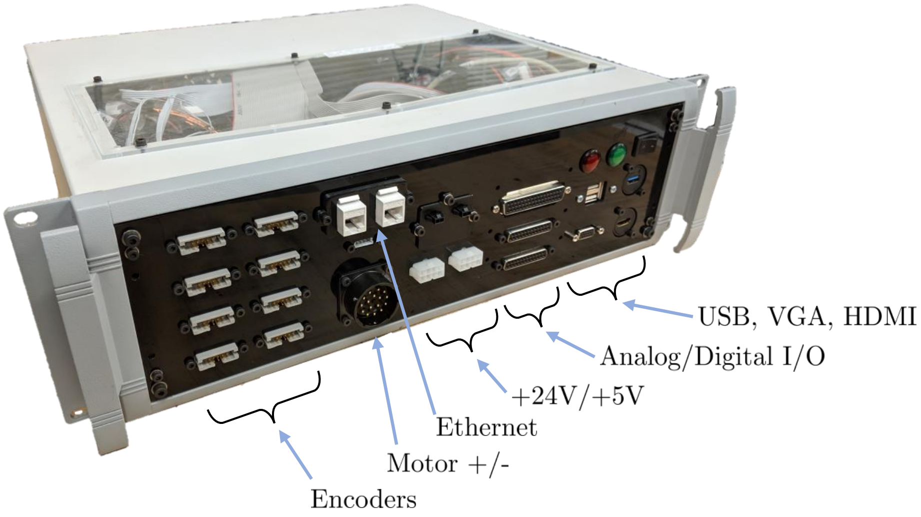


Figure 3.9: The continuum segment motors are controlled using a custom-built PC/104 motor control box.

The control box software runs Ubuntu with a real-time Linux kernel patch (PREEMPT-RT). Two lower priority real-time threads send/receive and process UDP packets, and a third high priority real-time thread manage the control of all motors. For this dissertation, the motors are controlled by a PI velocity controller that runs on this third real-time thread that updates the desired motor current based on the desired position/velocity profile. The UDP interface defines UDP packets that control the state of the control box, change the desired motor current, velocity, or position, and also provides the state of the motors to the higher-level ROS system.

3.4 String Encoder Design and Calibration

Although off-the-shelf string potentiometers and encoders are available, we designed and built a custom string encoder to provide a more convenient mechanical interface for mounting the sensors in the endplate assembly and also provide a digital I^{2}C interface that could be attached to one of the three I^{2}C buses used by the sensor disks. Most off-the-shelf string potentiometers do not provide a digital communication interface and we are not aware of one that provides an I^{2}C interface.

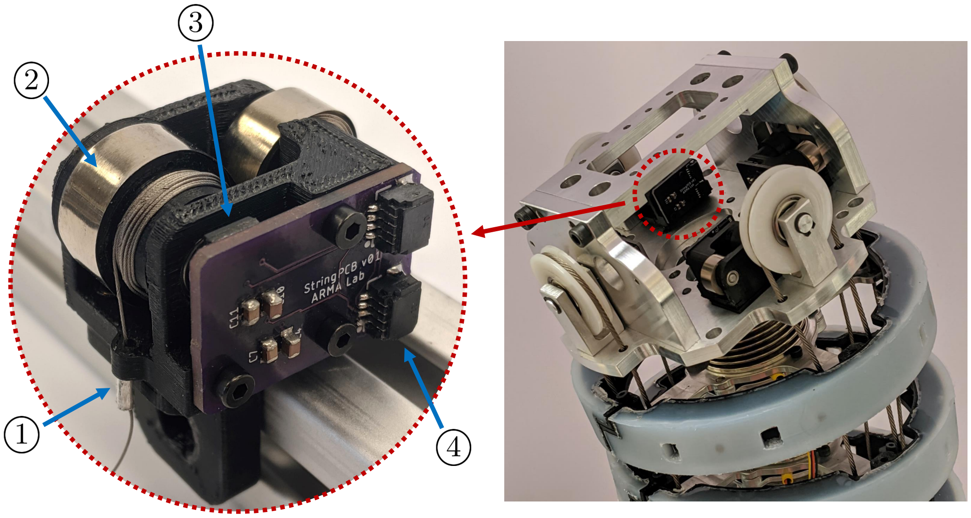


Figure 3.10: The string encoder consists of ① a wire-rope string, ② a constant-torque return spring, ③ a magnetic encoder, and ④ two I^{2}C bus connectors.

The string itself is a 0.33 mm diameter wire rope and is wound around a ∅12 mm output capstan. A constant-torque spring (Vulcan Spring SV3D48) provides 0.02 Nm of torque to the output drum, and when the string is extended the constant-torque spring unwraps from the output drum and wraps onto a storage drum. Set screws are used to secure the output drum to a shaft, but the storage drum rotates freely. The shafts for both drums are supported on each end with radial ball bearings. A magnet is mounted on the end of the output shaft. A custom PCB with a 12 bit magnetic encoder 12 bit (Renishaw AM4096) is mounted on the side of the housing to read the angle of the magnet. Our PCB also contains two connectors for daisy-chained connection of the sensor to the I^{2}C buses. All of these components are housed in a 3D-printed PLA housing. The magnetic encoder angle is read via I^{2}C by a Teensy 4.1 microcontroller that then provides the angle via UDP using a Wiznet WIZ850IO ethernet board to a Robot Operating System (ROS) node that publishes the string extensions on a ROS topic.

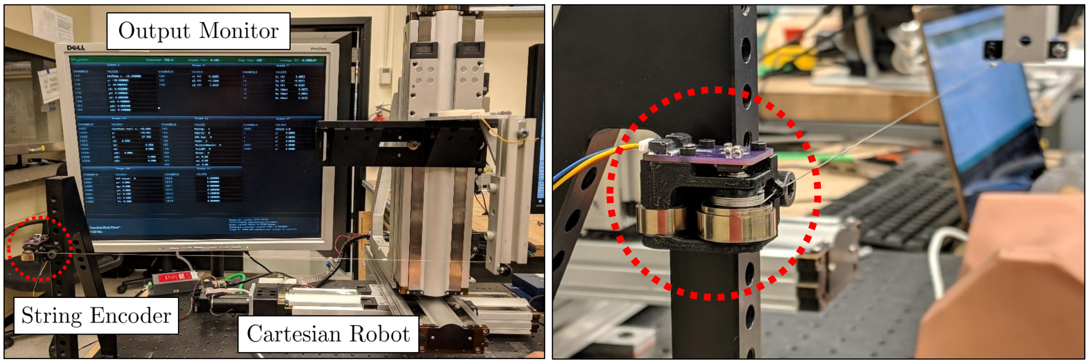


Figure 3.11: Experimental calibration of the string encoders using a Cartesian stage robot.

The radius of the capstan on each string encoder was calibrated by extending the string in 0.2 mm increments using a 3-DoF Cartesian robot, as shown in Fig. 3.11. The linear stages of the Cartesian robot are Parker 404XR ballscrew linear stages actuated with brushed DC motors (Maxon RE35) and equipped with 1000 counts-per-turn encoders. The motion control accuracy of the linear stages was evaluated

Table 3.1: Calibrated Capstan Radius and String Encoder Extension Measurement Error

String Encoder #	r_e (mm)	Avg. error (mm)	Max. error (mm)
1	6.17	0.064	0.24
2	6.18	0.049	0.13
3	6.22	0.087	0.24
4	6.18	0.094	0.27

at ±15 μm. The Cartesian robot was used to extend/release the string over $n_c = 1201$ positions given by $\mathbf{d}_r = [0, 0.2, 0.4, \ldots, 120, 119.8, 119.6, \ldots, 0]^{\mathrm{T}}$ mm. Using the wrapping capstan model while neglecting the helix angle, we obtain the measurement model $\mathbf{d}_r = \boldsymbol{\theta}_e r_e$, where $\boldsymbol{\theta}_e \in \mathrm{I\!R}^{n_c}$ is the vector of magnetic encoder angles and r_e is radius of the encoder capstan. We then solve for the value of r_e that minimizes the least-squares error between the predicted string extension based on string encoder angle and the distance traveled by the Cartesian robot by evaluating $r_e = \boldsymbol{\theta}_e^{+}\mathbf{d}_r$. We calibrated all four of the string encoders and found that, as shown in Table 3.1, the average extension measurement error was below 0.1 mm (0.08% of the total stroke) and the maximum extension measurement error across all four string encoders was 0.27 mm (0.23% of total stroke).

3.5 Conclusions

In this chapter, we have presented the mechanical design of a continuum robot module for an in-situ collaborative robot. We demonstrated that tendon actuation for a collaborative continuum robot can be integrated within the module as a first step towards achieving reconfigurable continuum robots with multiple sensing modalities, i.e. proximity sensing, contact sensing, and shape sensing. The hardware described here was used as an experimental test bed for exploring the shape sensing work in Chapter 4.

CHAPTER 4

LIE GROUP FORMULATION AND SENSITIVITY ANALYSIS FOR SHAPE SENSING OF VARIABLE CURVATURE CONTINUUM ROBOTS WITH GENERAL STRING ENCODER ROUTING

In this chapter, we present a kinematic modeling framework that enables continuum robot shape sensing with general string routing. We first present a literature review to motivate the need for a new modeling formulation. We then present the Lie group kinematic model, explain how to solve the shape sensing problem, and present a methodology for designing the string routing paths to improve then numerical conditioning of the Jacobians related to the shape sensing problem. We validate the approach experimentally for the collaborative continuum segment presented in Chapter 3, and in a simulation for a soft robot, showing that our approach can accurately capture variable curvature deflections of continuum robots utilizing general string encoder routing.

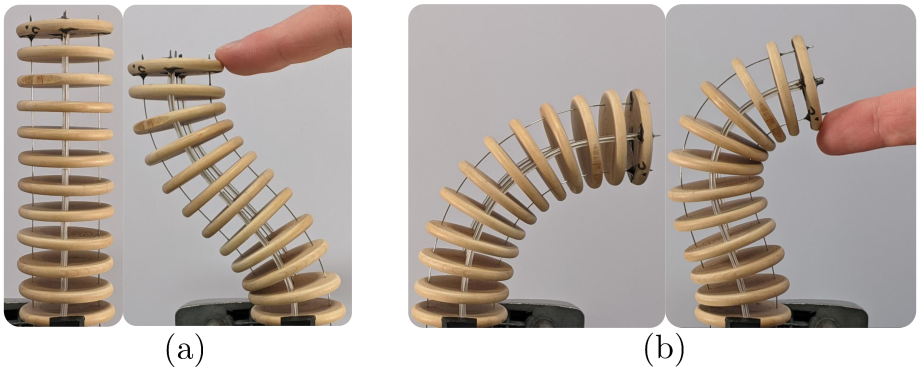


Figure 4.1: A continuum segment subject to passive deflections (a) starting in a straight configuration (b) starting with a bent configuration. This chapter addresses the problem of sensing the deflected shape.

4.1 Motivation and Prior Work

The flexible structure of continuum robots makes them more difficult to model than traditional rigid-link robots. Some modeling uncertainties, like material properties and geometric parameters, can be calibrated offline, but in applications like minimally invasive surgery and collaborative manufacturing, external forces and moments can occur anywhere along the robot's body, and the magnitude of these forces can vary with time. Online sensing is needed to estimate the robot's deflected shape under these external forces.

Prior work on shape sensing can be categorized into *proprioceptive* sensing and *external* sensing. Proprioceptive sensors are located on or within the body of the robot. Examples include optical fibers with fiber Bragg grating sensors [18, 130, 19], infrared reflectance sensors [131], joint-level force sensors [24, 25], load cells at the base of the robot [132], inclinometers [132], polyvinylidene fluoride film [133], and inductance sensing of wrapped wires [4]. In an external sensing approach, sensing components are located in the environment rather than fully on the robot. Examples include stereo-vision tracking of markers [134], computer vision approaches via image segmentation [23], and various intraoperative imaging modalities in medical applications [135]. Although electromagnetic sensors are mounted on a robot [20, 21, 22], this approach requires an external magnetic field generator so it could also be categorized as an external sensing approach. External sensing can simplify the design of the robot itself, but these approaches can be more difficult to use in dynamic changing environments because they require mounting components in the environment. They are also susceptible to visual occlusions and magnetic disturbances in the environment.

In the category of proprioceptive sensing, there are also a number of works that estimate shape by measuring the displacements of passive strings routed within the structure. These types of sensors are often called string potentiometers, because they typically use an analog potentiometer to measure the string displacement, but they have also been called draw-wire encoders, cable encoders, and cable-extension transducers. In all cases, a string is wrapped around a spring-loaded drum and an encoder or potentiometer measures the rotation of the drum as the string wraps and unwraps to provide an estimate of the string displacement. Here we will refer to these sensors as *string encoders*, because in our implementation, we use a custom-built sensor with a magnetic encoder, as described in Chapter 3.

Note that recent soft stretchable sensors would operate in a similar manner to a string encoder if calibrated to measure strain, so the methods herein could potentially be used with soft sensors as well. Examples of soft sensors that could be used in this way include conductive rubber [136], electro-conductive yarn [137], and hydrogels [138].

There are several benefits of string encoders that makes them worthy of study. First, they enable proprioceptive sensing, so they are more suitable for dynamically changing environments. Second, they are relatively low cost compared to load cells and optical fiber sensors. Third, the mechanical and electrical sensing components do not have to be collocated with the string itself, providing additional options for integrating into a system. For example, in the design presented in Chapter 3, the string encoders are located within the distal endplate assembly to allow more room for actuation components in the base and for additional sensing modalities to be integrated into the intermediate disks. In medical applications where miniaturization is needed, string encoders can be mounted in the base of the robot away from the end effectors.

Prior work on shape sensing with string encoders include a passive insertable device for sensing the shape of a lumen [28] and a haptic device for teleoperating a continuum robot [139, 26]. Shape sensing with a mechanics model derived using the principle of virtual power, which includes torsion of the segment and requires estimates of the cable tensions, was presented in [27]. String encoders have also been used to measure the extension/contraction of the pneumatic chambers in soft, pneumatically-actuated robots in conjunction with a variety of models, including in-plane constant-curvature models [140], a planar Cosserat rod model [132], finite-element models [141], rigid-link linkage models [142, 143], neural networks [144], and hybrid approaches [145]. Although these papers present different string arrangements and termination points, none of the above papers considered non-parallel string routing. A closely related modeling work in [146], which provided a Cosserat-rod model with prescribed tendon displacements, does admit non-parallel tendon routings, but this work did not consider passive string encoders and the model requires *a priori* knowledge of the applied forces to compute the shape.

In our modeling approach, we use modal shape functions to describe the backbone curvature as a function of arc length. We then recover the position and orientation along the rod, i.e. the spatial curve, by integrating the curvature functions. Compared to methods in prior work, the benefits of this are simplified analytical expressions that provide design insight and computational efficiency while preserving the ability to describe general spatial deflections.

This approach has not been applied to the problem of shape sensing with string encoders, although similar formulations using polynomials to describe the curvature have been presented for solving continuum robot mechanics models [40, 38, 41] and writing soft robot controllers [147]. Modal shape functions and splines have also been used to describe the spatial curve or the bending angle directly (rather than through a curvature integration step) to solve the inverse kinematics of hyper-redundant robots [148, 149], modeling and calibration of soft cochlear electrode arrays [150], shape sensing with polyvinylidene fluoride film [133], and for modeling continuum robots [151, 33, 31]. This chapter extends these modal shape function modeling approaches to the case of shape sensing with general string routing.

We will show in Chapter 5 how to extend our kinematic formulation to solve Cosserat rod mechanics models, but for the shape sensing methods in this chapter, we intentionally work only with the kinematic equations. The first reason for this is that a mechanics deflection model requires knowledge of the external forces, but the location and magnitude of external forces are not typically known *a priori*. The second reason is that a kinematics-only formulation is more computationally efficient. We can avoid computing a mechanics model, which sometimes requires solving a boundary value problem, and since analytical expressions for the Jacobians and their derivatives are available, the global design measures defined across the entire workspace can be computed efficiently.

Another important consideration for shape sensing is the placement of the sensors. Poorly placed sensors can produce inaccurate shape estimates due to poor kinematic conditioning. A few papers have begun to explore this question in the context of continuum robots [52, 21, 152], but as far as we are aware, no prior work has studied the effect of different string routings and termination points on shape sensing performance. We will show that the analytical expressions from our modeling approach can provide insight into the kinematic conditioning of string encoder routing. We provide an algorithmic approach for designing the string routing to improve the numerical conditioning of the shape sensing Jacobian using a design measure from the literature on robot calibration [153].

In summary, this chapter addresses two of the gaps in the literature by providing 1) a kinematic formulation that handles general deflection shapes and general string encoder routing, and 2) methods for designing the string encoder routing to improve shape sensing performance. We will first describe the kinematic formulation for describing variable curvature backbone shapes and general string routing functions and show how to solve for the shape of the robot given a set of string extension measurements. We then present the noise amplification index as a design measure for the error propagation from noise in the string extension measurements to pose error, and we provide a methodology for designing the string routing to maximize the noise amplification index and therefore prevent ill-conditioning of the shape sensing Jacobians. Finally, we validate the approach on the torsionally-stiff collaborative continuum segment presented in Chapter 3 and in a simulation study on a soft robot with helical routing that is subject to torsional deflections.

4.2 Lie Group Kinematic Formulation

This section presents the kinematic formulation and modal representation for parameterizing the variable curvature backbone shape and deriving the resulting pose of local frames along the continuum segment. This parametrization will be used for describing general string routing and the associated kinematic Jacobians for shape sensing.

4.2.1 Central Backbone Kinematics

Referring to Fig. 4.2, we assume a robot with a total arc length L and define the arc length coordinate s as shown. We also assume that the robot's central backbone has a high slenderness ratio consistent with the assumption of negligible shear strains. With this assumption, the shape of the backbone can be described by its curvature distribution along its arc length s in three directions, $\mathbf{u}(s) = [u_x, u_y, u_z]^{\mathrm{T}} \in \mathrm{I\!R}^3$. For a given location, s, a local frame $\mathbf{T}(s)$ is assigned with its z-axis tangent and pointing in the direction of arc length growth and its two other axes in the backbone's local cross section:

$$\mathbf{T}(s) = \begin{bmatrix} {}^{0}\mathbf{R}_t(s) & {}^{0}\mathbf{p}(s) \\ 0 & 1 \end{bmatrix} \in SE(3), \quad s \in [0, L] \tag{4.1}$$

where ${}^0\mathbf{R}_t(s)$ and ${}^0\mathbf{p}(s)$ are expressed in the world frame $\{0\}$[1]. As the backbone changes its local curvature, this local frame undergoes a twist $\boldsymbol{\eta}(s)$ defined with the angular velocity preceding the linear velocity. The set of local frames associated with local curvatures $\mathbf{u}(s)$ has a corresponding twist distribution $\boldsymbol{\eta}(s)$. Describing the twist $\boldsymbol{\eta}(s)$ in its local frame $\mathbf{T}(s)$, we can write $\boldsymbol{\eta}(s) = [\mathbf{u}(s)^{\mathrm{T}}, \mathbf{e}_3^{\mathrm{T}}]^{\mathrm{T}} \in \mathrm{I\!R}^6$ where $\mathbf{e}_3 = [0, 0, 1]^{\mathrm{T}}$ denotes the local tangent unit vector.

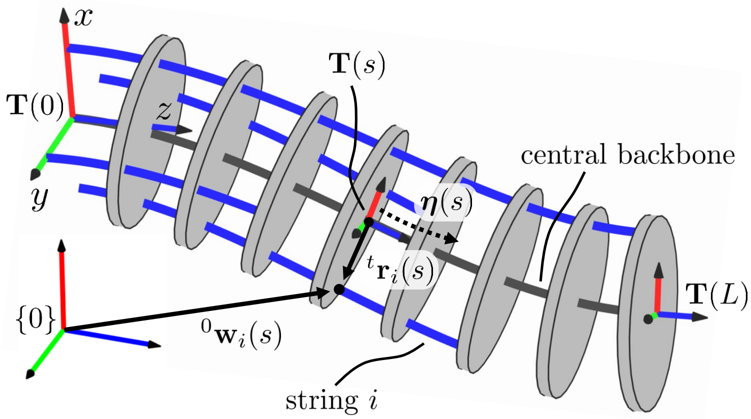


Figure 4.2: Variables used in our kinematic model to describe variable curvature deflections and general string routing.

As the frame $\mathbf{T}(s)$ undergoes the body twist $\boldsymbol{\eta}$, it satisfies the following differential equation [154]:

$$\mathbf{T}'(s) = \mathbf{T}(s)\widehat{\boldsymbol{\eta}}(s), \quad \widehat{\boldsymbol{\eta}}(s) = \begin{bmatrix} \widehat{\mathbf{u}}(s) & \mathbf{e}_3 \\ 0 & 0 \end{bmatrix} \in se(3) \tag{4.2}$$

where the local curvature $\mathbf{u}(s)$ is given by

$$\widehat{\mathbf{u}}(s) = \begin{bmatrix} 0 & -u_z(s) & u_y(s) \\ u_z(s) & 0 & -u_x(s) \\ -u_y(s) & u_x(s) & 0 \end{bmatrix} \in so(3) \tag{4.3}$$

[1] The notation ${}^a\mathbf{y}$ designates vector $\mathbf{y}$ described in frame $\{A\}$ and ${}^a\mathbf{R}_b$ is the orientation of frame $\{B\}$ with respect to frame $\{A\}$

and $(\cdot)'$ denotes the derivative with respect to s and the hat operator $(\widehat{\cdot})$ forms the standard matrix representations of $so(3)$ and $se(3)$ from their vector forms $\mathbf{u}$ and $\boldsymbol{\eta}$, respectively. We also define the adjoint representation of $se(3)$, which will be used below for computing Jacobians:

$$\mathrm{ad}\left(\boldsymbol{\eta}(s)\right) = \begin{bmatrix} \widehat{\mathbf{u}}(s) & \mathbf{0} \\ \widehat{\mathbf{e}}_3 & \widehat{\mathbf{u}}(s) \end{bmatrix} \tag{4.4}$$

In the following analysis, we choose to represent the curvature distribution $\mathbf{u}(s)$ as a weighted sum of polynomial functions (similar to [41]). We denote the polynomial functions as $\boldsymbol{\phi}_x(s)$, $\boldsymbol{\phi}_y(s)$, and $\boldsymbol{\phi}_z(s)$ and the weights as $\mathbf{c}_x$, $\mathbf{c}_y$, and $\mathbf{c}_z$, for the x, y, and z directions, respectively. The curvature distribution then takes the following form:

$$\begin{aligned} \mathbf{u}(s) = \begin{bmatrix} \boldsymbol{\phi}_x^{\mathrm{T}}\mathbf{c}_x \\ \boldsymbol{\phi}_y^{\mathrm{T}}\mathbf{c}_y \\ \boldsymbol{\phi}_z^{\mathrm{T}}\mathbf{c}_z \end{bmatrix} &= \begin{bmatrix} \boldsymbol{\phi}_x^{\mathrm{T}} & 0 & 0 \\ 0 & \boldsymbol{\phi}_y^{\mathrm{T}} & 0 \\ 0 & 0 & \boldsymbol{\phi}_z^{\mathrm{T}} \end{bmatrix} \begin{bmatrix} \mathbf{c}_x \\ \mathbf{c}_y \\ \mathbf{c}_z \end{bmatrix} \\ &= \boldsymbol{\Phi}(s)\mathbf{c}, \quad \boldsymbol{\Phi}(s) \in \mathrm{I\!R}^{3\times m}, \quad \mathbf{c} \in \mathrm{I\!R}^m \end{aligned} \tag{4.5}$$

where the columns of $\boldsymbol{\Phi}(s)$ form a *modal shape basis*, and $\mathbf{c}$ is a vector of constant *modal coefficients*.

The modal description of the curvature distribution offers a description of a variety of variable curvature deflections using a finite set of modal coefficients. This also provides simplified kinematic expressions for computing the workspace, solving for the shape using the string encoder measurements, and designing the string encoder routing to improve the numerical conditioning of the shape sensing problem.

For a given configuration $\mathbf{c}$, the frames $\mathbf{T}(s)$ are found by integrating (4.2). A number of Lie group integration methods could be used for this, as reviewed in [155], but here we use an approach based on the Magnus expansion because both fourth and sixth order expansions can be computed efficiently and because of the

Magnus expansion's large convergence bound [155]. After integration with a Magnus expansion method (or another Lie group integration method), the spatial curve is given as a product of matrix exponentials [156]:

$$\mathbf{T}(s) = \mathbf{T}(0) \prod_{i=0}^{k} e^{\boldsymbol{\Psi}_i}, \quad \boldsymbol{\Psi}_i \in se(3) \tag{4.6}$$

Details on computing $\boldsymbol{\Psi}_i$ can be found in [155, 156]. We will show below that because we use a modal shape basis, Lie group integration is not needed to solve the shape sensing problem (i.e. determining the modal coefficients), however, once the configuration $\mathbf{c}$ is determined, (4.2) must be integrated to compute the robot's forward kinematics and Jacobian, as shown in Section 4.2.6.

4.2.2 Modal Shape Basis with Chebyshev Polynomials

Although any set of modal shape functions could be used to form $\boldsymbol{\Phi}$, (e.g. Euler curves [157, 158], monomials [52], or trigonometric functions), in this chapter we use Chebyshev polynomials of the first kind since each polynomial is bounded by ±1, which provides improved scaling of the modal coefficients and simplifies computation of the admissible workspace, as described in Section 4.3.1. The Chebyshev polynomials can be expressed recursively as [159]:

$$\begin{gathered} T_0 = 1, \quad T_1(x) = x \\ T_n(x) = 2xT_{n-1}(x) - T_{n-2}(x), \quad n = 2, 3, ... \end{gathered} \tag{4.7}$$

where $T_n(x),\ x \in [-1, 1]$ is the n^{th} degree Chebyshev polynomial. Since the arc length coordinate is given by $s \in [0, L]$ and $T_n(x)$ is defined for $x \in [-1, 1]$, the following coordinate transformation is used:

$$x(s) = \frac{2s - L}{L} \tag{4.8}$$

We then evaluate $T_n(x(s))$ via (4.7). In the remainder of this chapter, we refer to the polynomials as simply $T_n(s)$ with the transformation (4.8) implied. The first five Chebyshev polynomials, shifted to $s \in [0, L]$, are shown in Fig. 4.3.

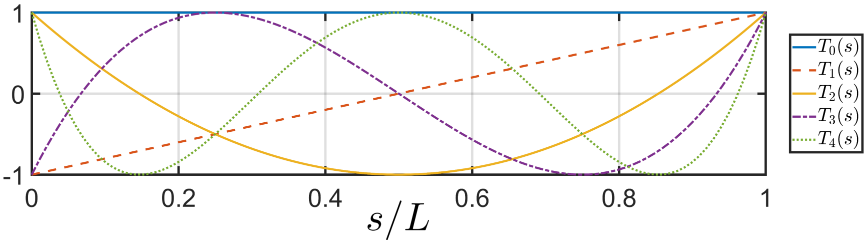


Figure 4.3: The first five shifted Chebyshev polynomials.

As an example, to represent a y direction curvature with a second-order Chebyshev series, the modal shape basis and modal coefficients would be given by:

$$\begin{gathered} u_y(s) = \boldsymbol{\phi}_y(s)\mathbf{c}_y \\ \boldsymbol{\phi}_y^{\mathrm{T}}(s) = \begin{bmatrix} T_0 & T_1(s) & T_2(s) \end{bmatrix}^{\mathrm{T}}, \quad \mathbf{c}_y \in \mathrm{I\!R}^3 \end{gathered} \tag{4.9}$$

where the first three Chebyshev polynomials, shifted to $s \in [0, L]$ are given by:

$$T_0 = 1, \quad T_1(s) = \frac{2s - L}{L}, \quad T_2(s) = \frac{8s^2}{L^2} - \frac{8s}{L} + 1 \tag{4.10}$$

Figure 4.4 illustrates shapes generated by these first three Chebyshev polynomials for a planar continuum segment. Of note is that shapes in the T_0 and T_1 directions of the modal shape basis correspond to experimentally observed deflection shapes in tendon-actuated and multi-backbone robots. Shapes along the T_0 direction correspond to constant curvature deflections exhibited in continuum segments with actuation wires equidistantly distributed about the central backbone. Shapes along the T_1 direction correspond to constant-orientation deflections due to external forces applied to the end disk of a tendon-actuated continuum segment, as shown in Fig. 4.1.

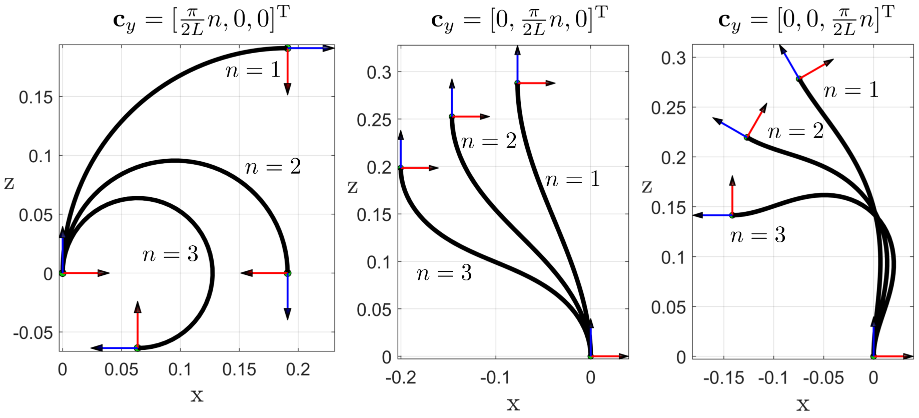


Figure 4.4: Deflections generated on a rod with $L = 300$ mm and second order Chebshev series modal shape basis on the y direction curvature, as given by (4.9). Deflections were generated by taking $\frac{\pi}{2L}$ steps in each of the three modal coefficient directions.

The shape sensing methods presented below would apply to other choices for $\mathbf{\Phi}$ as well. For example, a modal shape basis with coupling between the modal coefficients could be used, e.g. [41]. If we choose $\mathbf{\Phi}$ as the identity matrix, $\mathbf{c}$ corresponds directly to constant curvatures in the x, y, and z directions (see Fig. 4.4), so the results here also apply to robots modeled with the commonly used constant-curvature assumption [30].

4.2.3 General String Routing Kinematics

In [45], a Cosserat-rod based mechanics model was presented for tendon-actuated robots with general tendon routing. In the kinematics model presented here, we use a similar method for describing the string routing. Assuming p strings, the string path, as shown in Fig. 4.2, is expressed in the moving frame $\mathbf{T}(s)$ and given by:

$$^{t}\mathbf{r}_i(s) = [r_{x_i}(s), r_{y_i}(s), 0]^{\mathrm{T}}, \quad i = 1, 2, \ldots, p \tag{4.11}$$

Our kinematic formulation permits any differentiable function for $\mathbf{r}(s)$, but in Sections 4.5 and 4.6 we consider constant pitch-radius paths and helical paths. The position of a point along the string/wire rope path in the world frame is given as the vectorial sum of the point $\mathbf{p}(s)$ along the central backbone and the radial vector $\mathbf{r}(s)$, defined in the moving frame:

$$^{0}\mathbf{w}_i(s) = {}^{0}\mathbf{p}(s) + {}^{0}\mathbf{R}_t(s)\, {}^{t}\mathbf{r}_i(s) \tag{4.12}$$

Noting that vector norms are invariant under rotations, the length of the i^{th} string is given by:

$$\ell_i = \int_0^{s_{a_i}} \| {}^{t}\mathbf{w}'_i(s) \| \, ds, \quad i = 1, 2, \ldots, p \tag{4.13}$$

where s_{a_i} designates arc length along the central backbone at which the string is anchored to the spacer disk/end disk. Taking the derivative of (4.12) with respect to s, substituting (4.5), and then using ${}^{t}\mathbf{w}'_i(s) = {}^{0}\mathbf{R}_t^{\mathrm{T}}\, {}^{0}\mathbf{w}'_i(s)$ results in:

$${}^{t}\mathbf{w}'_i(s) = \mathbf{e}_3 - {}^{t}\widehat{\mathbf{r}}_i(s)\boldsymbol{\Phi}(s)\mathbf{c} + {}^{t}\mathbf{r}'_i(s), \quad i = 1, 2, \ldots p \tag{4.14}$$

Deriving the above result also requires (4.2), which states that ${}^{0}\mathbf{p}'(s) = {}^{0}\mathbf{R}_t(s)\mathbf{e}_3$ and ${}^{0}\mathbf{R}'_t(s) = {}^{0}\mathbf{R}_t(s)\widehat{\mathbf{u}}(s)$.

Recalling that ${}^{t}\mathbf{r}_i(s)$ is in a moving frame having a body angular velocity $\widehat{\mathbf{u}}(s)$, we can visualize (4.14) as the velocity of a point traversing the string path as the arc length s is increased at a unit speed. This point speed is given as the sum of the induced velocity due to the rotation of the moving frame and the velocity of that point relative to the moving frame due to traversal along the central backbone and the rate of change of the string's radial placement ${}^{t}\mathbf{r}'_i(s)$. Since (4.14) is a function only of the modal basis, the string routing function, and the modal coefficients, we can numerically integrate (4.13) with any quadrature rule (e.g. the trapezoid rule) and avoid the cost of integrating the Lie group differential equation in (4.2).

The string routings ${}^{t}\mathbf{r}_i$, $i = 1 \ldots p$ can be chosen by the designer, but must each satisfy a geometric constraint that the string path in world frame must not have a cusp at any configuration of the continuum segment. To ensure physically realizable string paths, the local tangent to the actuation string must point in the same direction as the local tangent of the central backbone. It is possible for the angular rate of changes u_x and u_y to be large enough to cause ${}^t\mathbf{w}_i'$ to point in the opposite direction of $\mathbf{e}_3$, which causes the string path to change directions. To avoid this scenario, we require that ${}^t\mathbf{w}_i'$ always point in the same direction as $\mathbf{e}_3$. Using (4.14), we obtain the following set of p constraint equations for each string:

$$\left({}^t\mathbf{w}_i'\right)^{\mathrm{T}} \mathbf{e}_3 = r_{y_i}(s)u_x(s) - r_{x_i}(s)u_y(s) + 1 > 0 \tag{4.15}$$

This can be rewritten as constraints on u_x and u_y:

$$u_x(s) \leqslant \frac{r_{x_i}(s)u_y(s) + 1}{r_{y_i}(s)}, \quad u_y(s) \leqslant \frac{r_{y_i}(s)u_x(s) + 1}{r_{x_i}(s)} \tag{4.16}$$

In the coming sections, the string routings will be incorporated into a model that captures their effect on increasing the robustness to noise in string length measurements to changes in the estimated shape of the continuum segment.

4.2.4 Solving for the Modal Coefficients

To solve for the shape of the robot, we concatenate (4.13) for each string together with the string length measurements $\boldsymbol{\ell}^*$:

$$\boldsymbol{\ell}(\mathbf{c}) - \boldsymbol{\ell}^* = 0, \quad \boldsymbol{\ell} \in \mathrm{I\!R}^p, \quad \boldsymbol{\ell}^* \in \mathrm{I\!R}^p \tag{4.17}$$

and solve this system of equations for the modal coefficients $\mathbf{c}$. We then integrate $\mathbf{T}'(s)$ once to find the backbone pose at any desired arc length. In some cases, for particular choices of ${}^t\mathbf{r}_i$ and $\mathbf{\Phi}$, unique closed-form solutions to (4.17) can be found by explicit integration of (4.13). This occurs below for the planar case and for the case of robots with high torsional stiffness.

In general however, the system of equations in (4.17) can be nonlinear in $\mathbf{c}$ and may have multiple solutions, so it must be solved by an iterative numerical method, e.g. Gauss-Newton. A necessary condition for using the Gauss-Newton algorithm is $p \geqslant m$ where p is the number of strings and m is the number of columns in $\mathbf{\Phi}$. The Jacobian needed for each iteration of the Gauss-Newton method is provided in the next section.

4.2.5 Configuration Space Jacobian

Since the modal vector $\mathbf{c}$ uniquely defines the shape of the continuum segment for a given modal basis $\mathbf{\Phi}$, we use $\mathbf{c}$ as the configuration space variable. We also define the *configuration-space Jacobian* as the Jacobian relating small changes in the string lengths to small changes in the modal coefficients:

$$d\boldsymbol{\ell} = \mathbf{J}_{\ell c} d\mathbf{c}, \quad \mathbf{J}_{\ell c} \in \mathrm{IR}^{p \times m} \tag{4.18}$$

The i^{th} row of $\mathbf{J}_{\ell c}$ can be found by definition as:

$$\frac{d\ell_i}{d\mathbf{c}} = \int_0^{s_{a_i}} \left({}^t\mathbf{r}_i \times \frac{({}^t\mathbf{w}_i')}{\|{}^t\mathbf{w}_i'\|} \right)^{\mathrm{T}} \mathbf{\Phi} \; ds \tag{4.19}$$

where we have dropped the dependence on s in the integrand terms for brevity and ${}^t\mathbf{w}_i'$ was given in (4.14). As with the integral in (4.13), (4.19) can be computed with any quadrature rule. A similar expression to (4.19) is given in [42].

4.2.6 Body Jacobian

Next, we define the *body Jacobian* as the Jacobian relating the twists of the moving frame $\mathbf{T}(s)$ expressed in $\mathbf{T}(s)$ (also called body twists) to small changes in the modal coefficients:

$$\boldsymbol{\xi}(s) = \mathbf{J}_{\xi c}(s) d\mathbf{c}, \quad \boldsymbol{\xi} \in se(3) \tag{4.20}$$

where $\boldsymbol{\xi}(s) = [\boldsymbol{\omega}(s)^{\mathrm{T}}, \mathbf{v}(s)^{\mathrm{T}}]^{\mathrm{T}}$ is the instantaneous twist of $\mathbf{T}(s)$ produced by $d\mathbf{c}$, expressed in the body frame $\mathbf{T}(s)$ and with angular velocity followed by linear velocity. The i^{th} column of $\mathbf{J}_{\xi c}(s)$, denoted by $\mathbf{J}_{\xi c}^{[i]}(s)$ is the twist produced by a small change in the i^{th} element of $\mathbf{c}$, denoted by c_i:

$$\mathbf{J}_{\xi c}^{[i]}(s) = \left(\mathbf{T}^{-1}(s)\frac{\partial}{\partial c_i}\left(\mathbf{T}(s)\right)\right)^{\vee} \in se(3) \tag{4.21}$$

where $(\cdot)^{\vee}$ denotes the inverse operation of $(\hat{\cdot})$. To compute the partial derivatives in **??**, we take the partial derivatives of the product of matrix exponentials directly. These partial derivatives can be expressed in a recursive form:

$$\frac{\partial}{\partial c_i}\mathbf{T}_{j+1} = \left(\frac{\partial}{\partial c_i}\mathbf{T}_j\right){}^{j}\mathbf{T}_{j+1} + \mathbf{T}_j\left(\frac{\partial}{\partial c_i}{}^{j}\mathbf{T}_{j+1}\right) \tag{4.22}$$

We now require the terms $\frac{\partial}{\partial c_i}\left(e^{\boldsymbol{\Psi}_j}\right)$ to be able to compute $\frac{\partial}{\partial c_i}\left(\mathbf{T}(s)\right)$. These derivatives are given by:

$$\frac{\partial}{\partial c_i}\left({}^{j}\mathbf{T}_{j+1}\right) = \frac{\partial}{\partial c_i}e^{\boldsymbol{\Psi}_j} = e^{\boldsymbol{\Psi}_j}\mathrm{dexp}\left(\frac{\partial \boldsymbol{\Psi}_j}{\partial c_i}\right) \tag{4.23}$$

where the dexp operator is defined in [160] as:

$$\mathrm{dexp}\left(\frac{\partial \boldsymbol{\Psi}_j}{\partial c_i}\right) = \left(\sum_{k=0}^{\infty}\frac{(-1)^k}{(k+1)!}\mathrm{ad}^k\left(\boldsymbol{\Psi}_j\right)\right)\left(\frac{\partial \boldsymbol{\Psi}_j}{\partial c_i}\right) \tag{4.24}$$

Closed-form expressions for dexp for the case of $se(3)$ are also available in [161]. The value of $\frac{\partial \boldsymbol{\Psi}_j}{\partial c_i}$ can be derived symbolically and will vary depending on the order of the Magnus expansion used. Alternatively, it can be estimated via finite difference approximation. To compute dexp, the infinite series can be truncated (we have found

5-10 terms to be sufficient for the problems studied here) or closed-form expressions derived for $SE(3)$ can be used [161]. The particular form of the partial derivatives of $\mathbf{\Psi}_i$ depend on the Lie group integration method used. We refer the reader to Chapter 5 for details on the form of $\mathbf{\Psi}_j$ in the case of a Magnus expansion method.

These equations allow us to compute $\mathbf{J}_{\xi c}$ column-by-column. At first glance these equations look expensive to compute, however, the dexp series only needs to be computed once for each $\mathbf{\Psi}_j$, so these can be stored and reused as the columns of $\mathbf{J}_{\xi c}$ are filled. The recursive nature of (4.22) also helps reduce computation cost. In Chapter 5 these equations made up one part of a larger MATLAB code that solves the statics of Cosserat rods at rates of 30-180Hz, depending on the integration step size used. In this chapter, we use a fine discretization with 100 points along the backbone and are able to compute (4.6) at a rate of ~45 Hz. Larger steps can be used in practice for faster computation times [156], but we use a fine discretization here to avoid introducing additional integration error into this study.

The kinematic expressions defined above provide the equations needed for solving the shape sensing problem and computing the forward/inverse kinematics for a particular continuum robot design and string routing function definition. In Section 4.5 we provide experimental validations of this modeling and shape sensing approach.

4.3 String Routing Optimization

A benefit of our proposed kinematic formulation is that in addition to capturing variable curvature deflections, it provides a designer flexibility in the design of the string routing by allowing non-straight string routings. In some cases, there may be practical mechanical integration considerations that would benefit from non-straight routing, e.g. routing strings around proprioceptive sensing electronics embedded in the continuum structure [128] or routing strings in a tapered path for a segment with a decreasing diameter from base to tip [162]. In this section, we provide considerations for choosing the string routing paths to reduce the propagation of string measurement error to pose error while avoiding ill-conditioned Jacobians.

For a general purpose manipulator, the external loading magnitude and location may not be known *a priori*. For this reason, here we propose a Jacobian-based method to optimize the string routing path ${}^{t}\mathbf{r}_i(s)$ without assuming any particular loading conditions. We do this by designing ${}^{t}\mathbf{r}_i(s)$ to improve the numerical conditioning of both the task space and configuration space Jacobians to reduce the upper bound on error propagation from the string length measurements to errors in the spatial curve $\mathbf{T}(s)$. We validate the approach in simulations and experiments in Sections 4.4, 4.5, and 4.6.

We first define the noise amplification index, a design measure used in robot calibration [153]. For an expression $\mathbf{Ax} = \mathbf{b}$ the noise amplification is given by:

$$\aleph\left(\mathbf{A}\right) = \frac{\sigma_{min}^2\left(\mathbf{A}\right)}{\sigma_{max}\left(\mathbf{A}\right)} \tag{4.25}$$

where $\sigma_{min}\left(\mathbf{A}\right)$ and $\sigma_{max}\left(\mathbf{A}\right)$ are the minimum and maximum singular values of $\mathbf{A}$, respectively. As shown in [153], the noise amplification index provides a bound on how errors in $\mathbf{b}$ propagate to errors in $\mathbf{x}$:

$$\|\delta\mathbf{x}\| \leqslant \frac{1}{\aleph\left(\mathbf{A}\right)}\|\delta\mathbf{b}\| \tag{4.26}$$

Within the context of shape sensing, (4.18) is the mapping relating noise in string length measurement $d\boldsymbol{\ell}$ to a change in modal coefficients $d\mathbf{c}$. Also, (4.20) provides the mapping relating twist and changes in $d\mathbf{c}$. We solve (4.18) for $d\mathbf{c}$ and substitute into (4.20), then solve the equation for $d\boldsymbol{\ell}$ to produce:

$$d\boldsymbol{\ell} = \mathbf{J}_{\ell\xi}\boldsymbol{\xi}(s), \quad \mathbf{J}_{\ell\xi} = \left(\mathbf{J}_{\xi c}\mathbf{J}_{\ell c}^{+}\right)^{+} \tag{4.27}$$

Using (4.26), (4.27) has the following noise amplification bound:

$$||\delta\boldsymbol{\xi}(s)|| \leqslant \frac{1}{\aleph\left(\mathbf{J}_{\ell\xi}\right)}||\delta d\boldsymbol{\ell}|| \tag{4.28}$$

Since the pose $\mathbf{T}(s)$ is an integral of the twist, minimizing the effect of string encoder measurement noise on the estimates of the segment shape requires minimizing $||\delta\boldsymbol{\xi}(s)||$, which in turn requires maximizing $\aleph(\mathbf{J}_{\ell\xi})$. The Jacobian $\aleph(\mathbf{J}_{\ell\xi})$ contains linear/angular velocity units, so we multiply the first three rows corresponding to angular velocity by a characteristic length c_ℓ. For our experimental and simulation results, we chose c_ℓ to be the corresponding segment's kinematic radius.

We must also consider the numerical conditioning of the configuration space Jacobian $\mathbf{J}_{\ell c}$, since it is used when iteratively solving (4.17). Since maximizing $\aleph(\mathbf{J}_{\ell\xi})$ does not guarantee a well-conditioned $\mathbf{J}_{\ell c}$, we seek to also prevent $\aleph(\mathbf{J}_{\ell c}) = 0$, which would indicate a singular $\mathbf{J}_{\ell c}$. We define this design problem as a constrained optimization problem:

$$\max_{\mathbf{k}} \ \aleph_g\left(\mathbf{J}_{\ell\xi}\right) \quad \text{s.t.} \ \ \aleph\left(\mathbf{J}_{\ell c}\right) \geqslant \epsilon \tag{4.29}$$

where $\aleph_g$ is the global noise amplification index defined below, $\mathbf{k}$ are a set of string path design parameters, and ϵ is a lower-bound on the noise amplification index. We will provide two examples of defining $\mathbf{k}$ to ensure the string paths are physically realizable in the simulation and experimental studies below. For these two examples, $\mathbf{k}$ are restricted to a set of integers such that (4.29) can be solved by a brute-force search.

We define $\aleph_g(\mathbf{A})$ as the global noise amplification index, which is the average of $\aleph(\mathbf{A})$ over an admissible configuration workspace $\mathcal{C}_a$ denoting all admissible configurations $\mathbf{c}$. This global performance measure can be numerically approximated by sampling $\mathcal{C}_a$ along j sample points $\mathbf{c}_1 \ldots \mathbf{c}_j$ and computing $\aleph(\mathbf{A})$ at each sampled configuration:

$$\aleph_g(\mathbf{A}) \triangleq \frac{\int_{\mathcal{C}_a} \aleph(\mathbf{A})\, d\mathcal{C}_a}{\int_{\mathcal{C}_a} d\mathcal{C}_a} \approx \frac{1}{j} \sum_{i=1}^{j} \aleph(\mathbf{A}_i) \tag{4.30}$$

4.3.1 Defining the admissible workspace

We define the admissible workspace $\mathcal{C}_a$ as the set of all shapes that a segment can achieve:

$$\mathcal{C}_a = \{\mathbf{c} \mid \mathbf{f}(\mathbf{c}) \leqslant \mathbf{0}\} \tag{4.31}$$

where $\mathbf{f}(\mathbf{c})$ is a vector of constraints on the robot's configuration. To compute the admissible workspace, we take samples in the configuration space $\mathbf{c}$ and discard samples that violate the constraints $\mathbf{f}(\mathbf{c})$. The constraints $\mathbf{f}(\mathbf{c})$ need to be defined on a case-by-case basis, but in this chapter we define three constraints that are relevant to the robots considered in our simulation studies and experimental results.

The first set of constraints we consider are maximal strain limits on the continuum structure $\boldsymbol{\epsilon}_{max} = [\epsilon_{max,x}, \epsilon_{max,y}, \epsilon_{max,z}]^{\mathrm{T}}$. Assuming a backbone diameter d_b and a beam model following linear elasticity, the local curvature limits are given by:

$$\max_s \left(u_j(\mathbf{c}, s)\right) - \frac{\epsilon_{max,j}}{(d_b/2)} \leqslant 0, \quad j \in \{x, y, z\} \tag{4.32}$$

where $\max_s$ denotes the maximum over $s \in [0, L]$ for a configuration $\mathbf{c}$.

The second set of constraints we consider are the maximum curvatures to prevent the intermediate disks from colliding. Considering a subsegment of the robot between two intermediate disks with length L_s, and assuming the subsegment is under constant curvature as shown in Fig. 4.5, we have:

$$\tan\left(\frac{\theta_s}{2}\right) = \frac{h_d}{2\left(\rho^* - r_d\right)} \tag{4.33}$$

where θ_s is the angle between the intermediate disks, h_d is the height of each inter-

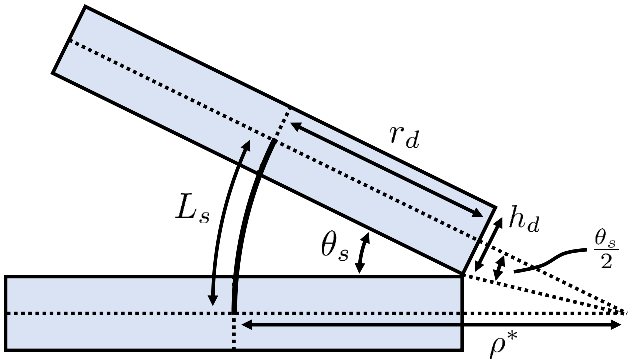


Figure 4.5: Kinematic variables for a subsegment in a constant curvature configuration with disk collision.

mediate disk, r_d is the radius of each disk, and ρ^* is the radius of curvature when the disks collide. Substituting $\theta_s = \frac{L_s}{\rho^*}$ into (4.33), we have the following

$$2\left(\rho^* - r_d\right)\tan\left(\frac{L_s}{2\rho^*}\right) = h_d \tag{4.34}$$

which we solve numerically for ρ^*. To prevent disk collision, we then require that the curvatures in the x and y direction are low enough to avoid this collision condition:

$$\max_s\left(\|\mathbf{u}_{x/y}(s)\|\right) \leqslant \frac{1}{\rho^*} \tag{4.35}$$

where $\mathbf{u}_{x/y}(s) = \left[u_x(s), u_y(s)\right]^{\mathrm{T}}$ and $\max_s$ again denotes the maximum over $s \in [0, L]$.

The third set of constraints we include are given by (4.16), which ensure that the string paths are physically realizable. Using these three sets of constraints (as specified by (4.32), (4.16), and (4.35)), we determine the admissible workspace by searching for configurations that do not violate the constraints, and compute $\aleph_g$ for these admissible configurations. This allows us to explore how to design the string routing paths to maximize $\aleph_g$, which we do below for several simulation and experimental examples.

4.4 Planar Case Study

In this section, we present a simulation case study for a segment subject only to planar deflections. We show that for planar deflections and string routings with constant pitch radius, the configuration space Jacobian $\mathbf{J}_{\ell c}$ is constant for any choice of the modal basis $\mathbf{\Phi}(s)$. We then explore choices of string anchor points and pitch radii that improve the noise amplification indices $\aleph_g\left(\mathbf{J}_{\ell\xi}\right)$ and $\aleph\left(\mathbf{J}_{\ell c}\right)$.

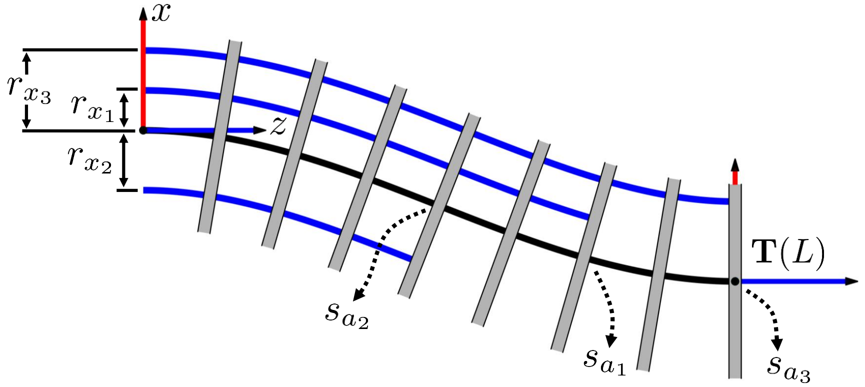


Figure 4.6: Variables of the kinematic model used in the planar case study.

Consider a planar continuum segment that is restricted to deflect in the $x-z$ plane, i.e. $u_x = u_z = 0$, as shown in Fig. 4.6. We choose to represent the curvature distribution using a second-order Chebyshev series as given in (4.9), and we assume three strings are routed within the segment and anchored at arc lengths s_{a_1}, s_{a_2}, and s_{a_3}. We also assume the strings are routed in a path with a constant pitch radius, i.e. ${}^{t}\mathbf{r}_i(s) = [r_{x_i}, 0, 0]^{\mathrm{T}}$, where $r_{x_i} \in \mathrm{I\!R}$ is a constant scalar. Noting that ${}^{t}\mathbf{r}_i'(s) = [0, 0, 0]^{\mathrm{T}}$, (4.14) simplifies to:

$$ {}^{t}\mathbf{w}_i'(s) = \left[0,\ 0,\ \left(1 - r_{x_i}\boldsymbol{\phi}_y^{\mathrm{T}}(s)\mathbf{c}_y\right)\right]^{\mathrm{T}} \tag{4.36}$$

Applying the requirement from (4.15) that $\left({}^{t}\mathbf{w}_i'\right)^{\mathrm{T}}\mathbf{e}_3 > 0$, for this planar case, (4.13) simplifies to:

$$ \ell_i = s_{a_i} - r_{x_i}\int_0^{s_{a_i}} \boldsymbol{\phi}_y^{\mathrm{T}}(s)\mathbf{c}_y\, ds \tag{4.37}$$

We will now consider how many strings are needed to accurately predict the tip pose $\mathbf{T}(L)$ of this planar segment. We assume here that the number of columns in the modal shape basis is equal to the number of strings, i.e. $p = m$ and $\mathbf{J}_{\ell c}$ is square. This means each additional sensing string enables an additional higher-order term to be added to the shape basis to further reduce the tip pose error. Prior works that used shape functions for modeling continuum robots have shown that low-order shape functions can be sufficient for capturing variable curvature deflections [150, 157, 41]. We will also demonstrate this here for this simulation study and experimentally in Sections 4.5 and 4.6.

We used a Cosserat rod mechanics model from [156], which neglected shear strains and extension, to simulate a Nitinol rod with a length $L = 300$ mm and a diameter of 4 mm (similar to the central backbone Nitinol rod used in the robot in Section 4.5). To generate a variety of variable curvature rod shapes, we subjected the rod to planar forces in the world frame's x direction and moments in the world frame's y direction, with the world frame assigned as shown in Fig. 4.6. A subset of these variable curvature shapes is shown in Fig. 4.7. The shape of the rod was obtained as a solution to a boundary value problem using the shooting method for each applied wrench. The maximum applied force and moment were $\mathbf{f}_e = [\pm 60, 0, 0]$ N and $\mathbf{m}_e = [0, \pm 6, 0]^{\mathrm{T}}$ Nm, with 10 wrenches selected between these maximum values, for a total of 100 applied wrenches and rod shapes that were solved for.

For each set of 100 shapes and number of strings considered, we determined the string routing radii and anchor points by solving the following constrained optimization problem:

$$\max_{r_{x_i}, s_{a_i}} \aleph\left(\mathbf{J}_{\ell c}\right) \quad \text{s.t. } 0 \leqslant s_{a_i} \leqslant L \quad i = 2, 3, \ldots, p \tag{4.38}$$

where we have assumed that $r_{x_1} = 0.25$ and $s_{a_1} = L$ to represent an actuation tendon anchored at the end disk. We solved (4.38) using the interior point solver provided by MATLAB's *fmincon()* with an initial guess of evenly spaced radii and anchor points. For each additional string, we used the Cosserat rod model to determine the string

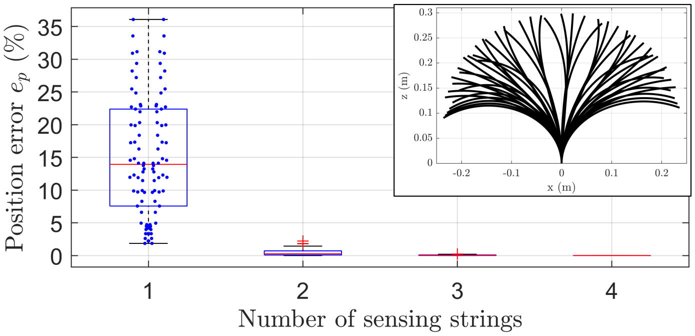


Figure 4.7: The tip position error between our kinematic model and a simulated Nitinol rod rapidly converges as the number of strings p is increased. Individual data points are shown for $p = 1$, and a subset of the 100 simulated variable curvature rod shapes is shown in the inset.

lengths, then used these simulated string lengths to predict the shape and tip pose $\mathbf{T}(L)$ by solving (4.17).

We report the error between the tip pose predicted by the mechanics model and the solution given by solving (4.17) as a percent of the segment length:

$$e_p = \frac{\|\mathbf{p}_s - \mathbf{p}_p\|}{L} \times 100 \tag{4.39}$$

where $\mathbf{p}_s$ and $\mathbf{p}_p$ are the tip positions given by the mechanics model and predicted via (4.17), respectively. The position errors across the 100 shapes and for different numbers of strings are shown in Fig. 4.7. We do not report rotation errors because, as shown in [163], one string anchored to the end disk is sufficient to provide the tip angle of a planar segment and the rotation errors were therefore within numerical precision across all of the simulations.

Fig. 4.7 shows rapid convergence of the tip position error as the number of strings is increased. The average tip position errors across the 100 simulated shapes was 15%, 0.47%, 0.059% and 0.0052% of the segment length for one, two, three and four strings, respectively. We proceed with this case study by choosing $p = 3$.

Concatenating the string lengths for $i = [1, 2, 3]$ results in

$$\boldsymbol{\ell} = \begin{bmatrix} s_{a_1} \\ s_{a_2} \\ s_{a_3} \end{bmatrix} - \underbrace{\begin{bmatrix} r_{x_1} & 0 & 0 \\ 0 & r_{x_2} & 0 \\ 0 & 0 & r_{x_3} \end{bmatrix} \begin{bmatrix} \int_0^{s_{a_1}} \boldsymbol{\phi}_y^{\mathrm{T}}(s)\, ds \\ \int_0^{s_{a_2}} \boldsymbol{\phi}_y^{\mathrm{T}}(s)\, ds \\ \int_0^{s_{a_3}} \boldsymbol{\phi}_y^{\mathrm{T}}(s)\, ds \end{bmatrix}}_{\mathbf{J}_{\ell c}} \mathbf{c}_y \tag{4.40}$$

where we have denoted the Jacobian $\mathbf{J}_{\ell c}$ with an underbrace. For the planar case, $\mathbf{J}_{\ell c}$ is independent of the configuration $\mathbf{c}$ (i.e. it is constant throughout the workspace).

Solving for the modal coefficients $\mathbf{c}_y$ requires inverting $\mathbf{J}_{\ell c}$. This Jacobian is the product of a diagonal matrix whose elements are the pitch radii of each string and a Vandermonde-like matrix of polynomial functions determined by the choice of modal basis. Here we seek to design the string routing paths to improve the numerical conditioning of $\mathbf{J}_{\ell c}$. For a given choice of the modal basis $\boldsymbol{\phi}_y$, the design parameters for each string are 1) the pitch radii r_{x_i}, and 2) the anchor points s_{a_i}.

There are several string routing design insights that can be observed directly from (4.40). First, r_{x_i} must not be zero to avoid rank deficiency. Second, increasing the pitch radius r_{x_i} reduces the sensitivity of $\mathbf{c}_y$ to changes in $\boldsymbol{\ell}$, so increasing r_{x_i} (while keeping the ratios of the pitch radii close to one) will increase $\aleph(\mathbf{J}_{\ell c})$. Third, if any two string anchor points s_{a_i} are equal, $\mathbf{J}_{\ell c}$ will lose rank, so the strings should be anchored at unique s_{a_i} along the segment. Noting that actuation tendons endowed with motor encoder sensing provide the same shape information as a passive string encoder, this means that although actuation tendons are commonly anchored at the end disk to expand the segment's workspace, any additional actuation tendon anchored to the end disk provides no additional shape information (in the planar case). In Section 4.5, we provide conditions under which strings or tendons anchored at the end disk do provide additional shape information for out-of-plane deflections.

We will now consider how to choose s_{a_i} to maximize $\aleph(\mathbf{J}_{\ell c})$ (the noise amplification

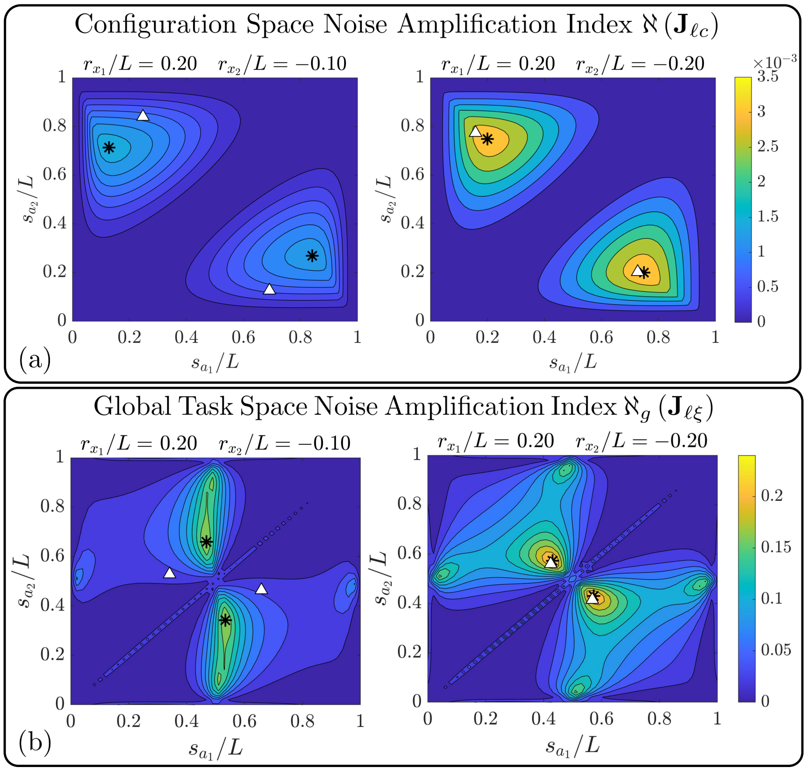


Figure 4.8: The values of a) $\aleph(\mathbf{J}_{\ell c})$ and b) $\aleph(\mathbf{J}_{\ell\xi})$ for different choices of string anchor points ($\frac{s_{a_1}}{L}$ and $\frac{s_{a_2}}{L}$) shown for two sample string radii ($\frac{r_{x_1}}{L}$ and $\frac{r_{x_2}}{L}$) in the planar robot shown in Fig. 4.5. The location of the two peaks on each plot are marked with an asterisk ($*$), and a triangle ($\triangle$) denotes the peak values for the other designs included in Table 4.1 and Table 4.2. The contours for the peaks denoted with triangles are not shown for clarity, but they follow a similar pattern to the ones shown.

index of the configuration space Jacobian) for our planar example. To represent an actuation tendon that is anchored at the end disk, we choose $s_{a_3} = L$. We choose $r_{x_3} = 0.25L$ to correspond to the typical length-diameter ratio of most continuum robots. We now investigate how to optimally choose the radii and anchor points of the two remaining strings.

Figure 4.8a and Table 4.1 show $\aleph(\mathbf{J}_{\ell c})$ for different choices of string radius and anchor point, from which we can make several observations. First, we note that each plot contains two peaks, and that $\aleph(\mathbf{J}_{\ell c}) = 0$ when $s_{a_1} = s_{a_2}$ and when $s_{a_i} = 0$ or $s_{a_i} = L$. Second, we observe that increasing the radius of a string path increases $\aleph(\mathbf{J}_{\ell c})$. Third, we note that increasing r_{x_1} causes s_{a_2} to be slightly reduced at the

Table 4.1: Planar Case: Local Maxima of $\aleph(\mathbf{J}_{\ell c})$ Compared to Equidistant Spacing of String Placement. β Shows the Percent Improvement in $\aleph(\mathbf{J}_{\ell c})$

r_{x_1}/L	r_{x_2}/L	s_{a_1}/L	s_{a_2}/L	$\aleph(\mathbf{J}_{\ell c})$ $(\times 10^{-3})$	β
0.10	-0.10	0.204	0.772	1.03	64%
		0.772	0.204	1.03	64%
0.10	-0.20	0.269	0.841	1.32	54%
		0.714	0.128	1.50	76%
0.20	-0.10	0.128	0.714	1.50	65%
		0.841	0.269	1.32	46%
0.20	-0.20	0.200	0.749	3.29	53%
		0.749	0.200	3.29	53%

peak values of $\aleph(\mathbf{J}_{\ell c})$ (and vice versa for r_{x_2} and s_{a_1}).

Of note is the fact that the intuitive design choice of evenly spacing out the anchor points along $s = [0, L]$ does not result in the best kinematic conditioning for $\mathbf{J}_{\ell c}$. Table 4.1 shows the values of the noise amplification index for each of the peaks in Fig. 4.8 as well as for designs where the string anchor points are evenly spaced, i.e. $s_{a_1} = s_{a_2}$. Table 4.1 also shows the improvement in the noise amplification index over the evenly spaced designs, calculated as:

$$\beta = \left[\left(\aleph\left(\mathbf{J}_{\ell c}\right) - \aleph\left(\tilde{\mathbf{J}}_{\ell c}\right)\right) \Big/ \aleph\left(\tilde{\mathbf{J}}_{\ell c}\right)\right] \times 100 \tag{4.41}$$

where $\tilde{\mathbf{J}}_{\ell c}$ denotes the Jacobians for design having evenly spaced wire anchor points at $s_{a_1} = L/3, s_{a_2} = 2L/3, s_{a_3} = L$. Each one of these Jacobians corresponds with the third (shaded) row in Table 4.1 for each combination of the string radii r_{x_1} and r_{x_2}. As shown in Table 4.1, placing the anchor points at one of the peaks instead of using evenly spaced anchor points resulted in 46% or greater improvement in $\aleph\left(\mathbf{J}_{\ell c}\right)$ for all the cases considered.

We now consider improving the numerical conditioning of the full kinematic mapping (joint to task space) $\aleph_g\left(\mathbf{J}_{\ell\xi}(L)\right)$. We computed $\aleph_g\left(\mathbf{J}_{\ell\xi}(L)\right)$ for this robot using 67 configurations sampled in the admissible workspace, which we defined as any configuration that did not exceed 5% strain for a 4 mm central backbone. Figure 4.8b shows the noise amplification for different string radii and anchor points. We observe that the design parameters that optimize $\aleph_g\left(\mathbf{J}_{\ell\xi}(L)\right)$ are not the same parameters that optimize $\aleph\left(\mathbf{J}_{\ell c}\right)$ in Fig. 4.8a, and in fact there is a conflict between these two design objectives, since the peaks in Fig. 4.8b correspond to valleys in Fig. 4.8a. Since the design objective is usually to minimize the pose error at a particular location, designing for maximizing $\aleph_g\left(\mathbf{J}_{\ell\xi}(L)\right)$ is sufficient as long as the string routing solution avoids singularity of $\mathbf{J}_{\ell c}$.

Table 4.2: Planar Case: Local Maxima of $\aleph_g(\mathbf{J}_{\ell\xi})$ Compared to Equidistant Spacing of String Placement. β Shows the Percent Improvement in $\aleph(\mathbf{J}_{\ell\xi})$

r_{x_1}/L	r_{x_2}/L	s_{a_1}/L	s_{a_2}/L	$\aleph_g(\mathbf{J}_{\ell\xi}(L))$	β
0.10	-0.10	0.573	0.428	0.159	146%
		0.428	0.573	0.159	146%
0.10	-0.20	0.343	0.534	0.181	204%
		0.660	0.468	0.181	204%
0.20	-0.10	0.468	0.660	0.181	213%
		0.534	0.343	0.181	213%
0.20	-0.20	0.573	0.431	0.228	76%
		0.431	0.573	0.228	76%

We also observe again that the intuitive design of using evenly spaced anchor points does not result in an optimally conditioned kinematic mapping. Table 4.2 shows the values of $\aleph(\mathbf{J}_{\ell\xi}(L))$ for the peaks shown in Fig. 4.8b as well as the percent improvement in $\aleph(\mathbf{J}_{\ell\xi}(L))$ as compared to a design using evenly spaced anchor points, i.e. $s_{a_2} = \frac{L}{3}$ and $s_{a_3} = \frac{2L}{3}$. The peak values of $\aleph_g(\mathbf{J}_{\ell\xi}(L))$ were increased by 76% or greater for all cases considered.

In this planar example, we have shown that increasing the number strings reduces the pose estimation error, provided string path design considerations, and showed that the intuitive choice of evenly spaced anchor points does not lead to optimal values for the noise amplification. The optimal placement of string anchor points depends to some degree on the family of deflected shapes that a given robot experiences under target design operating conditions (loading and reach). We have simulated the same case using a monomial basis as opposed to a Chebyshev polynomial basis and noted that the peaks in Fig. 4.8 experience negligible changes. Nevertheless, the designer should carry out a simulation as in Fig. 4.8 to achieve a qualitative understanding of the optimal location of wire anchor points and then determine these points while respecting practical design considerations. Building on these simulation results, we will now demonstrate our modeling and shape sensing approach on two other continuum robot embodiments that are subject to more realistic spatial deflections.

4.5 Robots with High Torsional Stiffness and Constant Pitch String Paths

We now consider robots that are subject to spatial deflections but have sufficiently high torsional stiffness that renders the torsional deflections negligible. We consider this category of robots because a number of continuum robots with high torsional stiffness have been presented in prior work [164, 127, 114, 123], and we presented in Chapter 3 a new modular collaborative continuum robot in this category which we will use to experimentally validate the model we present. This model is also directly applicable to hyper-redundant robots with torsionally stiff universal joint backbones, e.g. [2, 1]. This category of robots is also of interest because, as we will show, if the string paths are restricted to constant pitch radius paths, the configuration space Jacobian is constant and the modal coefficients are linear with respect to the string lengths, which simplifies the solution of the shape sensing problem.

First, we will present the kinematic formulation for this category of robots and provide considerations for designing the string paths. In particular, we show that *two strings anchored to the same disk add distinct information only if their polar coordinates are distinct by angle difference other than 0°and 180°(i.e. they are not* *collinear in the radial direction of the disk).* We then present the results of the string routing optimization for our collaborative continuum segment and experimentally validate the sensing approach, showing that the end disk position can be sensed with position errors below 5% of arc length using information from four passive string encoders and two actuators.

4.5.1 Kinematic model for torsionally stiff continuum robots

Under the assumption of negligible torsional deflections, i.e. $u_z(s) = 0$ the shape basis (4.5) takes the form

$$\mathbf{u}(s) = \begin{bmatrix} \boldsymbol{\phi}_x^{\mathrm{T}} & 0 \\ 0 & \boldsymbol{\phi}_y^{\mathrm{T}} \\ 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{c}_x \\ \mathbf{c}_y \end{bmatrix} = \boldsymbol{\Phi}(s)\mathbf{c} \tag{4.42}$$

We restrict our consideration here to constant pitch-radius string routings give by:

$${}^{t}\mathbf{r}_i(s) = \begin{bmatrix} r_{x_i} & r_{y_i} & 0 \end{bmatrix}^{\mathrm{T}}, \; r_{x_i} \in \mathrm{I\!R}, \; r_{y_i} \in \mathrm{I\!R} \tag{4.43}$$

Noting that ${}^{t}\mathbf{r}_i'(s) = [0, 0, 0]^{\mathrm{T}}$, the string path derivative (4.14) simplifies to:

$${}^{t}\mathbf{w}_i'(s) = \left[0,\, 0,\, \left(r_{y_i}\boldsymbol{\phi}_x^{\mathrm{T}}\mathbf{c}_x - r_{x_i}\boldsymbol{\phi}_y^{\mathrm{T}}\mathbf{c}_y + 1\right)\right]^{\mathrm{T}} \tag{4.44}$$

Applying the requirement from (4.15) that $\left({}^{t}\mathbf{w}_i'\right)^{\mathrm{T}}\mathbf{e}_3 > 0$ and using (4.13) results in the string length as:

$$\ell_i = s_{a_i} + \left(\int_0^{s_{a_i}} \begin{bmatrix} r_{y_i}\boldsymbol{\phi}_x^{\mathrm{T}}, & -r_{x_i}\boldsymbol{\phi}_y^{\mathrm{T}} \end{bmatrix} ds\right)\mathbf{c} \tag{4.45}$$

where $\mathbf{c} = [\mathbf{c}_x^{\mathrm{T}}, \mathbf{c}_y^{\mathrm{T}}]^{\mathrm{T}}$. Concatenating for each string $i \in [1, \ldots, p]$ gives:

$$\boldsymbol{\ell} = \begin{bmatrix} s_{a_1} \\ \vdots \\ s_{a_p} \end{bmatrix} + \underbrace{\begin{bmatrix} \int_0^{s_{a_1}} \begin{bmatrix} r_{y_1}\boldsymbol{\phi}_x^{\mathrm{T}}, & -r_{x_1}\boldsymbol{\phi}_y^{\mathrm{T}} \end{bmatrix} ds \\ \vdots \\ \int_0^{s_{a_p}} \begin{bmatrix} r_{y_p}\boldsymbol{\phi}_x^{\mathrm{T}}, & -r_{x_p}\boldsymbol{\phi}_y^{\mathrm{T}} \end{bmatrix} ds \end{bmatrix}}_{\mathbf{J}_{\ell c}} \mathbf{c} \tag{4.46}$$

As in the planar case, we observe that $\mathbf{J}_{\ell c}$ is independent of the configuration $\mathbf{c}$.

We now consider how to choose the anchor points and string paths to avoid singularities in $\mathbf{J}_{\ell c}$. Consider the scenario where two anchor points are equal, $s_{a_i} = s_{a_j}, i \neq j$. In this scenario, the two rows of $\mathbf{J}_{\ell c}$ corresponding to these strings will be dependent if one is a scalar multiple of the other. Considering a cross section of the segment at $s = s_{a_i} = s_{a_j}$, this will occur if the string radii $[r_{x_i}, r_{y_i}]$ and $[r_{x_j}, r_{y_j}]$ lie on a line passing through the origin of the body frame $\mathbf{T}(s_{a_i}) = \mathbf{T}(s_{a_j})$.

Furthermore, no more than two parallel-routed strings attached to a disk of an inextensible continuum segment are needed. This is apparent since two radially non-collinear strings define the plane of the disk. Therefore, for a group of strings/tendons of dimension n, in order to be able to calculate the modal coefficients $\mathbf{c} \in \mathrm{IR}^n$, no more than two strings can be attached to the same intermediate/end disk (to prevent rank deficiency of $\mathbf{J}_{\ell c}$).

4.5.2 Experimental validation on a collaborative continuum robot module

We now validate our kinematic model the collaborative continuum segment module presented in Chapter 3. Additional detail on the design and calibration of the string encoders can be found in Chapter 3.

Due to space considerations for integrating sensing electronics in the intermediate disks [128], the strings are restricted to be routed in constant pitch radius paths.

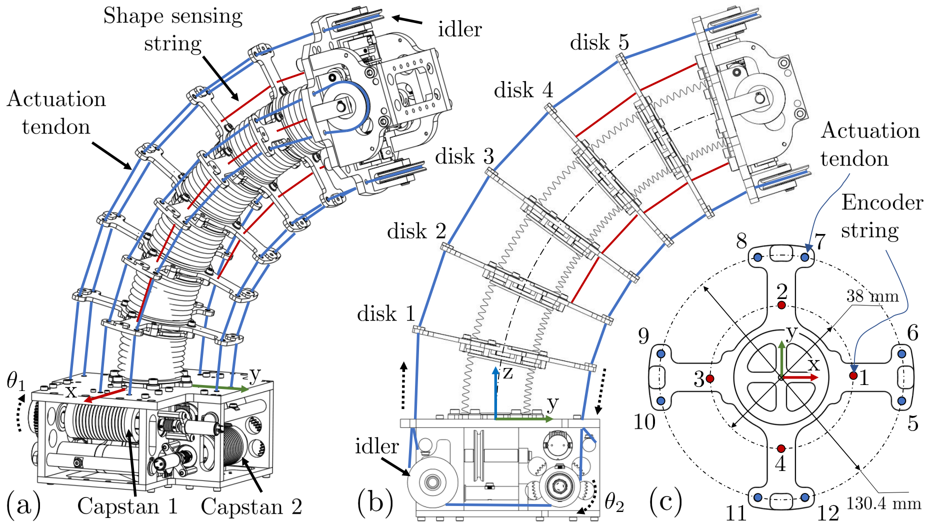


Figure 4.9: The path of the actuation tendons (numbered 5-12) and string encoders (numbered 1-4) are shown in (a) a 3D view, and (b) a simplified side view. Shown in (c) is top view of an intermediate disk with the locations where each string/tendon passes.

The strings paths are therefore given by (4.43), where the values for r_{x_i} and r_{y_i} are determined from the geometry in Fig. 4.9c. Since the string encoders are mounted in the distal endplate (instead of in the robot's base), the string lengths $\ell_i,\ i \in [1, 2, 3, 4]$ are found via (4.13), except integration is performed from s_{a_i} to L rather than from 0 to s_{a_i}, resulting in an expression similar to (4.45).

We also use the actuation tendon lengths $\ell_i,\ i \in [5, \ldots, 12]$ as additional input to the problem of shape estimation. The routing path definitions for the actuation tendons require special consideration because of the idler pulleys in the end plate that reduce the tendon force by rerouting the tendons to the base of the robot. We separate the actuation tendon paths into eight different curves, as shown in Fig. 4.9, and require the difference in the tendon path lengths on each side of the actuation capstan be equal to the change in length due to the actuation capstan rotation:

$$\begin{aligned} \Delta\ell_{\theta_1} &= \Delta\left(\ell_5 + \ell_6\right) = -\Delta\left(\ell_9 + \ell_{10}\right) \\ \Delta\ell_{\theta_2} &= -\Delta\left(\ell_7 + \ell_8\right) = \Delta\left(\ell_{11} + \ell_{12}\right) \end{aligned} \tag{4.47}$$

where $\ell_5 \ldots \ell_{12}$ refer to the actuated tendon lengths corresponding with the numbered bushings shown in Fig. 4.9b and ℓ_{θ_1} and ℓ_{θ_2} are the change in tendon length due to rotation of the capstan for joints 1 and 2, respectively. The values of ℓ_{θ_1} and ℓ_{θ_2} are determined from the motor angles while accounting for the helical wrapping pattern on the capstan:

$$\ell_{\theta_i} = \frac{\theta_i}{2\pi}\sqrt{(2\pi r_c)^2 + \gamma^2}, \quad i \in [1, 2] \tag{4.48}$$

where θ_1 and θ_2 are the angles of the first and second actuation capstans, respectively, r_c is the radius of the capstan, and γ is the lead of the helical groove on the capstan. The actuation tendon lengths $\ell_i,\ i \in [5 \ldots 12]$ are found via (4.45).

Information from the two motor encoders and the four string encoders allows for a modal basis with six columns ($p = 6$). Neglecting torsional deflections, the curvature distribution is given by (4.42) with the following shape functions:

$$\boldsymbol{\phi}_x(s) = \boldsymbol{\phi}_y(s) = \Big[T_0, T_1(s), T_2(s)\Big]^{\mathrm{T}} \tag{4.49}$$

The string encoder length equations for $i \in [1, 2, 3, 4]$ are concatenated together with (4.47) to give the string lengths in a similar form as in (4.46), but accounting for the string encoder routing as described above while adding the tendon lengths:

$$\begin{bmatrix} \ell_1 \\ \ell_2 \\ \ell_3 \\ \ell_4 \\ \ell_{\theta_1} \\ \ell_{\theta_2} \end{bmatrix} = \begin{bmatrix} L - s_{a_1} \\ L - s_{a_2} \\ L - s_{a_3} \\ L - s_{a_4} \\ 0 \\ 0 \end{bmatrix} + \mathbf{J}_{\ell c}\mathbf{c}, \quad \mathbf{J}_{\ell c} \in \mathrm{I\!R}^{6\times 6}, \quad \mathbf{c} \in \mathrm{I\!R}^{6} \tag{4.50}$$

Given a set of string measurements, we then solve (4.50) for the modal coefficients $\mathbf{c}$ through a single matrix inversion. Our MATLAB 2019b implementation computes

$\mathbf{c}$ at a rate of ~125 kHz. As with the planar example above, the configuration space Jacobian $\mathbf{J}_{\ell c}$ is constant when assuming zero torsional deflections, so $\mathbf{J}_{\ell c}$ and $\mathbf{J}_{\ell c}^{-1}$ can be computed once and stored.

Based on the analysis in Section 4.5.1 of singularities in $\mathbf{J}_{\ell c}$, the two actuation tendon equations in (4.47) do not introduce singularities into $\mathbf{J}_{\ell c}$, because the tendon pairs that are collinear with the central backbone point are in the same equations in (4.47). However, any additional string anchored at the end disk (or equivalently at the base disk for the strings mounted in the endplate) would not provide additional shape information. The radius of the passive sensing strings is fixed, so in designing the string paths for this segment, we can only change the string anchor points. The anchor points are also restricted to the discrete points along the central backbone where the five intermediate disks lie. We therefore have five possible choices for s_{a_i} for each of the four strings.

To design the string routings, we run a brute-force search across all possible combinations of s_{a_i} (625 possible designs) to find the combination with the largest $\aleph_g\left(\mathbf{J}_{\ell\xi}(s)\right)$. We chose the characteristic length to be 0.0652, the kinematic radius of the actuation tendons. The global noise amplification index $\aleph_g$ was computed for a set of 320 configurations in the segment's admissible workspace, and the noise amplification index for the end disk, denoted as $\aleph_g\left(\mathbf{J}_{\ell\xi}(L)\right)$, and the noise amplification index for the third disk, denoted as $\aleph_g\left(\mathbf{J}_{\ell\xi}(s_3)\right)$, where s_3 is the arc length at which the third disk is located, were computed at each configuration.

Out of the 625 string routing designs considered, 225 resulted in a singular $\mathbf{J}_{\ell\xi}(L)$ and $(\mathbf{J}_{\ell\xi}(s_3))$. Figure 4.10 shows the noise amplification indices for 400 of the string routing designs in our brute-force search that did not result in singularities. Some designs resulted in the noise amplification being particularly close to zero. For example, one poorly conditioned routing design was a design with anchor points at disk 2, disk 4, disk 3, and disk 5, for strings 1-4, respectively, with the disk numbers

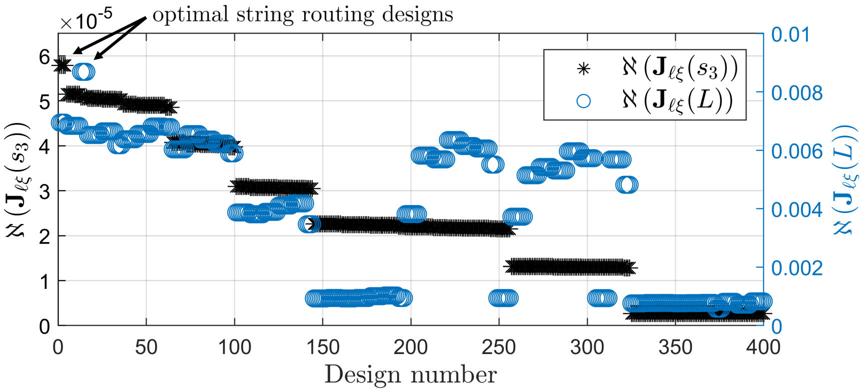


Figure 4.10: Noise amplification indices at the end disk and the third disk for all physically realizable string routing designs of the collaborative continuum robot. A large range of noise amplification indices are possible, but the noise amplification indices for the design optimal for end disk pose estimation is not significantly different than for the design optimal for Disk 3 pose estimation.

given in Fig. 4.9. This design places one string on each disk except disk 1. Although this choice might seem reasonable to a designer at first glance, its noise amplification index is $\aleph_g(\mathbf{J}_{\ell\xi}(L)) = 8.3\text{e-}3$, which is two orders of magnitude smaller than for the optimal design. This highlights the importance of carrying out the sensitivity analysis we present herein when choosing a string routing design to avoid these ill-conditioned string routings.

From the designs in Fig. 4.10, we found that the anchor points that maximize $\aleph_g\left(\mathbf{J}_{\ell\xi}(L)\right)$ were disk 4, disk 4, disk 2, and disk 2, for strings 1-4 respectively. Below, we will refer to this string routing design as the *end disk routing*. For maximizing $\aleph_g\left(\mathbf{J}_{\ell\xi}(s_3)\right)$, the optimal anchor points were disk 1, disk 1, disk 3, and disk 3, for strings 1-4, respectively. Below, we will refer to this string routing design as the *third disk routing*.

Table 4.3: Noise Amplification Indices for Optimized String Routing Designs on Collaborative Continuum Module

String Routing Design	$\aleph_g\left(\mathbf{J}_{\ell\xi}(L)\right)$	$\aleph_g\left(\mathbf{J}_{\ell\xi}(s_3)\right)$
Optimized for End Disk Pose	8.69e-3	5.08e-5
Optimized for Disk 3 Pose	6.94e-3	5.79e-5

The noise amplification indices for these two optimized designs are given in Table 4.3. Although the brute-force search optimization increased $\aleph_g\left(\mathbf{J}_{\ell\xi}(L)\right)$ for the end disk routing by 25% when compared to the middle disk routing, and the value of $\aleph_g\left(\mathbf{J}_{\ell\xi}(L)\right)$ for the middle disk routing was increased by 14% when compared to the end disk routing, we will show in our experimental results below that these changes in the noise amplification index are not large enough to have a significant effect on the pose error either at the end disk or at the third disk. Either of these string routing designs could be chosen for this robot without having a significant effect on the pose error (which we will demonstrate below).

We experimentally validated the shape sensing approach on the physical continuum robot. We routed the string encoders according to the two routing designs given by our optimization procedure, and for each string routing design, we measured the shape of the segment across a large variety of variable curvature shapes, a subset of which are shown in Fig. 4.11. The segment was mounted on a revolute joint driven by an off-the-shelf actuator (Dynamixel PH54-200-S500-R), and the angle of the revolute joint was commanded to 0, 45°, and 90°.

For each of these three angles, we commanded the segment to move from the initial home configuration to four different configurations: $\boldsymbol{\theta} = [0°, 500°]$, $\boldsymbol{\theta} = [0°, -500°]$, $\boldsymbol{\theta} = [500°, 0°]$, and $\boldsymbol{\theta} = [-500°, 0°]$, where $\boldsymbol{\theta} = [\theta_1, \theta_2]^{\mathrm{T}}$ contains the angles of the actuation capstans. The motion profile to reach these four poses was generated using a fifth-order polynomial trajectory planner, with a time of 45 seconds to move to the desired configuration from the initial home configuration. During these motions, we continuously captured the string lengths using the *rosbag* ROS package. We captured ground-truth pose of the third intermediate disk and the end disk using a stereo vision optical tracker (ClaroNav H3-60) and optical markers mounted on the robot. The data collection rate is limited by the optical tracker's ∼8.5 Hz sample rate. This experiment was carried out with the end disk routing, and then repeated for the third

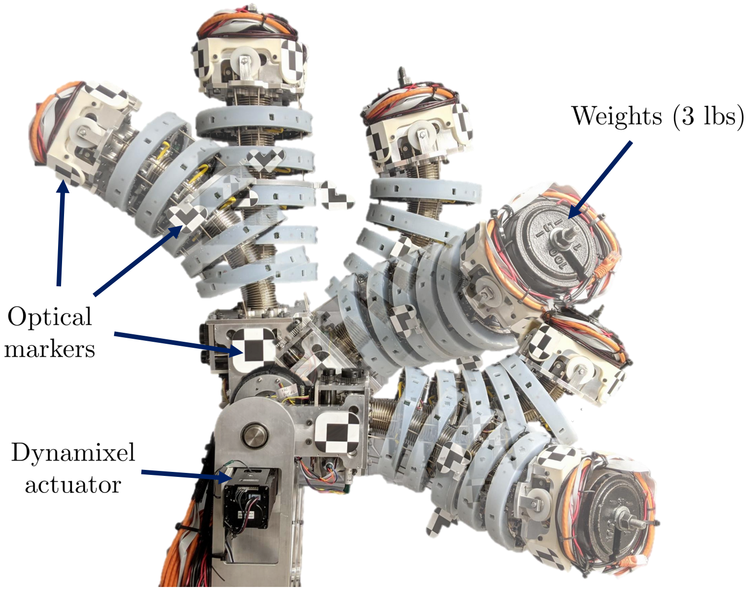


Figure 4.11: A subset of the variable curvature spatial configurations used to validate our shape sensing approach for a torsionally stiff continuum segment with straight string routing. The third and end disk poses were captured using optical trackers with and without weights attached to the end disk.

disk routing.

Using the string length measurements acquired with the string encoders, we solved for the modal coefficients using the full shape sensing model given by (4.50). To compare against a scenario where the robot did not have shape sensing encoders, we also reconstructed the shape using only the actuation variables $\boldsymbol{\theta}$. For this case, we used a modal basis with two columns where $\phi_x = \phi_y = 1$. This is identical to the commonly used constant-curvature model [30]. We report the constant-curvature model results using the data set collected with the third disk routing, but note that the errors were similar for this constant-curvature model using the end disk routing.

The mean and maximum errors of our shape sensing model (with the two different routing designs) as well as the constant curvature model are given in Table 4.4. The position error is given by $p_e(s) = \|\mathbf{p}_{model}(s) - \mathbf{p}_{meas}(s)\|$, where $\mathbf{p}_{model}(s) \in \mathbb{R}^3$ is the model-predicted position, and $\mathbf{p}_{meas}(s) \in \mathbb{R}^3$ is the measured position, and we denote

Table 4.4: Average (Maximal) Absolute Position (mm) and Orientation Errors (°) for the Segment in Fig. 4.11. The Errors are Specified for the End Disk ($s = L$) and the 3^{rd} Disk ($s = s_3$).

	End Disk Routing	Third Disk Routing	Constant Curvature
$\bar{p}_e(L)$, $\max(p_e(L))$	5.9 (14.4)	6.0 (13.8)	56.2 (104.9)
$\bar{p}_e(s_3)$, $\max(p_e(s_3))$	3.8 (10.2)	3.3 (9.2)	31.8 (58.6)
$\bar{\theta}_e(L)$, $\max(\theta_e(L))$	1.5 (8.6)	1.5 (3.9)	3.6 (14.4)
$\bar{\theta}_e(s_3)$, $\max(\theta_e(s_3))$	2.0 (6.0)	1.6 (4.2)	15.4 (28.1)

the average of $p_e(s)$ across all configurations as $\bar{p}_e(s)$. The angular error is given by:

$$\theta_e(s) = \cos^{-1}\left(\frac{\text{trace}(\mathbf{R}_{meas}(s)\mathbf{R}_{model}(s)^{\mathrm{T}} - 1}{2}\right) \tag{4.51}$$

where $\mathbf{R}_{meas}(s) \in SO(3)$ is the measured rotation matrix and $\mathbf{R}_{model}(s) \in SO(3)$ is the model-predicted rotation matrix. Both the end disk routing and the middle disk routing reduced the mean end disk errors compared to the constant curvature model by more than 89% in position and 58% in angle. The maximum end disk error was reduced compared to the constant curvature model by more than 85% in position and 40% in angle for both string routing designs. Both routing designs have average tip position errors below 2.0% of the total arc length, a significant improvement compared to the constant curvature model error of 18.7% of total arc length. The maximum tip position error of our approach was 4.8% of the total arc length, compared to 34.9% for the constant-curvature model.

The third disk routing reduced the mean position error at s_3 by 0.5 mm and the maximum by 1 mm. The third disk routing design also reduced the mean angular error at s_3 by 0.4°and the maximum by 1.8°. Figure 4.12 shows the histograms of the normalized error $e_n \in \mathbb{R}$ at the end disk and the third disk:

$$e_n = \sqrt{\|\mathbf{p}_{model} - \mathbf{p}_{meas}\| + c_\ell \theta_e} \tag{4.52}$$

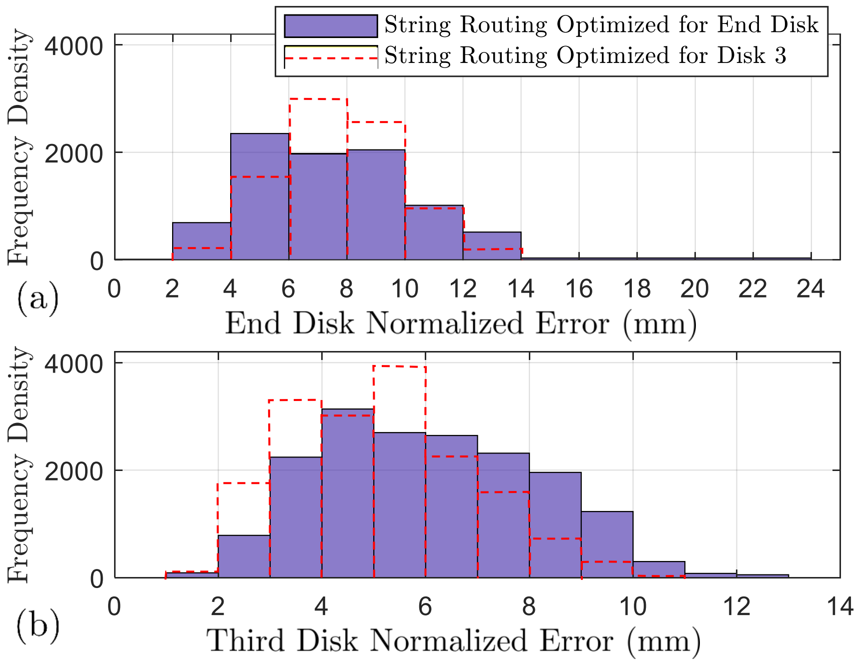


Figure 4.12: The frequency histogram of pose error when the string routing is optimized to minimize the effect of measurement noise for either the end disk pose (end disk routing) or the pose of the third disk (third disk routing). (a) The end-disk position error histograms for both routings showing that the error distribution for $\mathbf{T}(L)$ is shifted leftward compared to the third disk routing (b) The third-disk position error histograms for both routings showing that the error distribution for $\mathbf{T}(s_3)$ is shifted leftward compared to the end disk routing.

where c_ℓ is the characteristic length used in the string routing design optimization procedure (in this case, we used the kinematic radius of the disk $0.1304/2$ m as shown in Fig. 4.9). We observe that, as expected, the error distribution of $\mathbf{T}(s_3)$ is shifted leftward towards reduced error due to the increase in $\aleph_g\left(\mathbf{J}_{\ell\xi}(s_3)\right)$. However, we also observe that the maximum errors for $\mathbf{T}(L)$ were higher using the routing optimized for the end disk. This indicates that the change in the noise amplification index at $\mathbf{T}(L)$ when using the routing optimized for the third disk was not significant enough to affect the tip pose error in a way that would overcome other sources of error, i.e. friction, mechanical clearances, and the continuously parallel routing assumption. Overall, while we the general trends in the errors match the expected behavior due to the noise amplification index, we observe that the pose error of the middle and end disk was not substantially affected, meaning that either of the two optimized string routing designs could be used, providing flexibility in the string routing design choice.

In this section, we have presented the kinematic formulation for sensing deflections of robots with negligible torsional stiffness. The formulation results in a constant configuration space Jacobian and linear shape sensing equations. We validated the model and approach for a collaborative continuum robot with high torsional stiffness and constant pitch radius string paths, showing that our approach sensed the tip position with errors below 4.8% of total arc length. We now demonstrate the approach for the more general case of robots with non-negligible torsional deflections and helical string routing.

4.6 Sensing Torsional Deflections with Helical String Paths

In the section above, we validated our shape sensing approach on a segment with high torsional stiffness. Neglecting torsional stiffness significantly simplifies the model equations and reduces the computation cost of solving for the shape. However, many continuum robots have relatively low torsional stiffness, so torsional deflections cannot always be neglected. In this section, we consider helical string routing as a way to sense torsional deflections. We then validate the approach in a simulation study using a Cosserat rod mechanics model.

We chose the geometry of the our simulated continuum robot based on a concept for a modular soft continuum robot, shown in Fig. 4.13. The subsegments of the robot are built with cylindrical silicone over-molded on the outer circumference of the intermediate disks to act as a soft outer cover, and the bottom plate of each subsegment has locking tabs that mate with a slot in the top plate of the previous subsegment. A 293 mm long solid Nitinol rod passes through the center of the continuum robot to prevent compression of the structure.

Four string encoders (as described in Section 4.5) are mounted below the segment,

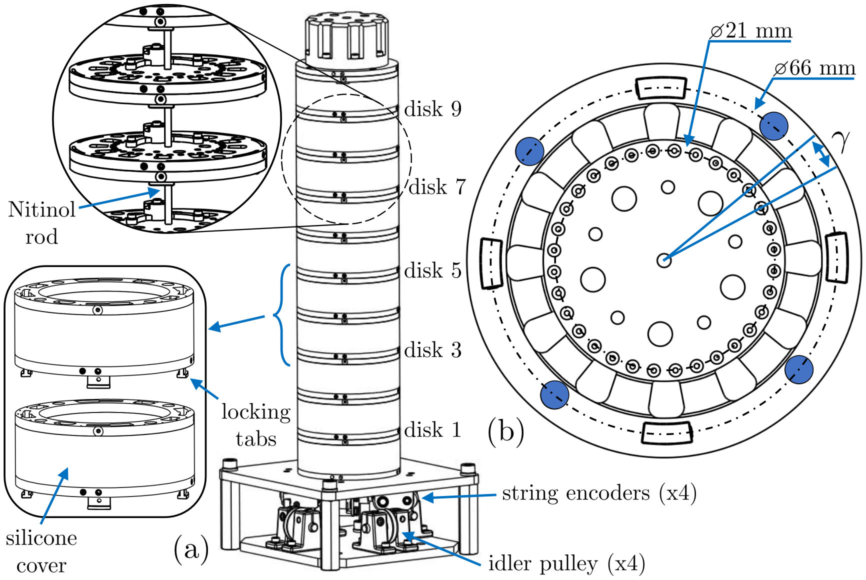


Figure 4.13: To validate our approach for torsional deflections and helical string routing, we simulated a modular soft continuum segment. The segment's interlocking subsegments are over-molded with silicone, and a Nitinol rod passes through their centers. Four string encoders are mounted at the segment's base, and each intermediate disk (b) has 32 holes to enable helical string paths.

and four tendons are routed to actuators via idler pulleys below the segment's base. In this embodiment, the actuation tendons are routed in straight paths with the tendon path given by (4.43). We assume that two of the tendons are anchored at the end disk, and that two of the tendons are anchored at disk 7. This choice in anchor points for the actuation tendons allows all four actuation tendons to provide shape information while still allowing the segment to bend in all four directions. The string encoders are routed in helical paths given by (4.53). As shown in Fig. 4.13c, the bottom plate of each subsegment has 32 holes through which the strings pass, allowing the strings to be routed in the desired helical shape. We now optimize the anchor points of the four string encoders and the twist rate of their helical paths.

The helical routing string path function is given by [45]:

$$^{t}\mathbf{r}_i(s) = r_s[\cos(\omega s + \alpha_i), \sin(\omega s + \alpha_i), 0]^{\mathrm{T}} \tag{4.53}$$

where r_s is the radius of the helical path, ω is the twist rate of the helical path (which we assumed was constant and equal for all four strings to prevent the string paths from intersecting), and α_i is an angular offset for each string. Since we have eight inputs to our shape sensing model (four passive string encoders and four active actuation tendons), we choose a modal basis with eight columns. The modal shape basis is given by (4.5) with the following shape functions:

$$\begin{aligned} \boldsymbol{\phi}_x(s) = \boldsymbol{\phi}_y(s) &= \Big[T_0, T_1(s), T_2(s) \Big]^{\mathrm{T}} \\ \boldsymbol{\phi}_z(s) &= \Big[T_0, T_1(s) \Big]^{\mathrm{T}} \end{aligned} \tag{4.54}$$

where the Chebyshev functions $T_i(s)$ are given by (4.10). We compute the string lengths and the configuration space Jacobian $\mathbf{J}_{\ell c}$ by numerically integrating (4.13) and (4.19), respectively.

We solved the optimization problem (4.29) with ϵ = 1e-7 through a brute-force search of all possible string anchor points (disks 1-10) and helical path design parameters. Given the 32 holes on the intermediate disks, as shown in Fig. 4.13, the helical path twist rate can be approximated as $\omega = n_\omega \gamma$, where $n_\omega \in \mathbb{R}$ determines the number of routing holes to skip in between the intermediate disks when routing the string, and $\gamma = \frac{2\pi}{32}$ is the angle between the routing holes, as shown in Fig. 4.13. We constrained the twist rate to $n_\omega < 2$ to prevent large twist rates, since excessive twist rates increase the possibility of binding between the string and the routing holes. With the ten possible string anchor points and two possible twist rates for four strings, there were 20,000 possible string routing designs that we evaluated. The admissible workspace was defined as a set of 32 sampled configurations that did not exceed 10° of bending and 7.5° of twist for each subsegment. We chose the characteristic length to be radius of the segment, 37.5 mm. We discarded all designs that violated $\aleph_g(\mathbf{J}_{\ell c}) \geqslant$ 1e-7. We stored the noise amplification indices for $s = L$,

$s = s_4$ (the location of the fourth disk), and $s = s_6$ (the location of the sixth disk).

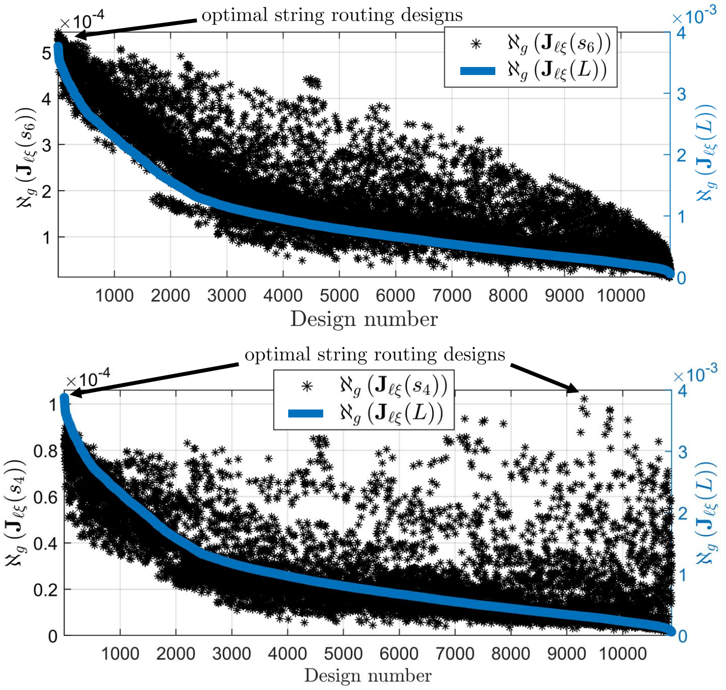


Figure 4.14: The noise amplification indices across all physically realizable string routing designs for the simulated soft robot with torsional deflections. The routing that maximizes $\aleph_g(\mathbf{J}_{\ell\xi}(s_4))$ results in a significantly reduced $\aleph_g(\mathbf{J}_{\ell\xi}(L))$, but for disk 6, $\aleph_g(\mathbf{J}_{\ell\xi}(L))$ does not significantly change.

Table 4.5: Noise Amplification Indices for Optimized String Routing Designs on Simulated Segment with Helical Routing

	Disk Used For Routing Optimization		
	End	6th	4th
$\aleph_g\left(\mathbf{J}_{\ell\xi}(L)\right)$	3.47e-3	3.50e-3	3.35e-4
$\aleph_g\left(\mathbf{J}_{\ell\xi}(s_6)\right)$	5.24e-4	5.40e-4	2.27e-4
$\aleph_g\left(\mathbf{J}_{\ell\xi}(s_4)\right)$	7.91e-5	8.27e-5	8.85e-5

Figure 4.14 shows the noise amplification indices across all string routing designs that did not violate $\aleph_g(\mathbf{J}_{\ell c}) \geqslant$ 1e-7, sorted in descending order of $\aleph_g\left(\mathbf{J}_{\ell\xi}(L)\right)$. Table 4.5 shows the values $\aleph_g\left(\mathbf{J}_{\ell\xi}(L)\right)$, $\aleph_g\left(\mathbf{J}_{\ell\xi}(s_6)\right)$, and $\aleph_g\left(\mathbf{J}_{\ell\xi}(s_4)\right)$ for the designs that maximize each of these three values (which are also indicated with arrows in Fig. 4.14). We observe that the design that maximizes $\aleph_g\left(\mathbf{J}_{\ell\xi}(s_6)\right)$ results in only a 0.8%

decrease in $\aleph_g\left(\mathbf{J}_{\ell\xi}(L)\right)$, however, the design that maximizes $\aleph_g\left(\mathbf{J}_{\ell\xi}(s_4)\right)$ results in a 90% decrease in $\aleph_g\left(\mathbf{J}_{\ell\xi}(L)\right)$. Choosing the string routing design that maximizes $\aleph_g\left(\mathbf{J}_{\ell\xi}(s_4)\right)$ would therefore tend to increase the pose error at $s = L$. We also note that $\aleph_g\left(\mathbf{J}_{\ell\xi}(s_4)\right)$ only increases by 4.6% between the end disk routing and the fourth disk routing, indicating that we would not expect to see a significant change in the pose error at s_4 between these two designs. We will now validate these predicted behaviors in a simulation study.

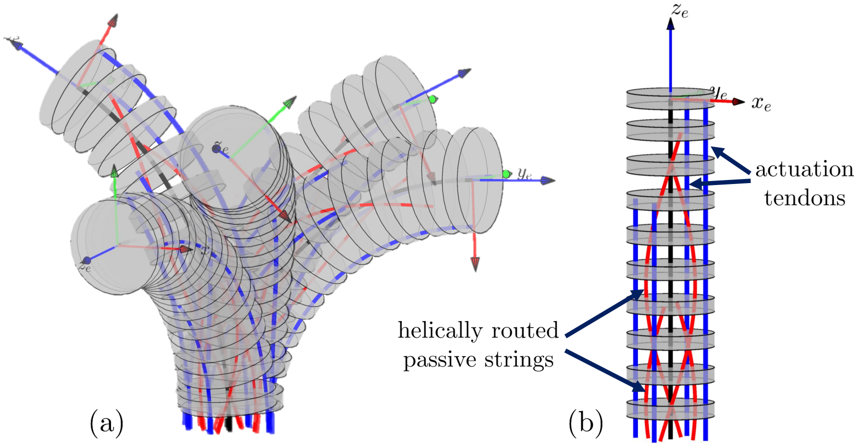


Figure 4.15: (a) A subset of the spatial configurations used to validate our shape sensing approach on a segment subject to torsional loads utilizing helical routing, and (b) the segment in its zero-curvature configuration.

We simulated the robot in Fig. 4.13 using the Cosserat rod model from [45]. This model takes as inputs the applied tensions on the actuation tendons as well as the external forces/moments applied to the tip of the segment and returns the curvature, shear, and extension of the segment. We simulated the robot with a preload force of 25 N applied on all four tendons, and applied additional forces of up to 300 N on the tendons to bend the segment in 8 different directions. For each direction, we applied 12 different external wrenches with forces of ±20 N and moments of ± 4 Nm, expressed in the world frame. We selected these loads to generate a large variety of shapes without exceeding the maximum curvature to keep the segment within its admissible workspace. We also included constant forces of 3.3 N on the helical strings due to the constant-torque spring in the string encoder housing.

For each tendon tension/external wrench combination, we started with the segment initially unloaded and incremented the external wrench with 5 steps to incrementally apply the load and solve for the shape at each external wrench increment. We did this to ensure that a good initial guesswas provided to the solver. This simulation resulted in 480 different configurations of the continuum segment. For each configuration, we applied normally distributed noise to the simulated string/tendon lengths. The noise was normalized to the maximum calibration error of 0.27 mm found in Section 3.4. We then used these lengths as inputs to our kinematic shape sensing model to compare the accuracy of our shape sensing approach. Our unoptimized MATLAB 2019b implementation used MATLAB's *lsqnonlin()* to solve (4.17) for $\mathbf{c}$ at rate of ∼4 Hz on average, using an 80 point trapezoid rule to compute $\mathbf{J}_{\ell c}$. Future code can be orders of magnitude faster with direct implementation in C++.

We compared the mechanics model to two different shape sensing models. The first used all 8 length measurements (4 string encoders and 4 actuation tendons) with the modal basis given by (4.54). The second used only the 4 actuation tendons to compare our shape sensing model to a scenario without any helical shape sensing strings. For this second model, the modal basis has 4 columns and is given by (4.42), with the modal functions given by $\boldsymbol{\phi}_x(s) = \boldsymbol{\phi}_y(s) = \begin{bmatrix} T_0, T_1(s) \end{bmatrix}^{\mathrm{T}}$.

Table 4.6: Average (Maximal) Absolute Position (mm) and Orientation Errors (°) for the Segment in Fig. 4.13. Errors Are Specified For the End Disk ($s = L$) and the 4^{th} Disk ($s = s_4$)

	End Disk Routing	Fourth Disk Routing	w/o Passive Strings
$\bar{p}_e(L)$, max $(p_e(L))$	1.29 (3.39)	1.94 (5.64)	5.61 (32.87)
$\bar{p}_e(s_4)$, max $(p_e(s_4))$	0.45 (0.90)	0.52 (1.51)	0.63 (3.07)
$\bar{\theta}_e(L)$, max $(\theta_e(L))$	1.17 (3.04)	6.72 (21.78)	4.55 (23.44)
$\bar{\theta}_e(s_4)$, max $(\theta_e(s_4))$	0.61 (1.55)	0.57 (1.20)	2.15 (12.17)

The statistical results are given in Table 4.6, where we denote the string routing

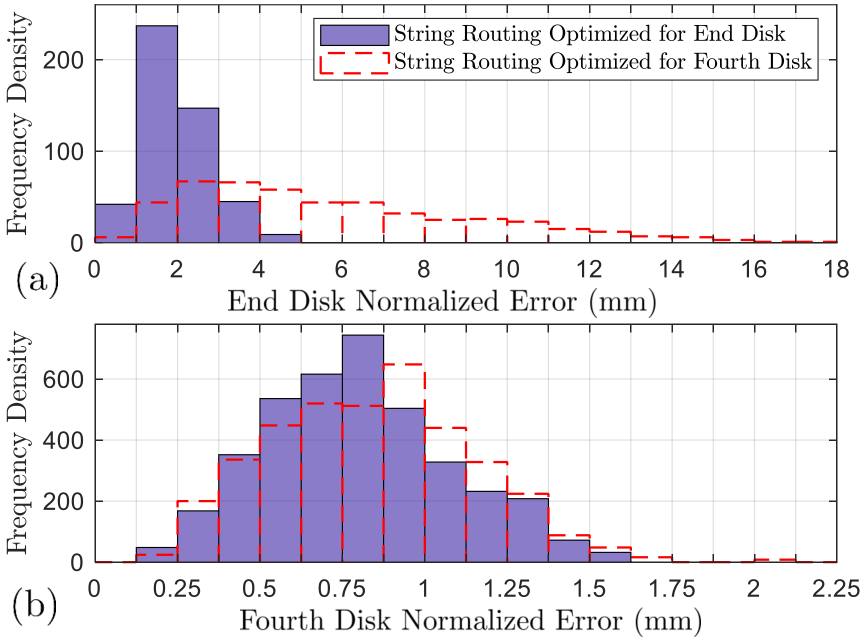


Figure 4.16: Histograms of normalized pose error at the end disk and at the fourth intermediate disk for the routing that maximizes $\aleph_g(\mathbf{J}_{\ell\xi}(L))$ and for the routing that maximizes $\aleph_g(\mathbf{J}_{\ell\xi}(s_4))$. The pose error at the fourth intermediate disk is not significantly effected by the change in routing design, but the tip pose error is significantly effected due to the larger change in $\aleph_g(\mathbf{J}_{\ell\xi}(L))$ between the two designs.

design that maximizes $\aleph_g\left(\mathbf{J}_{\ell\xi}(L)\right)$ as the *end disk routing* and the string routing design that maximizes $\aleph_g\left(\mathbf{J}_{\ell\xi}(s_4)\right)$ as the *fourth disk routing*. Both optimized string routing designs had maximum absolute end disk position errors below 2% of arc length. However, the fourth disk routing resulted in a 616% increase in $\max(\theta_e(L))$ compared to the end disk routing. There was also a 474% increase in $\bar{\theta}_e(L)$, and small increases in both $\bar{p}_e(L)$ and $\max\left(p_e(L)\right)$ when compared to the fourth disk routing. This confirms the expected behavior due to the large decrease in $\aleph_g\left(\mathbf{J}_{\ell\xi}(L)\right)$ in the fourth disk routing given in Table 4.5.

Figure 4.16 shows the histograms of the error at the end disk and the fourth disk, normalized using (4.52). We observe that the end disk error with the end disk routing is shifted leftward compared to the fourth disk routing errors, as expected due to the large change in $\aleph_g\left(\mathbf{J}_{\ell\xi}(L)\right)$. However, since $\aleph_g\left(\mathbf{J}_{\ell\xi}(s_4)\right)$ did not significantly change, we do not see a significant difference in the error distribution for the fourth disk pose, and in fact see a small shift rightward with the fourth disk routing. This increase in error using the fourth disk routing is explained by the known fact that kinematic conditioning indices are not guaranteed to directly correlate with the true errors (see [165]). Large changes in the conditioning index should be sought to increase the possibility of reducing the true errors. Note that we repeated the simulation study above with the external wrenches applied to disk four instead of applied the end disk to simulate the scenario where the robot is bracing against the environment or is being collaboratively guided by a worker, and obtained histograms that looked similar to 4.16.

Compared to the model without string measurements, the end disk routing reduced the maximum end disk position error by 90% and the maximum angular end disk error by 87%. The fourth disk routing reduced the fourth disk maximum position error by 51% and the fourth disk maximum angular error by 90%. These simulation results demonstrate that 4 passive string (together with the four actuation tendons) can significantly improve the accuracy of continuum robot kinematics over actuation-based sensing alone.

4.7 Conclusions

In this chapter, we have presented a Lie group kinematic formulation for capturing variable curvature deflections of continuum robots using general string encoder routing. We used this formulation for a sensitivity analysis of the error propagation from error in string extension measurements to error in the modal coefficients and error in the central backbone shape. This analysis allows the designer to avoid string encoder routings that lead to ill-conditioned Jacobians. We then applied the approach on a planar example, a segment with high torsional stiffness, and for a robot subject to torsional deflections using helical routing, showing that this shape sensing approach can result in mean and maximal absolute position error below 2% and 5% of arc length, respectively. We believe this sensing approach is a practical and relatively low-cost way of sensing the variable curvature deflections of large continuum robots, and that the kinematic formulation and analysis presented herein enables practitioners to incorporate this sensing approach into new continuum robot designs.

We have shown that four string encoders can provide accurate shape sensing, and have demonstrated a practical physical embodiment of a segment utilizing this approach, but a drawback of this approach compared to other sensing methods is the physical space required to mount the string encoders within the robot. Directions for future work include investigating more tightly integrated design of the string encoder mechanical components into the robot body to overcome this drawback, as well as using the shape sensing information and Lie group kinematic formulation to provide updates to mechanics models of continuum robots.

CHAPTER 5

SOLVING COSSERAT ROD MODELS VIA COLLOCATION AND THE MAGNUS EXPANSION

This chapter is adapted from [156] and has been reproduced with the permission of the publisher and the co-authors. © 2020 IEEE.

In this chapter, we present an approach for solving the statics of a tip-loaded Cosserat rod using orthogonal collocation and a Lie group integration method. This work is a preliminary study on the potential benefits of these methods for robotics problems. After reviewing prior work on solving the Cosserat rod equations, we present our approach and validate it in a simulation study. In Chapter 6, we use the same Lie group kinematics equations to derive the configuration-space and task-space compliance matrices of tendon-actuated continuum robots.

5.1 Motivation and Background

Continuum and soft robot architectures have been studied for a variety of useful applications [7, 8], but their passively compliant structures make them difficult to model. There are many different kinematic and dynamic models presented in the literature (see [7, 8, 30, 31] for reviews), but one commonly used method is to model the robot's flexible structure as one or more Cosserat rods. This approach has been experimentally validated for concentric tube robots [43, 44], tendon-driven robots [45, 47], multi-backbone robots [48], and soft robots with fluidic [33] and tendon [166] actuation in both kinematic and dynamic studies [32].

Experimental validations have shown relatively accurate open-loop prediction of shape (position errors of 1-8% of arc length are typical), however, computing the model involves numerically solving a set of boundary value problems (BVPs) which can be computationally expensive. Although a number of works have demonstrated implementations fast enough for control in the cases of concentric tube robots [43], parallel continuum robots [46], and single-backbone tendon-driven robots [32], less accurate modeling methods are still attractive due to their low computational cost when compared to the Cosserat rod models [47]. Furthermore, in cases where the model consists of many kinematically coupled Cosserat rods (e.g. multi-backbone robots [167, 48] and eccentric pre-curved tube robots [49]), the computational cost of the Cosserat rod models is a significant obstacle and more efficient numerical methods are still needed.

Another drawback of Cosserat rod modeling is that after the BVP has been numerically solved (typically with a shooting method), it can be difficult to compute forward/inverse Jacobians when many kinematic constraints are active. Although in many cases the partial derivatives associated with the Jacobian can be computed together with the forward integration of the Cosserat differential equations (DEs) [51], for more complicated continuum structures one typically has to resort to finite-difference estimation. A numerical method that results in an analytical expression can aid in computing these matrices and conducting other analytical analysis of the rod equilibrium shape for design and control.

This chapter is motivated by these two limitations of Cosserat rod models and is a preliminary step towards addressing them. We propose a method that solves for the rod's curvature distribution with global orthogonal collocation and uses the Magnus expansion, a Lie group integration method, to recover the shape of the rod. Solving the BVP in this way provides some computational advantages and results in a product of matrix exponentials expression for the shape, which, as shown in [39, 41], allows the Jacobian to be derived in an analytic form. Although we consider the case of a single rod and additional evaluation is need for practical robotics scenarios, our results show that this method can potentially be competitive with other approaches.

Collocation, and other weighted residuals methods, have been demonstrated previously [33, 31, 34, 35, 36], but these works apply polynomial interpolation on internal wrenches or on the position and orientation instead of curvature. Curvature-based parameterizations have been used in [39, 40], but they do not combine this with collocation, and they use constant-twist deformation elements, in contrast to the global interpolation functions we use here. A polynomial curvature model was used in [147], although this work was focused on control of a planar robot and not modeling of Cosserat rods. In [38], a curvature parametrization is combined with collocation for Cosserat rod dynamics, but the Magnus expansion is not used.

Our method is most similar to recent works [41, 42] that also use polynomial approximations on curvature within a Lie-group framework. The Magnus expansion is used in [41] and Chebyshev polynomials are used in [42]. This chapter complements and extends these works by 1) demonstrating the computational benefits of combining Chebyshev orthogonal collocation with the Magnus expansion for Cosserat rod BVPs, 2) analysing the maximum step sizes to guarantee convergence of the Magnus expansion, and 3) validating the approach with high-order polynomials against two known methods in a simulation study.

5.2 The Cosserat Rod Equations

Here we briefly review the Cosserat rod equations of static equilibrium. We assume that shear strains and extension are negligible, which has been shown to be a reasonable assumption for long, slender rods [43]. We also assume the rod is not subject to distributed loads, is not pre-curved, and has a uniform cross-section and bending stiffness. Although we consider this simpler case for brevity, the methods in this chapter extend to these more general cases, which have been discussed in other work [45].

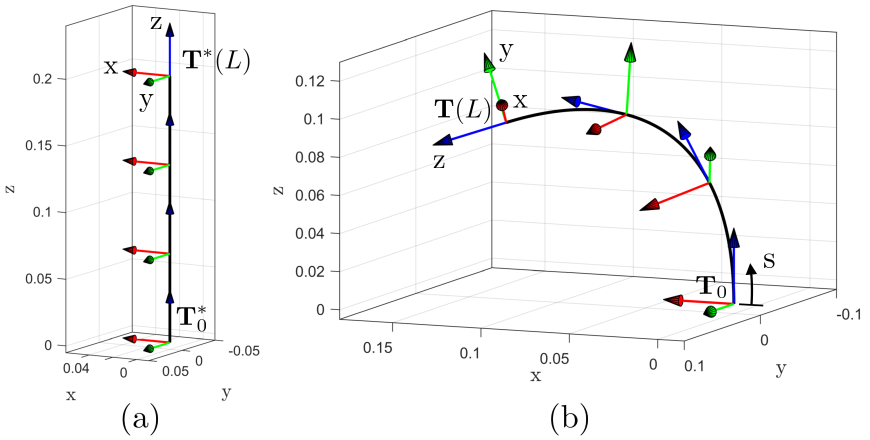


Figure 5.1: Kinematic notation and frame assignment: (a) the rod in its undeflected state with reference frames $\mathbf{T}^*(s)$, and (b) the rod after undergoing a spatial deflection.

Under these assumptions, the shape of a deflected rod of length L is parameterized as a matrix function of arc length $s \in [0, L]$ given as a homogenous transformation $\mathbf{T}(s)$:

$$\mathbf{T}(s) = \begin{pmatrix} \mathbf{R}(s) & \mathbf{p}(s) \\ 0 & 1 \end{pmatrix} \in SE(3) \tag{5.1}$$

where $\mathbf{p}(s) \in \mathbb{R}^3$ specifies the rod's shape in 3D and $\mathbf{R}(s) \in SO(3)$ describes the orientation of each local material frame (as shown in Fig. 5.1).

We assign reference frames $\mathbf{T}^*(s)$ to the unloaded rod's curve as shown in Fig. 5.1(a) with the z-axis of $\mathbf{T}^*$ aligned with the rod and pointing towards its tip. Defining $\mathbf{u} = [u_x, u_y, u_z]^{\mathrm{T}}$ as a vector of curvatures, we describe the motion of $\mathbf{T}(s)$ along the curve for unit-speed traversal along the arc length by a twist vector $\boldsymbol{\eta} = [\mathbf{u}^{\mathrm{T}}(s), 0, 0, 1]^{\mathrm{T}}$ expressed in the moving frame. Using the wedge operator ^, we map $\mathbf{u}$ to its skew-symmetric matrix $\hat{\mathbf{u}} \in so(3)$ and $\boldsymbol{\eta}$ to its $se(3)$ element, which is defined as:

$$\begin{gathered} \hat{\boldsymbol{\eta}}(s) = \mathbf{X}(s) = \begin{pmatrix} \hat{\mathbf{u}}(s) & \mathbf{e}_3 \\ 0 & 0 \end{pmatrix} \in se(3) \\ \mathbf{X}(s) = \mathbf{T}^{-1}(s)\mathbf{T}'(s) \end{gathered} \tag{5.2}$$

where $\mathbf{T}'(s)$ is the derivative of $\mathbf{T}$ with respect to arc length and $\mathbf{e}_3 = [0, 0, 1]^{\mathrm{T}}$. We use a moving frame twist so that the rod's internal moment can be obtained directly from the local curvature:

$$\mathbf{m}(s) = \mathbf{R}(s)\mathbf{K}\mathbf{u}(s), \quad \mathbf{m}(s) \in \mathbb{R}^3 \tag{5.3}$$

where $\mathbf{K} = \operatorname{diag}(EI, EI, JG)$ is the rod's arc-length normalized bending stiffness matrix. Note we have expressed $\mathbf{m}(s)$ in world frame since external wrenches are more easily expressed in world frame.

For a known tip-applied wrench, the Cosserat rod ordinary differential equations (ODEs) simplify to [45]:

$$\begin{aligned} \mathbf{T}'(s) &= \mathbf{T}(s)\mathbf{X}(s) \\ \mathbf{u}'(s) &= \mathbf{g}(\mathbf{u}(s)) = -\mathbf{K}^{-1}\left(\hat{\mathbf{u}}(s)\mathbf{K}\mathbf{u}(s) + \hat{\mathbf{e}}_3\mathbf{R}^{\mathrm{T}}(s)\mathbf{f}_e\right) \end{aligned} \tag{5.4}$$

where $\mathbf{u}'(s)$ denotes a derivative with respect to s and $\mathbf{f}_e$ is a force at the rod's tip expressed in world frame. The boundary conditions for a known applied tip wrench are given by:

$$\mathbf{u}(L) = \mathbf{K}^{-1}\mathbf{R}^{\mathrm{T}}(L)\mathbf{m}_e \tag{5.5}$$

where $\mathbf{m}_e = [m_{e,x}, m_{e,y}, m_{e,z}]^{\mathrm{T}}$ is the tip moment expressed in world frame. Solving the ODE's in (5.4) with the boundary conditions (5.5) provides the frames $\mathbf{T}(s)$ along the rod which gives the shape of the rod in space.

This boundary value problem (BVP) is typically solved via the shooting method. This has been successfully applied to a number of continuum robot architectures for forward/inverse kinematics [25] and forward dynamics [32]. One drawback of the shooting method is that it involves many forward integrations of $\mathbf{T}'(s)$ and $\mathbf{u}'(s)$ and therefore can be computationally expensive for Cosserat rod models with many kinematic constraints (e.g. [48, 49]). Another drawback is that, once a solution to the

BVP is found, computing Jacobians requires finite difference approximation. Several works addressed this issue for some continuum robot architectures [51, 46], but this difficulty remains for architectures with many coupled Cosserat rods.

In the sections below, we describe a procedure for solving (5.4) and (5.5) that uses collocation to solve $\mathbf{u}'(s)$, which circumvents the need to numerically integrate $\mathbf{u}'(s)$, and a result from the geometric integration literature called the *Magnus expansion* to forward integrate $\mathbf{T}'(s)$. The combination of these two known techniques allows for the solution to the BVP to be expressed as a product of matrix exponentials, an analytical expression that can be used for further analytical analysis (e.g. Jacobian-based studies [39, 40]).

5.3 Solving via Orthogonal Collocation

Here we show how to solve (5.4) via orthogonal collocation, a direct variational approach that has been applied to a variety of DEs and BVPs [168, 169, 159]. First, we review polynomial interpolation and collocation.

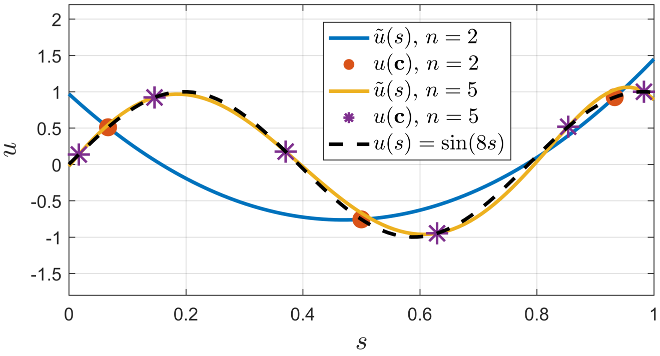


Figure 5.2: Example of polynomial interpolation of a function $u(s) = \sin(8s)$ using a 2nd order and 5th order Chebyshev polynomials with interpolation nodes at the zeros of the nth order Chebyshev polynomial. Orthogonal interpolation provides rapid convergence with increasing n.

Any continuous function can be approximated by obtaining the function values at a set of *interpolation nodes*, then fitting an *interpolating polynomial* to those function values. If we choose an interpolating polynomial of sufficiently high order, it can provide a reasonable approximation of the function. Figure 5.2 shows an example of this, where $u(s) = \sin(8s)$ is approximated to varying degrees of fidelity by $\tilde{u}(s)$ given as a 2nd order polynomial or a 5th order polynomial. To avoid Runge's phenomenon, the interpolation nodes are chosen to be the zeros of an orthogonal polynomial. Here we choose Chebyshev polynomials but other orthogonal polynomials could also be used.

In a collocation method, an interpolating polynomial is used to find an approximate solution to a DE. An interpolating polynomial is chosen to approximate the unknown solution to the DE (which in our case is $\boldsymbol{\eta}(s)$, the unknown twist distribution), a set of *collocation points* (similar to the interpolation nodes above) are chosen, and it is enforced that the interpolating polynomial satisfy the boundary conditions as well as the DE at the collocation points. This results in a set of algebraic equations that can be solved using standard nonlinear root-finding approaches. We now discuss how to carry out this procedure for a Cosserat rod.

In our context, we seek to find the unknown twist distribution $\boldsymbol{\eta}(s)$ that satisfies the ODEs (5.4) and the boundary conditions (5.5). We choose to describe this unknown twist distribution as a set of nth order Chebyshev polynomials of the first kind, denoted as $\tilde{\boldsymbol{\eta}}(s) = [\tilde{\mathbf{u}}(s), 0, 0, 1]^{\mathrm{T}}$, where $\tilde{\mathbf{u}}(s) = [\tilde{u}_x(s), \tilde{u}_y(s), \tilde{u}_z(s)]^{\mathrm{T}}, s \in [0, L]$. Note that here we use three interpolating polynomials to approximate $\boldsymbol{\eta}(s)$, but for a general Cosserat rod case where shear strains are included, six interpolating polynomials would be needed. We choose the collocation points $\mathbf{c} = [c_0, \ldots, c_n]^{\mathrm{T}}$, $c_i \in [0, L]$, $i = 0 \ldots n$ to be the Chebyshev polynomial zeros and call the function values $\tilde{\mathbf{u}}(c_i)$ the *collocation values.*

The Chebyshev polynomials of the first kind can be conveniently represented by a recurrence relation [159]:

$$\begin{gathered}T_n(x) = 2x\,T_{n-1}(x) - T_{n-2}(x), \quad n = 2, 3, \ldots \\ T_0 = 1, \quad T_1(x) = x\end{gathered} \tag{5.6}$$

where $T_n(x), x \in [-1, 1]$ is the nth Chebyshev polynomial. To shift the Chebyshev polynomials to the domain $s \in [0, L]$ we apply the linear transformation

$$x(s) = \frac{2s - L}{L} \tag{5.7}$$

and evaluate $T_n(x(s))$ via (5.6). Henceforth, we denote $T_n(x(s))$ as simply $T_n(s)$ with (5.7) implied.

In a typical direct variational method, one describes the interpolating polynomial as an expansion in a basis of orthogonal polynomials, which requires computing modal coefficients from collocation values. For example, $\tilde{u}_x(s)$ may be represented by the nth order interpolating polynomial:

$$\tilde{u}_x(s) = \frac{1}{2}a_0 T_0(s) + \sum_{i=1}^{n} a_i T_i(s) \tag{5.8}$$

where the modal coefficients a_i are found via:

$$a_i = \frac{2}{n+1}\sum_{k=0}^{n} \tilde{u}_x(c_k) T_i(c_k) \tag{5.9}$$

In an orthogonal collocation method, computing the modal coefficients is avoided by using a *differentiation matrix*. By taking advantage of the discrete orthogonality of Chebyshev polynomials, it has been shown that the derivatives at the collocation points can be written as a linear combination of the collocation values [159, 170]:

$$\frac{d}{ds}\begin{pmatrix} u_x(c_0) \\ \vdots \\ u_x(c_n) \end{pmatrix} = \mathbf{D}_n \begin{pmatrix} u_x(c_0) \\ \vdots \\ u_x(c_n) \end{pmatrix} \tag{5.10}$$

where the element in row $(i+1)$ and column $(j+1)$ of $\mathbf{D}_n$ is given by:

$$d_{ij} = \begin{cases} \frac{1}{2}\frac{T''_{n+1}(c_i)}{T'_{n+1}(c_i)}, & i = j \\ \frac{T'_{n+1}(c_i)}{(c_i - c_j)T'_{n+1}(c_j)}, & i \neq j \end{cases} \tag{5.11}$$

Note that the above result requires that the collocation points be the Chebyshev zeros and that $\mathbf{D}_n$ can be computed offline if the order of the interpolating polynomial is chosen *a priori*.

We now assemble the collocation values into a matrix $\mathbf{U}_c$ where each column contains an interpolating polynomial:

$$\mathbf{U}_c = \begin{pmatrix} \tilde{u}_x(c_0) & \tilde{u}_y(c_0) & \tilde{u}_z(c_0) \\ \vdots & \vdots & \vdots \\ \tilde{u}_x(c_n) & \tilde{u}_y(c_n) & \tilde{u}_z(c_n) \end{pmatrix} = \begin{pmatrix} \tilde{\mathbf{u}}^{\mathrm{T}}(c_0) \\ \vdots \\ \tilde{\mathbf{u}}^{\mathrm{T}}(c_n) \end{pmatrix} \tag{5.12}$$

We want the interpolating polynomials to satisfy the boundary conditions and (5.4) at the collocation points. To ensure a square error residual Jacobian, we remove the last row of $\mathbf{D}_n$ to form $\mathbf{D}_{n-1}$ and form a matrix of error residuals:

$$\mathbf{E} = \begin{pmatrix} \mathbf{D}_{n-1}\mathbf{U}_c \\ \tilde{\mathbf{u}}^{\mathrm{T}}(L) \end{pmatrix} - \begin{pmatrix} \mathbf{g}^{\mathrm{T}}(\tilde{\mathbf{u}}(c_0)) \\ \vdots \\ \mathbf{g}^{\mathrm{T}}(\tilde{\mathbf{u}}(c_{n-1})) \\ \left(\mathbf{K}^{-1}\mathbf{R}^{\mathrm{T}}(L)\mathbf{m}_e\right)^{\mathrm{T}} \end{pmatrix} \tag{5.13}$$

where we evaluate $\mathbf{g}(\tilde{\mathbf{u}}(c_i))$ by plugging the collocation value $\tilde{\mathbf{u}}(c_i)$ into (5.4). Note that $\mathbf{u}(L)$ is not one of the collocation values and must be computed from the interpolating polynomial. We show how to do this in Section 5.5.

Equation (5.13) is a set of nonlinear algebraic equations we must now solve for the collocation values $\tilde{\mathbf{u}}(c_i), i \in 0, \ldots, n$. This is achieved by minimizing the error in (5.13) using a nonlinear solver (e.g. Levenberg-Marquardt). Specifically, the error is defined as $\mathbf{e} = \text{vec}(\mathbf{E})$, where $\text{vec}(\mathbf{E})$ arranges the columns of $\mathbf{E}$ into a vector. Once $\tilde{\mathbf{u}}(c_i)$ are found, the modal coefficients and the interpolating polynomial $\tilde{\mathbf{u}}$ are defined. Therefore, $\mathbf{X}(s)$ is defined in terms of the Chebyshev polynomials.

Computing $\mathbf{g}(\tilde{\mathbf{u}}(c_i))$ in (5.13) requires integrating $\mathbf{T}'(s)$ to find $\mathbf{R}(s)$ at the collocation points. Examples of methods to do this include explicit Runge-Kutta methods, quaternion integration [171], and a variety of geometric integration methods [172, 155]. Although any of these approaches could be combined with collocation, here we take the geometric integration approach. We observe that $\mathbf{T}(s)$ is an element of the Lie group $SE(3)$, $\mathbf{X}(s)$ is an element of the Lie algebra $se(3)$, and that given a set of collocation values $\tilde{\mathbf{u}}(c_i)$ (which would be guessed at each iteration of a nonlinear solver), the twist distribution $\mathbf{X}(s)$ is known via the interpolating polynomial. Therefore, $\mathbf{T}'(s)$ in (5.4) represents a linear DE on a Lie group, a class of problems that has received extensive study in other work [155]. We use a known result from the Lie group integration literature called the *Magnus expansion* to forward integrate $\mathbf{T}'(s)$. Note that a similar Lie algebra expansion was given in [173], which could also be used here in place of the Magnus expansion.

The benefits of using the Magnus expansion are 1) the integration is done on the Lie algebra $se(3)$ and mapped to $SE(3)$ via an exponential mapping, so $\mathbf{T}(s)$ is guaranteed to stay on $SE(3)$, 2) the quadrature method given in [155] for the Magnus expansion is numerically efficient and allows for closed-form gradients to be computed, and 3) the resulting solution to the BVP is given as a product of matrix exponentials that can be used for further analytical study (e.g. computing Jacobians). In the next section, we provide the main results on the Magnus expansion without proof and refer the reader to [155] for additional details.

5.4 The Magnus Expansion

It was shown in [155] that for sufficiently small s the solution to $\mathbf{T}'(s) = \mathbf{X}(s)\mathbf{T}(s)$ can be expressed as a matrix exponential of a twist $\boldsymbol{\psi} = [\boldsymbol{\psi}_u, \boldsymbol{\psi}_v]^{\mathrm{T}} \in \mathrm{I\!R}^6$:

$$\begin{aligned} \mathbf{T}(s) &= \mathbf{T}_0 e^{\boldsymbol{\Psi}(s)} \\ \boldsymbol{\Psi}(s) \triangleq \hat{\boldsymbol{\psi}}(s) &= \begin{pmatrix} \hat{\boldsymbol{\psi}}_u(s) & \boldsymbol{\psi}_v(s) \\ 0 & 0 \end{pmatrix} \in se(3) \end{aligned} \tag{5.14}$$

A short proof in [155] shows that the twist matrix $\boldsymbol{\Psi}(s)$ satisfies the differential equation

$$\boldsymbol{\Psi}'(s) = \mathrm{dexp}^{-1}_{-\boldsymbol{\Psi}(s)}(\mathbf{X}(s)) \quad \boldsymbol{\Psi}(0) = \mathbf{0} \tag{5.15}$$

where the dexp operator is defined as:

$$\mathrm{dexp}^{-1}_{\boldsymbol{\Psi}} = \sum_{i=0}^{\infty} \frac{B_i}{i!} \mathrm{ad}^i_{\boldsymbol{\Psi}} \tag{5.16}$$

where B_i are the Bernoulli numbers and $\mathrm{ad}_{\boldsymbol{\Psi}}$ denotes the 6 × 6 adjoint representation of an element in $se(3)$ [161]:

$$\mathrm{ad}_{\boldsymbol{\Psi}} = \begin{pmatrix} \hat{\boldsymbol{\psi}}_u(s) & 0 \\ \hat{\boldsymbol{\psi}}_v(s) & \hat{\boldsymbol{\psi}}_u(s) \end{pmatrix} \tag{5.17}$$

To compute $\mathrm{dexp}^{-1}_{-\boldsymbol{\Psi}}$, one can either truncate the infinite series or, for special cases, derive a closed-form expression as done in [161] for $se(3)$. The DE in (5.15) can then be numerically integrated with a standard Runge-Kutta method to find $\boldsymbol{\Psi}(s)$.

To reduce the cost of numerically integrating, we take an alternative approach that avoids computing (5.16). In [174], Magnus solved (5.15) via Picard iteration, leading to a solution for $\mathbf{\Psi}(s)$ written as an infinite series of terms consisting of integrals of commutators. In [155], order analysis showed which terms can be dropped for a given order, resulting in the following fourth order Magnus expansion:

$$\begin{aligned}\mathbf{\Psi}^{[4]}(s) = &\int_0^s \mathbf{X}(\eta)\,\mathrm{d}\eta \\ &+ \frac{1}{2}\int_0^s \left[\int_0^{\eta_1} \mathbf{X}(\eta_2)\,\mathrm{d}\eta_2, \mathbf{X}(\eta_1)\right] \mathrm{d}\eta_1\end{aligned} \tag{5.18}$$

where the matrix commutator is given by $[\mathbf{X}_1, \mathbf{X}_2] = \mathbf{X}_1\mathbf{X}_2 - \mathbf{X}_2\mathbf{X}_1$. We do not replicate it here, but [155] also provides the sixth order expansion with 7 terms and up to 4 integrals per term.

At first glance, the fourth and sixth order Magnus expansions seem expensive to compute, however, it was shown in [155] that both expansions can be efficiently computed with Gaussian quadrature. Gaussian quadrature is an approach for approximating definite integrals via interpolating polynomials and leads to expressing the integral as a weighted sum of the function values, which we call *quadrature values*, evaluated at the *quadrature points*, which are chosen to be the zeros of an orthogonal polynomial.

In addition to showing how to compute the expansion via Gaussian quadrature, it was shown in [155] that skew symmetry of the commutators allows many of the terms in the quadrature to be combined. It was also shown via order analysis that if the quadrature points are chosen to be symmetric about $\frac{1}{2}h$, where h is width of interval between two adjacent collocation points, many of the terms in the quadrature can be dropped for a given order. We first provide here the steps to compute the quadrature on the interval $[0, h]$ (as given in [155]) then describe how this can be combined with the collocation method above.

To integrate between collocation point c_i and c_{i+1}, we choose the quadrature interpolating polynomials to be the Legendre polynomials shifted to $[0,1]$ and choose ν quadrature points $\mathbf{t} = [t_1, \ldots, t_\nu]^{\mathrm{T}} \in [0,1]$ to be the zeros of the Legendre polynomials since they are symmetric about $\frac{1}{2}$. We then form the quadrature values:

$$\mathbf{X}_k = h\mathbf{X}(c_i + t_k h), \quad k = 1, 2, \ldots, \nu \tag{5.19}$$

Note that an order four quadrature requires $\nu \geqslant 2$ and an order six quadrature requires $\nu \geqslant 3$. In [155, 175], a change of basis is carried out to take advantage of the symmetry of the Magnus expansion. The change of basis is done by finding the solution of the following Vandermonde system:

$$\sum_{i=1}^{\nu} \left(t_k - \tfrac{1}{2}\right)^{i-1} \mathbf{Y}_i = \mathbf{X}_k \tag{5.20}$$

where $\mathbf{Y}_i \in se(3)$ are solved for by inversion:

$$\mathbf{V}_{ij} = \left(t_i - \tfrac{1}{2}\right)^{j-1}, \quad \mathbf{Y}_i = \sum_{j=1}^{\nu} \left(\mathbf{V}^{-1}\right)_{ij} \mathbf{X}_j \tag{5.21}$$

This leads to a quadrature rule for an order four expansion:

$$\mathbf{\Psi}^{[4]}(h) = \mathbf{Y}_1 + \frac{1}{12}[\mathbf{Y}_1, \mathbf{Y}_2] \tag{5.22}$$

and the following quadrature rule for a sixth order expansion:

$$\begin{aligned}\mathbf{\Psi}^{[6]} = \mathbf{Y}_1 + \frac{1}{12}\mathbf{Y}_3 + \frac{1}{12}[\mathbf{Y}_1, \mathbf{Y}_2] - \frac{1}{240}[\mathbf{Y}_2, \mathbf{Y}_3] + \frac{1}{360}[\mathbf{Y}_1, [\mathbf{Y}_1, \mathbf{Y}_3]\\ - \frac{1}{240}[\mathbf{Y}_2, [\mathbf{Y}_1, \mathbf{Y}_2]] - \frac{1}{720}[\mathbf{Y}_1, [\mathbf{Y}_1, [\mathbf{Y}_1, \mathbf{Y}_2]]]\end{aligned} \tag{5.23}$$

Note that in [155] the expansion is given for the form of $\mathbf{T}'(s) = \mathbf{X}(s)\mathbf{T}(s)$, which corresponds to twists expressed with respect to the world frame, but since we use the body twist in (5.4) there is a difference in sign for some terms.

We propose finding the poses at the collocation points $\mathbf{T}(c_i)$ using a Magnus expansion step between each collocation point. We assign either $\nu = 2$ or $\nu = 3$ quadrature points (for either fourth or sixth order expansions, respectively) between each pair of collocation points as well as between 0 and c_0 and between c_n and L. This leads to a total of $m = \nu(n + 2)$ quadrature points along the length of the rod. Fig. 5.3 shows an example of this for $\nu = 3$ and $n = 2$. We then compute a Magnus expansion between each collocation point, starting from $s = 0$ and stepping to $s = L$. The frame at each collocation point therefore given by a product of exponentials where $\boldsymbol{\Psi}_i$ is expressed in frame $\mathbf{T}(c_i)$:

$$\mathbf{T}(c_k) = \mathbf{T}_0 \prod_{i=0}^{k} e^{\boldsymbol{\Psi}_i} \tag{5.24}$$

Although not detailed here, we note that since the shape of the rod is given as a product of exponentials, closed-form gradients of the residual vector $\mathbf{e}$ can be found which facilitate faster solutions to (5.13). Furthermore, as shown in [39, 40] Jacobians useful for studying the kinematics of continuum manipulators can be found via (5.24).

5.5 Determining curvature at quadrature points

Evaluating $\boldsymbol{\Psi}_i$ according to the quadrature formula above ((5.22) or (5.23)) requires knowing the values at the quadrature points, $\tilde{\mathbf{u}}(q_k)$ where $q_k = c_i + t_k h$, $k = 1 \ldots \nu$. Given the collocation values $\tilde{\mathbf{u}}(c_i)$ (which are guessed at each iteration of a nonlinear solver), we show here that the function values of the interpolating polynomial at the quadrature points can be found directly as a linear combination of the collocation values. First, we define a matrix $\mathbf{U}_q$ that contains quadrature values,

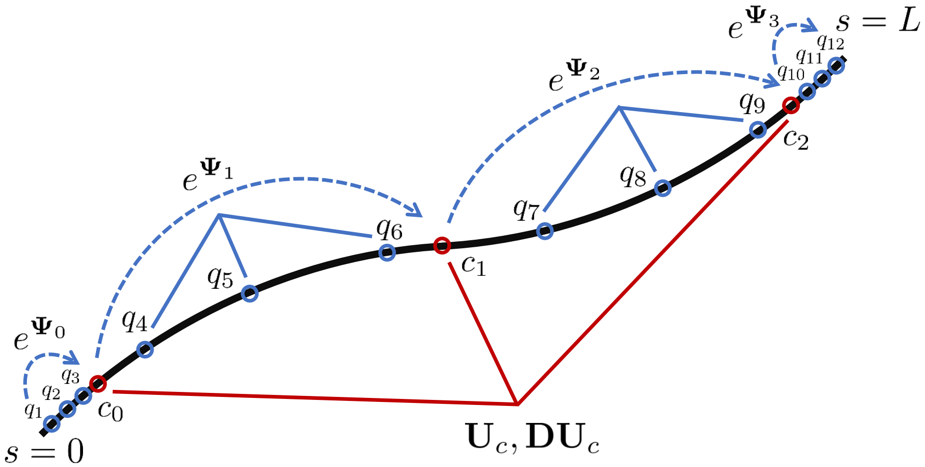


Figure 5.3: Collocation points and quadrature points for the case of $n = 2$ and $\nu = 3$. Using orthogonal polynomials and their zeros results in quadrature values being linearly related to the collocation values.

similar to $\mathbf{U}_c$:

$$\mathbf{U}_q = \begin{pmatrix} \mathbf{u}^{\mathrm{T}}(q_1) \\ \vdots \\ \mathbf{u}^{\mathrm{T}}(q_m) \end{pmatrix} \tag{5.25}$$

We then use (5.8) together with (5.9) to find a matrix relating the collocation values and the quadrature values:

$$\mathbf{U}_q = \mathbf{A}\mathbf{B}\mathbf{U}_c \tag{5.26}$$

where the matrix $\mathbf{B} \in \mathbb{R}^{(n+1)\times(n+1)}$ transforms the collocation values into modal coefficients which are then a transformed by $\mathbf{A} \in \mathbb{R}^{m\times(n+1)}$ into quadrature values:

$$\mathbf{A} = \begin{pmatrix} \frac{1}{2}T_0(q_1) & T_1(q_1) & \dots & T_n(q_1) \\ \vdots & \vdots & \vdots & \vdots \\ \frac{1}{2}T_0(q_m) & T_1(q_m) & \dots & T_n(q_m) \end{pmatrix} \tag{5.27}$$

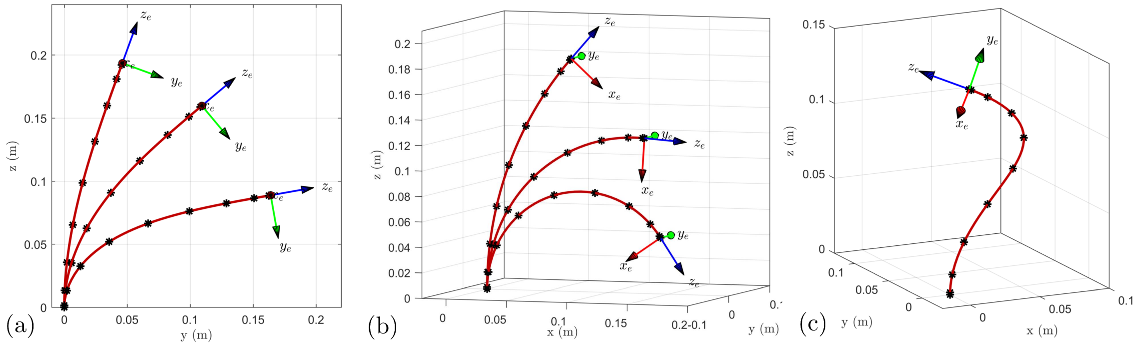


Figure 5.4: Examples comparing the shooting method and our collocation approach for (a) force loads (which we also validate with elliptic integrals) (b) moment loads which result in constant-curvature, and (c) combined force and moment.

$$\mathbf{B} = \frac{2}{n+1}\begin{pmatrix} T_0(c_0) & \dots & T_0(c_n) \\ \vdots & \vdots & \vdots \\ T_n(c_0) & \dots & T_n(c_n) \end{pmatrix} \tag{5.28}$$

This same procedure is also used to find $\mathbf{u}(L)$ which is necessary to compute the error residual for the boundary condition in (5.13). Note that both $\mathbf{A}$ and $\mathbf{B}$ can be computed offline as long as the collocation and quadrature points are chosen beforehand.

5.6 Convergence of the Magnus Expansion

One important note is that the infinite Magnus expansion is not guaranteed to converge. In this section, we use known sufficient conditions for convergence of the Magnus expansion to provide the maximum step sizes that will guarantee convergence. These bounds are not necessary conditions, so we are able to violate the bounds in our simulation results, but we discuss convergence here since it cannot in general be guaranteed. For real matrices, the expansion converges in the 2-norm provided that [176]:

$$\int_0^h \|\mathbf{X}(\xi)\|_2 \, \mathrm{d}\xi < \pi \tag{5.29}$$

Here we relate this bound on convergence to the case of the Cosserat rod and provide a bound on the maximum step size as a function of the maximum curvature. Recall that the Euclidian 2-norm is bounded by the Frobenius norm:

$$\|\mathbf{X}\|_2 \leqslant \|\mathbf{X}\|_F = \sqrt{\text{trace}(\mathbf{X}^{\mathrm{T}}\mathbf{X})} \tag{5.30}$$

Assume the curvatures $\mathbf{u}(s)$ are bounded by a known scalar, i.e. $u_x \leqslant u_y \leqslant u_z \leqslant \beta$. We can then compute the Frobenius norm explicitly and provide the following bound:

$$\|\mathbf{X}\|_F = \sqrt{2u_x^2 + 2u_y^2 + 2u_z^2 + 1} \leqslant \sqrt{6\beta^2 + 1} \tag{5.31}$$

We then integrate (5.31) to find a conservative bound for integration step h to guarantee convergence:

$$\int_0^h \|\mathbf{X}(\xi)\|_2 \, \mathrm{d}\xi \leqslant \int_0^h \|\mathbf{X}(\xi)\|_F \, \mathrm{d}\xi < \pi \tag{5.32}$$

$$h_{max} < \frac{\pi}{\sqrt{6\beta^2 + 1}} \tag{5.33}$$

Given a particular task and continuum robot architecture, the bound β might be known beforehand through a simulation study. Another option is to choose β by considering the strain limits of the rod material. Consider as an example a superelastic nickel-titanium (Nitinol) rod, which can accept a strain of 5% with minimal loss of superelasticity due to cyclic fatigue [177]. The maximum bending strain is $\epsilon = ur$, where u is the rod's curvature and r is the rod's radius. Assuming the continuum robot is designed to avoid violating strain limits, we have $u_x \leqslant u_y \leqslant u_z \leqslant \epsilon/r = \beta$.

Table 5.1 shows the maximum step size that will guarantee convergence of the Magnus expansion under the assumption $\epsilon < 5\%$. For larger rod diameters the step sizes are not restrictive, and for smaller rod diameters the particular task can potentially be taken into account to determine a more suitable bound β and allow

Table 5.1: Maximum Step Size for Guaranteed Convergence with 6% Bending Strain

r (mm)	β	h_{max} (mm)
1	50	25.65
2	25	51.29
3	16.67	76.92
4	12.5	102.54

Table 5.2: Step Sizes Used in Simulations (2 mm OD rod)

n	2	4	6	8	10
h_{max} (mm)	86.60	58.78	43.38	34.20	28.17

larger step sizes. Online checks may be necessary to check for convergence of the expansion in cases where larger step sizes are used. In our simulations, we used the step sizes in Table 5.2 without observing issues.

5.7 Simulation Results

In this section, we will compare the accuracy of the fourth and sixth-order Magnus expansions with different numbers of collocation points. In all examples we use a shooting method as a ground-truth to compare our results against, since the shooting method has been validated in previous experimental studies [47, 46, 48]. We integrated (5.4) with a Runge-Kutta solver and a Levenberg-Marquardt algorithm (*ode45*() and *fsolve*() in MATLAB) to satisfy the boundary conditions with a termination tolerance of 10^{-9}. We used units of meters and radians. In the planar examples with in-plane forces, we also compared against the elliptic integral solution given in [178] and experimentally validated in [179, 180]. For all examples we simulated the case of a solid nickel-titanium (Nitinol) rod with a 2 mm diameter and a 200 mm length.

We first consider the planar case with an in-plane load. We solved the elliptic integral equations in [178] with tip angles of 20°, 50°, and 80°, giving the position of the tip and resultant tip forces of $\mathbf{f}_e = [0, 1.04, 0.104]^{\mathrm{T}}$ N, $\mathbf{f}_e = [0, 3.63, 0.362]^{\mathrm{T}}$ N, and $\mathbf{f}_e = [0, 18.9, 1.89]^{\mathrm{T}}$ N. We then used these forces in the shooting method and our collocation method (with a sixth order Magnus expansion and $n = 10$), both of which are shown in Fig. 5.4(a). All three methods showed agreement within 0.006 mm (0.003% of arc length).

Next we consider a planar case with an in-plane moment load, as shown in Fig. 5.4(b). In this case, our collocation method returns the predicted constant-curvature shape, and for all three cases shown in Fig. 5.4(b) the shooting and collocation method (with a sixth order Magnus expansion and $n = 10$) showed agreement within 3.98e-6 mm and rotation error within machine precision.

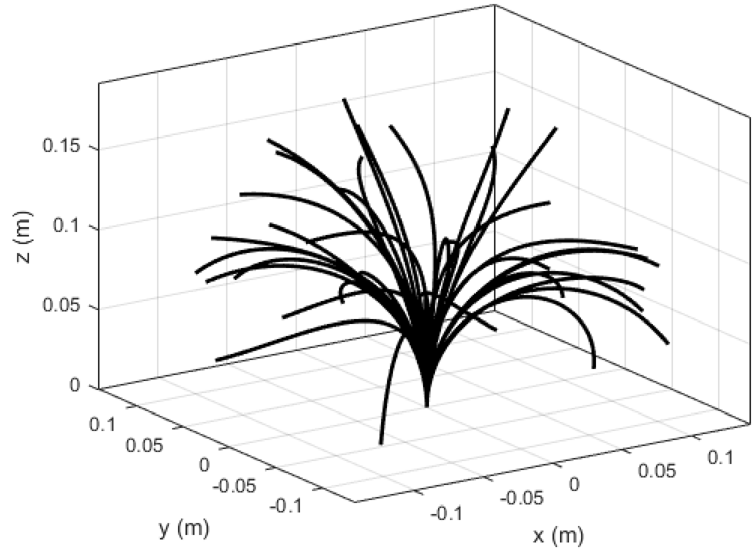


Figure 5.5: Samples from the set of 729 rod shapes used when comparing our approach to the shooting method.

The collocation method also agrees with the shooting method in cases with combined forces and moments, as shown in Fig. 5.4(c). In the next study, we compare the shooting and collocation methods for combinations of forces/moments and compare different numbers of collocation points. We simulated combinations of ±1 N tip forces and ±0.5 Nm moments applied to the tip of the rod (a total of 729 applied wrench cases). The solution speed of any BVP solver is affected by how close initial guesses are to the solution, so for each case, we begin the simulation with the rod in its straight configuration and linearly interpolate from a zero wrench to the final wrench in 3 steps. This is to more realistically simulate a scenario where the model is being continuously solved (e.g. kinematic control) and the previous solution is used as the initial guess for the solver. With the interpolated wrenches the total number of applied wrench cases is 2,187, samples of which are shown in Fig. 5.5. We solved for the equilibrium shape of the rod for each case with different numbers of collocation points, $n \in [2, 4, 6, 8, 10]$. We also solved each case using the fourth-order Magnus expansion ($\nu = 2$) and the sixth-order Magnus expansion ($\nu = 3$). The tip pose found with each collocation method was compared to the tip pose found with the shooting method using the following metrics:

$$\begin{aligned} e_p &= \frac{\|\mathbf{p}_c(L) - \mathbf{p}_s(L)\|}{L} \times 100 \\ e_r &= \cos^{-1}\left(\frac{\text{trace}\left(\mathbf{R}_s(L)\mathbf{R}_c^{\mathrm{T}}(L)\right) - 1}{2}\right) \end{aligned} \tag{5.34}$$

where we report the position error as percent of total arc length, $\mathbf{p}_c(L), \mathbf{R}_c(L)$ are the tip position/rotation found using a collocation method, and $\mathbf{p}_s(L), \mathbf{R}_s(L)$ are the tip poses found using the shooting method.

Tables 5.3 and 5.4 show the results of the simulations. The difference between the shooting method and our method rapidly converges to zero with increasing n. Both the fourth-order and sixth-order expansions provided rapid convergence with increasing n, and for $n \leqslant 6$ are comparable. However, for $n \geqslant 8$ sixth-order Magnus steps did a better job reducing the e_p and e_r to below 0.005%. This implies that for small n, the primary source of the error is in the interpolation error in collocation (and not the Magnus stepping), while for larger n the error in the Magnus steps becomes more important. Any method with $n \geqslant 6$ was able to calculate the tip position with agreement with shooting of less than 0.15% of arc length.

We also report in Tables 5.3 and 5.4 the average speed at which solutions were found across the set of 2,187 shapes, running MATLAB 2019b on an i7-4770 3.4 GHz CPU. Both tables show the results for the case where $L = 200$ mm. The fourth order and sixth order Magnus expansions were comparable in terms of speed, since the primary source of computation effort is not in evaluating the Magnus expansion but in computing the gradient of the error residual in (5.13), which despite being available in closed-form, becomes increasingly expensive as n increases. When n is not too large, the gradient is cheap to compute which provides quick solutions when combined with the efficient integration of $\mathbf{T}'(s)$ via the Magnus expansion.

Table 5.3: Fourth Order Magnus Tip Error as a Function of Collocation Polynomial Order

	Pos. e_p (%)		Rot. e_r (deg)		
	Avg.	Max	Avg.	Max.	Speed (Hz)
$n = 2$	2.97	28.0	4.28	36.3	179.6
$n = 4$	0.141	2.15	0.235	3.78	112.1
$n = 6$	0.00573	0.147	0.00889	0.183	71.6
$n = 8$	0.00122	0.0173	0.00453	0.0571	46.3
$n = 10$	5.46e-4	0.00707	0.00448	0.0543	33.1

Table 5.4: Sixth Order Magnus Tip Error as a Function of Collocation Polynomial Order

	Pos. e_p (%)		Rot. e_r (deg)		
	Avg.	Max.	Avg.	Max.	Speed (Hz)
$n = 2$	3.00	28.1	4.29	36.5	176.8
$n = 4$	0.140	2.26	0.234	3.79	106.2
$n = 6$	0.00467	0.115	0.00889	0.193	68.8
$n = 8$	1.95e-4	0.00493	0.00450	0.0553	42.5
$n = 10$	2.66e-5	0.00140	0.00448	0.0542	32.4

Our shooting method implementation in MATLAB solved the BVPs at an average rate of 17.6 Hz (with residual gradients estimated via finite differences). Our collocation implementation was faster than the shooting method in all examples, but we would like to stress that a fair comparison between the shooting and collocation is difficult since either method could be improved with more specialized implementations in pre-compiled code and by providing good initial guesses (see [46, 43] for shooting method implementations with >1kHz). The main takeaway from these results is that our method can be competitive in terms of computation speed, and due to the availability of analytic expressions, may improve speed in cases where lower-order polynomials are sufficient. More importantly, the Magnus expansion leading to analytic direct kinematics offers fast evaluation of instantaneous direct kinematics Jacobians.

5.8 Conclusions

In this chapter, we presented a numerical approach to solving the Cosserat rod BVP that combines orthogonal collocation and the Magnus expansion which, when solved, results in a analytic product of matrix exponentials equation. We have discussed the convergence of the Magnus expansion for the case of Cosserat rods and showed in simulation that both the fourth order and sixth order Magnus expansions provide accurate solutions to the BVP and that a small number of collocation points can provide reasonably accurate results. We believe these preliminary results demonstrate the potential of this approach to provide reduced computational cost when solving Cosserat rod models. In Chapter 6, we will use this formulation as a starting point to derive analytic expressions for the compliance matrix of Cosserat rods and tendon-actuated robots modeled as Cosserat rods.

CHAPTER 6

MODAL SHAPE LIE GROUP METHODS FOR COMPUTING CONTINUUM ROBOT COMPLIANCE MATRICES

In this chapter, we consider how to compute the passive compliance matrix (or equivalently, the inverse of the stiffness matrix) of continuum robots modeled as Kirchhoff rods (i.e. Cosserat rods with negligible shear strains and extension). This problem is important for several design, modeling, planning, and control problems. In design and modeling, the compliance of a continuum robot is important to consider because it affects the ability to apply external loads, dynamic behavior, and local kinematic uncertainty. In planning and control, the compliance matrix needs to be computed for online stiffness modulation [53, 76, 77] and stiffness control methods [58, 50] which are useful for adjusting dynamic performance online and for tasks like mechanical assembly of parts, estimating forces applied to the environment when bracing, and safe human-robot interaction. A specific example where the compliance matrix would be needed is to apply the stiffness modulation method from Chapter 2 to the kinematically-redundant ISCR system presented in Chapter 3.

Note that we make a distinction here between the local compliance matrix and a deflection model. The statics model discussed in Chapter 5 provides the deflected shape of a Cosserat rod given a force applied to its tip. Colloquially, a robot with "lower compliance" would have a smaller deflection from its unloaded configuration for a given external force. This notion of compliance/stiffness is used in [146] for designing tendon-routing to improve load-carrying capacity and in [181] for stiffness control of a concentric tube robot. This notion of compliance is important, but here we are concerned not with the global nonlinear deflected shape of the rod, but with the local linearized compliance behavior, i.e. the compliance matrix around a given equilibrium configuration. This compliance matrix is used to predict the future deflection/force behavior of the robot around its current configuration when the applied force is incrementally changed. With this model we can therefore choose what the local deflection/force behavior will be, either at the design stage or online using the compliance modulation method presented in Chapter 2.

There are a variety of ways in which the compliance matrix can be defined, depending on which parameters are used and what frame the parameters are written in. We will define the *configuration space compliance* as the relationship between forces projected into the configuration space and the resulting changes in the configuration variables. This notion of compliance was described in [50] for the case of constant curvature robots. We define the *task space compliance* as the more conventional notion, which provides the twist that a robot experiences as a result of an applied wrench. The task space compliance can be further categorized depending on what reference frame is used to write the twists and wrenches. Here, when deriving the task space compliance, we will write the twists and wrenches in a hybrid frame that is coincident with the end-disk frame but aligned with the world frame. We make this choice because the statics model from Chapter 5 that we use to validate against is more conveniently solved using the hybrid frame, but we note that the derivations below follow using spatial or body frame twists/wrenches.

A straightforward way to compute the compliance matrix is via finite differences. Given an equilibrium configuration, a small change in one element of the applied wrench can be made and then the statics model can be recomputed to find an estimate of the resultant twist motion due to that change in wrench. This can be repeated to compute the compliance matrix column-by-column. This method, however, requires the mechanics model to be computed six times at each configuration, and does not provide analytic expressions for the compliance.

An approach based on propagation of particular partial derivatives was given in [51] and applied to a concentric-tube robot. In this method, an additional set of differential equations are integrated together with the standard Cosserat rod equations to provide the compliance matrix. It was shown in [25] that a similar approach can be combined with finite difference steps to compute the compliance matrix of a parallel continuum robot. In the mechanics literature, variational formulations are used to derive stiffness matrices for geometrically exact models [182, 183], but these methods have not been translated into a robotics context for the problems we study here. In [50], a configuration-space stiffness matrix is derived for a multi-backbone continuum robot using a constant-curvature assumption and applied for stiffness control. Finally, [184] presented the task-space compliance of a continuum segment with discrete actuators applying moment loads along the segment's body. This work used modal basis functions to describe the backbone bending angle, resulting in an analytic expression for the task-space compliance, but it was limited to planar deflections. A gap in the literature is analytic expressions for the compliance matrices of continuum robots experiencing spatial deflections that can 1) be used for both low-order and high-order models and 2) provide both the configuration and task space compliance matrices.

We therefore seek an approach for computing the compliance matrix that leverages the Lie group kinematic formulation presented in Chapter 5 to provide analytic expressions that do not rely on either finite differences or integration of a differential equation. By writing the compliance matrix in this modeling framework, we can write both task space and configuration space compliance matrices, bridging the gap between constant-curvature models and higher order Cosserat rod models. This allows practitioners to more easily make tradeoffs between using the full task space compliance or the simpler configuration space compliance, as well as choose the order/accuracy of the model by changing the modal shape basis used (e.g. constant curvature vs variable curvature). We also believe that our approach could also potentially reduce computation cost by avoiding finite differences and numerical integration of differential equations if additional methods from the geometric integration literature were applied to reduce the number of Lie bracket operations [155].

We will first derive the compliance matrix of a single Kirchhoff rod (i.e. a Cosserat rod with shear strains neglected) with a known tip wrench using a virtual work argument and validate the result by comparing to the compliance matrix computed using finite differences and the Kirchhoff rod model presented in Chapter 5. We then extend the model to the case of a tendon-actuated continuum robot.

6.1 Compliance Matrix of a Single Rod via Virtual Work

The following derivation follows a similar set of steps as the derivation of the stiffness matrix for a parallel robot in Chapter 2 and the statics [167] and stiffness of multi-backbone robots [50]. We begin by defining a small perturbation (i.e. a twist) in the end effector pose $\mathbf{T}(L)$, denoted as $\delta\mathbf{x}_h$, which we find by computing the body twist and then transforming into the hybrid frame that is coincident with the body frame but aligned with the world frame:

$$\begin{aligned} \delta\mathbf{x}_h &= {}^h\mathbf{S}_b \left(\mathbf{T}^{-1}(L)\delta\mathbf{T}(L)\right)^\vee \in \mathrm{I\!R}^6 \\ {}^h\mathbf{S}_b &= \begin{bmatrix} {}^w\mathbf{R}_b & \mathbf{0}_{3x3} \\ \mathbf{0}_{3x3} & {}^w\mathbf{R}_b \end{bmatrix} \in \mathrm{I\!R}^{6\times 6} \end{aligned} \tag{6.1}$$

where ${}^w\mathbf{R}_b \in SO(3)$ is the rotation matrix of the body frame expressed in the world frame. We then define the task-space compliance matrix of a single rod as:

$$\delta\mathbf{x}_h = \mathbf{C}_{x_{rod}}\delta\mathbf{w}_h, \quad \mathbf{C}_{x_{rod}} \in \mathrm{I\!R}^{6\times 6} \tag{6.2}$$

where $\delta\mathbf{w}_h \in \mathrm{I\!R}^6$ is a small change in the applied wrench written in the hybrid frame and with the moment followed by the force, i.e. $\mathbf{w}_h = [\mathbf{m}_h^{\mathrm{T}}, \mathbf{f}_h^{\mathrm{T}}]^{\mathrm{T}}$.

Consider a single rod with a wrench applied to its tip. We assume here the rod is massless, and that shear strains and extension are negligible. Using the notation from Chapter 4 and 5, we denote $\mathbf{u}(s) \in \mathrm{I\!R}^3$ as the curvature distribution expressed a function of arc length, $\mathbf{K}_{bt} \in \mathrm{I\!R}^{3\times3}$ as the diagonal bending/torsion stiffness matrix of the rod's cross section, and $\mathbf{\Phi}(s) \in \mathrm{I\!R}^{3\times n}$ as the n^{th} order modal shape basis. The bending energy in the rod is given by

$$E = \int_0^L \frac{1}{2}\mathbf{u}(s)^{\mathrm{T}}\mathbf{K}_{bt}\mathbf{u}(s)\,\mathrm{d}s = \frac{1}{2}\mathbf{c}^{\mathrm{T}} \underbrace{\left(\int_0^L \mathbf{\Phi}^{\mathrm{T}}\mathbf{K}_{bt}\mathbf{\Phi}\,\mathrm{d}s\right)}_{\mathbf{\Phi}_k} \mathbf{c} \tag{6.3}$$

where we have substituted the modal shape basis curvature defined in Chapter 4, i.e. $\mathbf{u}(s) = \mathbf{\Phi}(s)\mathbf{c}$ from (4.5). Note that the matrix $\mathbf{\Phi}_k$ can be computed offline if the modal shape functions are chosen *a priori*. First, we write an equation for static equilibrium of the rod. For the work done by the applied wrench as it produces a small displacement $\delta\mathbf{x}$, there is a corresponding small change in bending energy δE:

$$\mathbf{w}_h^{\mathrm{T}}\delta\mathbf{x}_h = \delta E \tag{6.4}$$

Recalling the body Jacobian from (4.20), we denote the relationship between $\delta\mathbf{x}_h$ and $\delta\mathbf{c}$ as $\delta\mathbf{x}_h = {}^h\mathbf{S}_b\mathbf{J}_{\xi c}\delta\mathbf{c} = \widetilde{\mathbf{J}}_{\xi c}\delta\mathbf{c}$ and substitute this into (6.4):

$$\mathbf{w}_h^{\mathrm{T}}\widetilde{\mathbf{J}}_{\xi c}\delta\mathbf{c} = \left(\frac{\partial E}{\partial \mathbf{c}}\right)^{\mathrm{T}}\delta\mathbf{c} \tag{6.5}$$

By the principle of virtual work, to be in static equilibrium we require the virtual displacements associated with $\delta\mathbf{c}$ to vanish, resulting in the following equation of static equilibrium for the beam:

$$\widetilde{\mathbf{J}}_{\xi c}^{\mathrm{T}}\mathbf{w}_h = \frac{\partial E}{\partial \mathbf{c}} \tag{6.6}$$

Denoting c_i as the i^{th} modal coefficient (also the i^{th} element of $\mathbf{c}$) and taking small perturbations about the equilibrium configuration:

$$\delta\left(\widetilde{\mathbf{J}}_{\xi c}^{\mathrm{T}}\right)\mathbf{w}_h + \widetilde{\mathbf{J}}_{\xi c}^{\mathrm{T}}\delta\mathbf{w}_h = \delta\left(\frac{\partial E}{\partial \mathbf{c}}\right) \tag{6.7}$$

Using $\delta\left(\widetilde{\mathbf{J}}_{\xi c}^{\mathrm{T}}\right) = \sum_{i=1}^{n}\left[\frac{\partial \widetilde{\mathbf{J}}_{\xi c}^{\mathrm{T}}}{\partial c_i}\right]\delta c_i$ and $\delta\left(\frac{\partial E}{\partial \mathbf{c}}\right) = \left[\frac{\partial^2 E}{\partial \mathbf{c}^2}\right]\delta\mathbf{c}$, we obtain:

$$\left[\frac{\partial \widetilde{\mathbf{J}}_{\xi c}^{\mathrm{T}}}{\partial c_1}\mathbf{w}_h \ldots \frac{\partial \widetilde{\mathbf{J}}_{\xi c}^{\mathrm{T}}}{\partial c_n}\mathbf{w}_h\right]\delta\mathbf{c} + \widetilde{\mathbf{J}}_{\xi c}^{\mathrm{T}}\delta\mathbf{w}_h = \frac{\partial^2 E}{\partial \mathbf{c}^2}\delta\mathbf{c} \tag{6.8}$$

Solving for $\delta\mathbf{c}$ gives:

$$\delta\mathbf{c} = \left(\frac{\partial^2 E}{\partial \mathbf{c}^2} - \left[\frac{\partial \widetilde{\mathbf{J}}_{\xi c}^{\mathrm{T}}}{\partial c_1}\mathbf{w}_h \ldots \frac{\partial \widetilde{\mathbf{J}}_{\xi c}^{\mathrm{T}}}{\partial c_n}\mathbf{w}_h\right]\right)^{-1}\widetilde{\mathbf{J}}_{\xi c}^{\mathrm{T}}\delta\mathbf{w}_h \tag{6.9}$$

Recalling the basic definition of the task-space compliance $\mathbf{C}_{x_{rod}} = \frac{\delta\mathbf{x}_h}{\delta\mathbf{w}_h}$ and substituting the above result for $\delta\mathbf{c}$ in $\delta\mathbf{x}_h = \widetilde{\mathbf{J}}_{\xi c}\delta\mathbf{c}$ results in the analytic expression for the task-space compliance matrix:

$$\mathbf{C}_{x_{rod}} = \widetilde{\mathbf{J}}_{\xi c}\left(\frac{\partial^2 E}{\partial \mathbf{c}^2} - \left[\frac{\partial \widetilde{\mathbf{J}}_{\xi c}^{\mathrm{T}}}{\partial c_1}\mathbf{w}_h \ldots \frac{\partial \widetilde{\mathbf{J}}_{\xi c}^{\mathrm{T}}}{\partial c_n}\mathbf{w}_h\right]\right)^{-1}\widetilde{\mathbf{J}}_{\xi c}^{\mathrm{T}} \tag{6.10}$$

The above result, matches with the congruence transformation of stiffness as discussed in [10] for serial robots.

In the result above, the energy Hessian $\frac{\partial^2 E}{\partial \mathbf{c}^2}$, which can be computed offline, is found by differentiating (6.3) twice:

$$\frac{\partial E}{\partial \mathbf{c}} = \frac{1}{2}\left(\mathbf{\Phi}_k + \mathbf{\Phi}_k^{\mathrm{T}}\right)\mathbf{c} \qquad \Rightarrow \qquad \frac{\partial^2 E}{\partial \mathbf{c}^2} = \frac{1}{2}\left(\mathbf{\Phi}_k + \mathbf{\Phi}_k^{\mathrm{T}}\right) \tag{6.11}$$

The task-space Jacobian was provided in Chapter 4, and finite differences can be used to estimate the partial derivatives of $\widetilde{\mathbf{J}}_{\xi c}$. It is also likely possible to tediously derive these term-by-term from the Magnus expansion. This would result in a recursive expression similar to the partial derivatives of the product of matrix exponentials in (4.22).

6.1.1 Simulation Validation of Rod Compliance

We now validate the compliance matrix expression presented above by comparing to the Kirchhoff rod model presented in Chapter 5. We used 10 collocation points to simulate the Kirchhoff rod model, and we estimated the compliance matrix of the Kirchhoff rod model using finite differences. We then compared these compliance matrices to the analytic compliance matrix in (6.10).

We computed the compliance matrix for each method using the same rod geometry and loading conditions used in 5.7, i.e. a 2 mm diameter rod with a length of 200 mm and combinations of ± 1N tip forces and ±0.5 Nm tip wrenches. This generated a set of 2187 shapes for which we computed the compliance matrix. We also computed the compliance matrix in (6.10) with different numbers of modal coefficients in the modal shape basis $\mathbf{\Phi}(s)$. We kept the number of modal coefficients for the x, y, and z directions the same for each simulation, but varied the number of modal coefficients along all directions from 1 to 11. In other words, when $n = 1$, there were three columns in the modal shape basis, and when $n = 11$, there were 33 columns in the modal shape basis. Note that although we varied the number of modal coefficients in $\mathbf{\Phi}(s)$, we always used 10 steps when integrating with the Magnus expansion when computing $\mathbf{T}(L)$ to isolate the effect of choice in modal shape basis on the error in the compliance matrix.

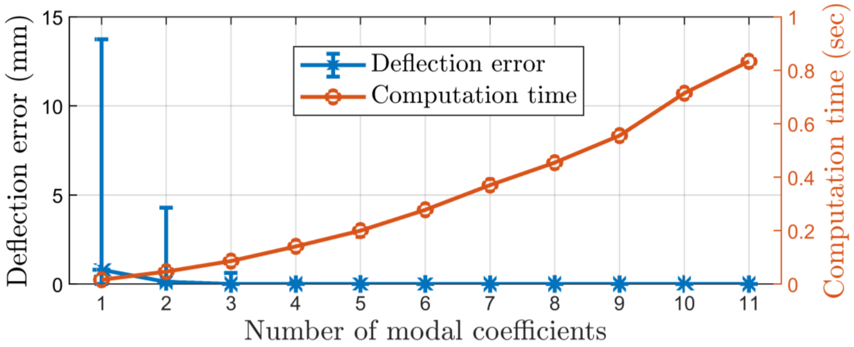


Figure 6.1: The tip deflection error with max/min error bars when comparing the analytic and finite difference compliance matrices for different numbers of modal coefficients and the computation time for the analytic compliance matrix, showing a trade-off between computation time and the accuracy.

Table 6.1: Tip Deflection Error versus Number of Modal Coefficients

	Position error (mm)		Rotation error (deg)		
	Avg.	Max.	Avg.	Max.	Speed (Hz)
$n = 1$	0.8	13.7	0.2	6.3	62.0
$n = 3$	1.3e-2	0.6	2.2e-3	0.10	11.6
$n = 5$	6.7e-5	6.3e-3	2.0e-5	1.2e-3	5.0
$n = 7$	1.5e-5	3.8e-4	1.3e-5	4.4e-4	2.7

We compared the two compliance matrices, one being the analytic compliance in (6.10) and the other computed with finite differences, by applying forces of 0.1 N and moments of 0.05 Nm (10% of the maximum load simulated) along all three axes, i.e. 6 different wrenches for each compliance matrix. We then compared the tip positional and rotational deflections predicted by the analytic expression to the finite differences result. The mean/max absolute deflection error results are shown in Table 6.1 and Figure 6.1. We see that as n (number of modal coefficients) is increased, the analytic expression rapidly converges to the finite difference estimate. We also observe that the computation time increases as n is increased, showing a tradeoff between the accuracy of the model and the computation cost.

The primary source of computational cost in computing $\mathbf{C}_{x_{rod}}$ is in computing the derivatives of the task-space Jacobian. For $n = 11$, estimating these derivatives accounts for approximately 95% of the computation time. We are currently using finite differences to estimate these derivatives, but we believe that analytic derivatives could be derived, which would remove the need for these finite difference estimates. It was also shown in [155] that the number of Lie brackets required to compute the Magnus expansion can be reduced using a theory of free Lie algebras. Our current implementation is not using these techniques, but applying them would significantly reduce the number of Lie brackets needed to integrate with the Magnus expansion and to compute the task space Jacobian.

One may ask whether it is possible to neglect the term that includes the derivatives of the task-space Jacobian. While this term is typically negligible for rigid-link robots [185], for compliant robots this term is no longer negligible, as shown in [67]. We show this here as well by computing the compliance matrices while neglecting this term and applying the same forces as above to compare to the finite differences compliance matrix. The mean absolute position error and error bars are shown in Figure 6.2. We see that regardless of the number of modal coefficients used, the error in the compliance matrix is large when neglecting the term with the derivatives of the task-space Jacobian.

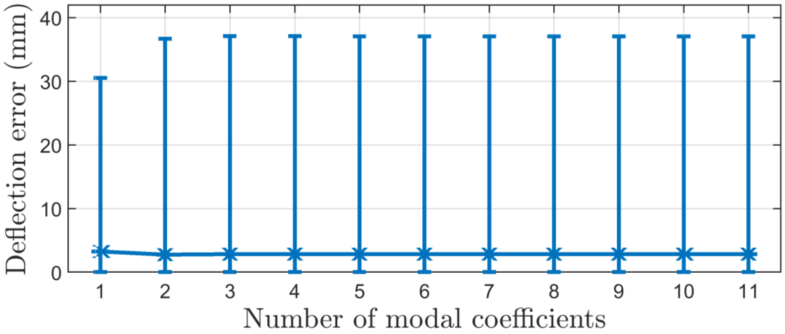


Figure 6.2: The tip deflection error when neglecting the term with the external wrench and the derivatives of the task-space Jacobian, showing that the compliance matrix error is large without this term.

6.2 Task-space Compliance Matrix of a Tendon-actuated Continuum Segment

We now extend the example above for a single rod to the case of a tendon-actuated continuum segment as the one described in Section 3. We first derive a statics expression for a variable curvature segment similar to [163], and then take partial derivatives of this statics expression to arrive at the compliance matrix in an analytic form as a result of our Lie group formulation.

For a given external wrench $\mathbf{w}_h$ on the tip of the segment and a perturbation $\delta\mathbf{x}_h$ in the end pose, we have a corresponding change in the forces applied to the actuation tendons $\boldsymbol{\tau}$ and a corresponding change in the bending energy stored in the segment, where the bending energy is given by 6.3. Following the principle of virtual work, we have:

$$\mathbf{w}_h^{\mathrm{T}}\delta\mathbf{x}_h + \boldsymbol{\tau}^{\mathrm{T}}\delta\boldsymbol{\ell} - \delta E = 0 \tag{6.12}$$

where $\delta\boldsymbol{\ell}$ is a change in the tendon length. Referring to (4.20) and (4.18), we substitute $\delta\mathbf{x}_h = {}^{h}\mathbf{S}_b\mathbf{J}_{\xi c}\delta\mathbf{c} = \widetilde{\mathbf{J}}_{\xi c}\delta\mathbf{c}$, $\delta\boldsymbol{\ell} = \mathbf{J}_{\ell c}\delta\mathbf{c}$, and $\delta E = \frac{\partial E}{\partial \mathbf{c}}^{\mathrm{T}}\delta\mathbf{c}$ to arrive at the statics of the segment about a given configuration:

$$\mathbf{J}_{\ell c}^{\mathrm{T}}\boldsymbol{\tau} = \frac{\partial E}{\partial \mathbf{c}} - \widetilde{\mathbf{J}}_{\xi c}^{\mathrm{T}}\mathbf{w}_h \tag{6.13}$$

We then take small perturbations of this statics expression:

$$\mathbf{C}_{\tau}\delta\mathbf{c} + \mathbf{J}_{\ell c}^{\mathrm{T}}\delta\boldsymbol{\tau} = \frac{\partial^2 E}{\partial \mathbf{c}^2}\delta\mathbf{c} - \mathbf{C}_{w_h}\delta\mathbf{c} - \widetilde{\mathbf{J}}_{\xi c}^{\mathrm{T}}\delta\mathbf{w}_h \tag{6.14}$$

where $\mathbf{C}_{\tau}$ and $\mathbf{C}_{w_h}$ are defined as:

$$\mathbf{C}_{\tau} = \left[\frac{\partial \mathbf{J}_{\ell c}^{\mathrm{T}}}{\partial c_1}\boldsymbol{\tau}, \ldots, \frac{\partial \mathbf{J}_{\ell c}^{\mathrm{T}}}{\partial c_n}\boldsymbol{\tau}\right] \in \mathrm{I\!R}^{n\times n} \tag{6.15}$$

$$\mathbf{C}_{w_h} = \left[\frac{\partial \widetilde{\mathbf{J}}_{\xi c}^{\mathrm{T}}}{\partial c_1}\mathbf{w}_h, \ldots, \frac{\partial \widetilde{\mathbf{J}}_{\xi c}^{\mathrm{T}}}{\partial c_n}\mathbf{w}_h \right] \in \mathrm{I\!R}^{n \times n} \tag{6.16}$$

The matrix $\mathbf{C}_\tau$ is the contribution to the compliance matrix of the forces on the tendons at the current configuration, and $\mathbf{C}_{w_h}$ is the contribution due to the external wrench. $\mathbf{C}_\tau$ can be readily affected by using actuation redundancy (internal preload) and $\mathbf{C}_{w_h}$ depends only on the external load. In a robot without actuation redundancy, $\boldsymbol{\tau}$ is determined by $\mathbf{w}_h$ through the statics equation. Therefore $\mathbf{C}_\tau$ and $\mathbf{C}_{w_h}$ are not completely independent. In a robot with actuation redundancy, these two matrices are independent. The above result also bears resemblance to the results of prior works on stiffness modulation of parallel robots where one defines an active stiffness term dependent on the joint-level forces and the derivative of the Jacobian and a passive stiffness term dependent on joint-level stiffness [89, 9, 11].

Now, let us define the *joint-level stiffness* as $\mathbf{K}_\ell = \frac{\delta \boldsymbol{\tau}}{\delta \boldsymbol{\ell}}$, a diagonal matrix in which each element along the diagonal contains the stiffness of an individual actuation tendon:

$$\mathbf{K}_\ell = \begin{bmatrix} \frac{\delta \tau_1}{\delta \ell_1} & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & \frac{\delta \tau_n}{\delta \ell_n} \end{bmatrix} \tag{6.17}$$

We substitute $\delta \boldsymbol{\ell} = \mathbf{J}_{\ell c}\delta \mathbf{c}$ into $\delta \boldsymbol{\tau} = \mathbf{K}_\ell \delta \boldsymbol{\ell}$ and then substitute into (6.14), combine like terms, and solve for $\delta \mathbf{c}$:

$$\delta \mathbf{c} = \left(\frac{\partial^2 E}{\partial \mathbf{c}^2} - \mathbf{C}_\tau - \mathbf{C}_{w_h} - \mathbf{J}_{\ell c}^{\mathrm{T}} \mathbf{K}_\ell \mathbf{J}_{\ell c} \right)^{-1} \widetilde{\mathbf{J}}_{\xi c}^{\mathrm{T}} \delta \mathbf{w}_h \tag{6.18}$$

We then substitute this expression into $\delta \mathbf{x}_h = \widetilde{\mathbf{J}}_{\xi c} \delta \mathbf{c}$ to arrive at the compliance matrix:

$$\mathbf{C}_x = \frac{\delta \mathbf{x}_h}{\delta \mathbf{w}_h} = \widetilde{\mathbf{J}}_{\xi c} \left(\frac{\partial^2 E}{\partial \mathbf{c}^2} - \mathbf{C}_\tau - \mathbf{C}_{w_h} - \mathbf{J}_{\ell c}^{\mathrm{T}} \mathbf{K}_\ell \mathbf{J}_{\ell c} \right)^{-1} \widetilde{\mathbf{J}}_{\xi c}^{\mathrm{T}} \tag{6.19}$$

We note the similarity of this matrix to the stiffness matrix in (2.10) used for the stiffness modulation results in Chapter 2. Similar to (2.10), this compliance matrix includes a diagonal joint-level stiffness matrix and a term with the external wrench multiplied by the derivatives of the task-space Jacobian. Additional terms include the Hessian of the bending energy and a term with the joint forces multiplied by the derivatives of the configuration-space Jacobian. These terms do not arise for the parallel robot in (2.10) because internal energy storage is negligible there, and there is no configuration-space Jacobian for that robot.

6.3 Configuration-space Compliance Matrix of a Tendon-actuated Continuum Segment

We now follow a similar set of steps as above to derive the configuration-space compliance matrix for a tendon-actuated continuum segment. The resulting expression does not require computation of the task-space Jacobian and its derivatives, and is therefore more computationally efficient to compute. Furthermore, this notion of compliance does not require knowledge of the external wrench, as required by the task-space compliance. Our derivation is similar to what was presented in [50], where a similar configuration-space stiffness matrix was defined for a constant-curvature continuum segment and used for compliant motion control. The compliance matrix we present here is an extension of the compliance matrix in [50] to the case of variable curvature continuum robots, and we believe can be used to implement similar compliant motion controllers as in [50] but for cases where external loads are large enough to create variable curvature shapes in the segment.

We first define the external wrench projected into the configuration space, which we denote as $\mathbf{w}_c$. Referring to (6.13), this projected wrench is given as:

$$\mathbf{w}_c = \widetilde{\mathbf{J}}_{\xi c}^{\mathrm{T}} \mathbf{w}_h = \frac{\partial E}{\partial \mathbf{c}} - \mathbf{J}_{\ell c}^{\mathrm{T}} \boldsymbol{\tau} \tag{6.20}$$

Taking small perturbations of this projected wrench results in

$$\delta \mathbf{w}_c = \frac{\partial^2 E}{\partial \mathbf{c}^2}\delta \mathbf{c} - \mathbf{C}_\tau \delta \mathbf{c} - \mathbf{J}_{\ell c}^{\mathrm{T}} \delta \boldsymbol{\tau} \tag{6.21}$$

where $\mathbf{C}_\tau$ was given in (6.14). We now substitute $\delta\boldsymbol{\tau} = \mathbf{K}_\ell \mathbf{J}_{\ell c} \delta \mathbf{c}$, combine like terms, and solve for $\delta \mathbf{c}$ (as done above for the task-space compliance):

$$\delta \mathbf{c} = \left(\frac{\partial^2 E}{\partial \mathbf{c}^2} - \mathbf{C}_\tau - \mathbf{J}_{\ell c}^{\mathrm{T}} \mathbf{K}_\ell \mathbf{J}_{\ell c} \right)^{-1} \delta \mathbf{w}_c \tag{6.22}$$

We define the configuration space compliance as the change in the configuration for a small change in the projected wrench, i.e. $\delta \mathbf{c} = \mathbf{C}_c \delta \mathbf{w}_c$. This gives us the configuration-space compliance matrix as:

$$\mathbf{C}_c = \left(\frac{\partial^2 E}{\partial \mathbf{c}^2} - \mathbf{C}_\tau - \mathbf{J}_{\ell c}^{\mathrm{T}} \mathbf{K}_\ell \mathbf{J}_{\ell c} \right)^{-1}, \quad \mathbf{C}_c \in \mathrm{I\!R}^{n \times n} \tag{6.23}$$

Note that (6.23) does not require that the task-space Jacobian and its derivatives be computed. It also does not require knowledge of the external wrench. It does in general require knowledge of the forces being applied to the actuation tendons. These forces can come either directly from force sensors placed in series with the actuation tendons [50, 25] or from estimates obtained via the commanded motor current. In special cases where the configuration space Jacobian is constant, e.g. torsionally stiff segments with constant pitch tendon routing (see Chapter 4), this term becomes zero and the configuration space compliance becomes constant.

6.4 Discussion: Obtaining Required Model Parameters

Both the configuration-space and task-space compliance matrices require the modal coefficients $\mathbf{c}$ to define the segment's configuration. There are two ways to obtain $\mathbf{c}$. The first is using the shape sensing approach presented in Chapter 4. This approach requires that shape-sensing strings be integrated into the segment, but the benefit is that it is purely kinematics-based, so it is more computationally efficient to compute, especially for segments with high torsional stiffness like the one presented in Chapter 3, where the modal coefficients are linear with respect to the string encoder measurements, as shown in Chapter 4.

For torsionally-stiff segments with straight string routing, the configuration space Jacobian is also constant, and therefore its derivative is zero. The term $\mathbf{C}_\tau$ therefore drops from the compliance equations. This means that the configuration-space compliance becomes constant, and that for robots with high torsional stiffness and straight string/tendon routing, the joint forces do not need to be measured to compute the configuration-space and task-space compliance.

The second way to obtain $\mathbf{c}$ is to use a mechanics model like the one presented in Chapter 6. After solving the mechanics model, $\mathbf{c}$ can be obtained by converting the collocation values into modal coefficients via (5.9). This second way to obtain $\mathbf{c}$ does not require shape-sensing strings, but does require an estimate of the external wrench applied to the segment, which may be known *a priori* in some cases or can be measured using a load cell attached to the end effector.

We also note that our formulation easily permits a high-order model to be used when computing the kinematics/statics but a lower-order model to be used when computing the compliance. This would be useful in a scenario where the computation cost of computing the derivatives of the task-space Jacobian in $\mathbf{C}_x$ was too high for the compute power available. Either a kinematics model with shape sensing or a mechanics model would be used to determine the modal coefficients. The higher-order terms of the modal basis would then be dropped and $\mathbf{C}_x$ determined using this reduced-order modal basis, allowing for a quick estimate of the compliance matrix to be computed for online control.

6.5 Conclusions

In this chapter, we presented a method for computing the compliance matrix of a tendon-actuated continuum robot utilizing the modal shape basis Lie group formulation from Chapters 4 and 5. This leads to analytic expressions for both the configuration-space and task-space compliance matrices for robots subject to variable curvature deflections. We believe these results lay the groundwork for future studies of stiffness modulation (as done in Chapter 2) and compliant motion control with a variety of continuum robot systems by providing formulations for the compliance matrix that are applicable to robots with different types of sensing modalities and under different choices of modeling assumptions.

CHAPTER 7

CONCLUSIONS

Collaborative manipulators need to be able to operate in dynamically changing, uncertain environments and must be able to sense and regulate contact forces with the environment. This is especially true for collaborative manufacturing in confined spaces, where contact forces can occur anywhere along the body of the robot. Although many works in the literature have pushed forward the state-of-the-art for collaborative manipulators, there are still research gaps and challenges that this dissertation seeks to address. This dissertation provided control, modeling, and sensing formulations to help address several challenges for compliant parallel and continuum robots in the context of physical human-robot interaction. We presented contributions in the areas of stiffness modulation of parallel robots, a new continuum robot design, shape sensing of variable curvature continuum robots subject to external loads using general string encoder routing, and new formulations for computing the statics and compliance matrices of continuum robots.

7.1 Summary of Findings

Robots with the ability to modulate their stiffness have been studied in prior work, but the combination of kinematic redundancy and variable stiffness actuation has not been previously studied for stiffness modulation of parallel robots. Chapter 2 presented a redundancy resolution method using gradient projection to modulate the passive directional stiffness of a kinematic redundant parallel robot equipped with variable stiffness actuators. We showed in a simulation study that the method successfully modulates the stiffness matrix to the desired values using our proposed redundancy resolution approach.

We then presented the design of a new modular continuum segment for collaborative manufacturing in confined spaces in Chapter 3, expanding the design space of continuum robots. The design integrates the actuation components into the base of the segment for improved modularity, provides a unique 2:1 tendon reduction to reduce the actuation component sizing, has high torsional stiffness to improve the load-carrying capacity and simplify the kinematic expressions (as shown in Chapter 4), and includes sensing modalities for shape, proximity, and contact sensing. In addition to presenting the mechanical design, we also discussed the electronics integration and motor control software implementation.

In Chapter 4, we proposed a modeling formulation for shape sensing of continuum segments with general string encoder routing. The formulation uses modal shape functions to describe the curvature of the central backbone. The Lie group expressions are used to derive the configuration-space and task-space Jacobians and the resulting formulation is used to solve for the modal coefficients and provide the shape of the segment. The approach was validated experimentally on the torsionally-stiff segment design from Chapter 3 with straight string routing where the high torsional stiffness results in a linear set of equations for the modal coefficients that can be solved rapidly. We also conducted a simulation study for a torsionally-compliant segment with helical routing, showing that, although more computationally expensive in this case, the approach is applicable to segments subject to general deflections. Both the experiments and the simulations demonstrated that our approach can accurately predict poses along the central backbone of each segment.

Building upon this shape sensing formulation, we used this Lie group kinematic formulation to solve Kirchhoff rod mechanics models. The approach combined orthogonal collocation (using the modal shape function concept presented in Chapter 4) with Lie group integration based on the Magnus expansion. This formulation resulted in an analytic expression for the residual Jacobian, allowing us to efficiently solve the Cosserat rod boundary value problem. We provided simulation results for the case of a Kirchhoff rod subject to external loads, showing that our approach can potentially be competitive with existing approach while providing analytic expressions for the kinematic Jacobians.

Finally, we showed how the modal shape basis Lie group formulation used to solve the shape sensing and mechanics problems results in analytic expressions for the compliance of a Kirchhoff rod as well as the configuration space and task space compliance matrices of tendon-actuated continuum robots. We discussed the model parameters that must be measured or estimated to compute each of these compliance matrices and the affect on computation burden associated with each of the compliance matrices and choice of model. We believe the task-space compliance matrix could allow the redundancy resolution method presented in Chapter 2 to be applied to a kinematically redundant continuum robot like the one presented in Chapter 3. The configuration-space compliance is also an extension of the compliance defined in [50], laying the groundwork for future studies of compliant motion control with variable curvature continuum robots subject to large external loads.

7.2 Future Research Directions

The work presented in this thesis opens the door to several areas of future research. One area is to investigate further improvements to the continuum module design presented in Chapter 3. First, we believe it is possible to integrate motor drive electronics into the base of the module to provide further modularity and reduce the number of wires being passed to the base of the robot. Second, we believe the 4.8 kg mass could be reduced through tighter integration of the mechanical components in the actuation unit, investigating designs that avoid the use of ball splines, and by optimizing the geometry of the end disk and intermediate disks to reduce mass.

Reducing the mass would increase safety by reducing the robot's inertia.

The shape sensing approach we presented in Chapter 4 used only kinematics expressions by design to avoid the computation cost of a mechanics model. However, in scenarios where sufficient compute power is available, the shape information provided by the string encoders could be used to improve the accuracy of a mechanics model similar to the one given in Chapter 5. We believe this could be achieved by adding the string encoder measurements as constraints on the mechanics boundary value problem and using the additional information provided by the string encoders to estimate other parameters of the mechanics model, e.g. using the string encoders to estimate the magnitude of an external force applied to the segment's end disk.

Another area to investigate is extending our approaches to cases with non-negligible extension and shear strain. In this dissertation, we have neglected extension and shear strains because, for most continuum robots, these deflections are reasonable to neglect. For some types of soft robots, however, these assumptions will likely no longer hold. Fortunately, the kinematic equations we present here can be extended to the case of extension and shear strains to allow the methods in Chapter 4 to be used, and the Kirchhoff rod models we used for the results in Chapter 5 can be extended to include these deflections. Future work to extend our methods to include extension and shear would expand the robot designs that our methods can be applied to.

Future work also includes further exploring the system design implications of implementing compliant motion control and compliance modulation using the compliance matrix formulations we presented in Chapter 6. We discussed in Chapter 6 the sensing needed to compute the configuration and task-space compliance depending on whether a kinematic model or a mechanics model is used, but it would be beneficial to explore these questions further with a physical system under real-world constraints. The analytic nature of our compliance matrix formulation would also allow for a study of the sensitivity of the model parameters on the compliance matrix accuracy so that the effect of sensing and calibration errors on the compliance model could be better understood.

There are also improvements that could be made in the Lie group integration formulation that would reduce computation cost. The fourth and sixth order Magnus expansions we used to carry out the Lie group integration and compute the task space Jacobians require many Lie bracket operations. Methods to reduce the number of Lie brackets are discussed in [155] using the theory of free Lie algebras. Nonlinear Lie group integration methods have also been studied [186] and could potentially be applied to the mechanics problems considered herein. We believe there are still substantial computation gains to be had by bringing more of these known methods from the geometric integration literature to bear on continuum robot modeling.

Appendix A

MINIMUM DISTANCE BETWEEN TWO LEGS

Modeling two of the prismatic legs as line segments, a minimum distance d_{min} must be maintained between these line segments. Considering two of the prismatic joint legs of the Stewart-Gough platform, the points on each leg defining the minimum distance vector are defined as:

$$\mathbf{p}_i = \mathbf{b}_i + t_i \hat{\mathbf{s}}_i, \quad t_i \in [0, l_i], \quad i = 1, 2, \tag{A.1}$$

The vector connecting these two points is:

$$\mathbf{m}_{12} = \mathbf{p}_2 - \mathbf{p}_1 = \mathbf{b}_2 - \mathbf{b}_1 + t_2 \hat{\mathbf{s}}_2 - t_1 \hat{\mathbf{s}}_1 \tag{A.2}$$

The minimum distance between the two lines is then found from the values of t_1 and t_2 that minimize the norm of $\mathbf{m}_{12}$:

$$\begin{aligned} \underset{t_1, t_2}{\text{minimize}} \quad & J = \frac{1}{2} \|\mathbf{m}_{12}\|^2 \\ \text{subject to} \quad & 0 < t_i < l_i, \quad i = 1, 2 \end{aligned} \tag{A.3}$$

Solving for the unconstrained minimum by equating partial derivative of J with respect to t_1 and t_2 to zero and subsequently solving for the values for t_1 and t_2 that define the common normal results in:

$$t_i^* = \frac{(\hat{\mathbf{s}}_1^{\mathrm{T}} - \hat{\mathbf{s}}_1^{\mathrm{T}} \hat{\mathbf{s}}_2 \hat{\mathbf{s}}_2^{\mathrm{T}})(\mathbf{b}_2 - \mathbf{b}_1)}{1 - (\hat{\mathbf{s}}_1^{\mathrm{T}} \hat{\mathbf{s}}_2)^2} \tag{A.4}$$

$$t_2^* = \frac{(\hat{\mathbf{s}}_2^{\mathrm{T}} \hat{\mathbf{s}}_1 \hat{\mathbf{s}}_1^{\mathrm{T}} - \hat{\mathbf{s}}_2^{\mathrm{T}})(\mathbf{b}_2 - \mathbf{b}_1)}{1 - (\hat{\mathbf{s}}_1^{\mathrm{T}} \hat{\mathbf{s}}_2)^2} \tag{A.5}$$

These expressions are undefined when the lines are parallel, i.e. $|\hat{\mathbf{s}}_1^{\mathrm{T}}\hat{\mathbf{s}}_2| = 1$. For parallel legs, t_2^* can be arbitrarily set to $\frac{1}{2}l_2$ and then t_1^* is found to be:

$$t_1^* = \hat{\mathbf{s}}_1^{\mathrm{T}}(\mathbf{b}_2 - \mathbf{b}_1) + \frac{1}{2}l_2\hat{\mathbf{s}}_1^{\mathrm{T}}\hat{\mathbf{s}}_2 \tag{A.6}$$

The values of t_1^* and t_2^* above may or may not be physically located on the legs of the robot. In the case that they are not both located on the legs of the robot, the other possible locations of the minimum distance are:

1. $t_1 = 0$ and solve for t_2 from $\partial J/\partial t_2 = 0$
2. $t_1 = l_1$ and solve for t_2 from $\partial J/\partial t_2 = 0$
3. $t_2 = 0$ and solve for t_1 from $\partial J/\partial t_1 = 0$
4. $t_2 = l_2$ and solve for t_1 from $\partial J/\partial t_1 = 0$
5. Any combination of the line segment end points

After finding the t_1 and t_2 that correspond to the physically feasible minimum distance, it is given by:

$$d = \|\mathbf{m}_{12}\| = \|\mathbf{b}_2 - \mathbf{b}_1 + t_2\hat{\mathbf{s}}_2 - t_1\hat{\mathbf{s}}_1\| \tag{A.7}$$

Defining the vector $\mathbf{u}$ for convenience, the partial derivative of d with respect to the i^{th} joint is then:

$$\mathbf{u} = \mathbf{b}_2 - \mathbf{b}_1 + t_2\hat{\mathbf{s}}_2 - t_1\hat{\mathbf{s}}_1 \tag{A.8}$$

$$\frac{\partial d}{\partial q_i} = \frac{\partial[\mathbf{u}^{\mathrm{T}}\mathbf{u}]^{\frac{1}{2}}}{\partial q_i} = \frac{\frac{1}{2}(\frac{\partial \mathbf{u}^{\mathrm{T}}}{\partial q_i}\mathbf{u} + \mathbf{u}^{\mathrm{T}}\frac{\partial \mathbf{u}}{\partial q_i})}{[\mathbf{u}^{\mathrm{T}}\mathbf{u}]^{\frac{1}{2}}} = \frac{\mathbf{u}^{\mathrm{T}}\frac{\partial \mathbf{u}}{\partial q_i}}{d} \tag{A.9}$$

The gradient of $\mathbf{u}$ is simple to find in closed form for each of the cases above. When either t_1 or t_2 is at an end point, the gradient is taken to be zero.

BIBLIOGRAPHY

[1] M. W. Hannan and I. D. Walker, "Kinematics and the implementation of an elephant's trunk manipulator and other continuum style robots," *Journal of Robotic Systems*, vol. 20, no. 2, pp. 45–63, 2003.

[2] V. C. Anderson and R. C. Horn, "Tensor arm manipulator," Feb. 24 1970. US Patent 3,497,083.

[3] S. Hirose, T. Kado, and Y. Umetani, "Tensor actuated elastic manipulator," *Proc. 6th IFToMM World Cong.*, vol. 2, pp. 978–981, 1981.

[4] W. Felt, M. J. Telleria, T. F. Allen, G. Hein, J. B. Pompa, K. Albert, and C. D. Remy, "An inductance-based sensing system for bellows-driven continuum joints in soft robots," *Autonomous Robots*, vol. 43, pp. 435–448, Feb. 2019.

[5] W. McMahan, V. Chitrakaran, M. Csencsits, D. Dawson, I. D. Walker, B. A. Jones, M. Pritts, D. Dienno, M. Grissom, and C. D. Rahn, "Field trials and testing of the OctArm continuum manipulator," in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pp. 2336–2341, May 2006.

[6] B. Vanderborght, A. Albu-Schaeffer, A. Bicchi, E. Burdet, D. G. Caldwell, R. Carloni, M. Catalano, O. Eiberger, W. Friedl, G. Ganesh, M. Garabini, M. Grebenstein, G. Grioli, S. Haddadin, H. Hoppner, A. Jafari, M. Laffranchi, D. Lefeber, F. Petit, S. Stramigioli, N. Tsagarakis, M. Van Damme, R. Van Ham, L. C. Visser, and S. Wolf, "Variable impedance actuators: A review," *Robotics and Autonomous Systems*, vol. 61, pp. 1601–1614, Dec. 2013.

[7] J. Burgner-Kahrs, D. C. Rucker, and H. Choset, "Continuum robots for

medical applications: A survey," *IEEE Transactions on Robotics*, vol. 31, no. 6, pp. 1261–1280, 2015.

[8] D. Rus and M. T. Tolley, "Design, fabrication and control of soft robots," *Nature*, vol. 521, no. 7553, pp. 467–475, 2015.

[9] B.-J. Yi and R. A. Freeman, "Geometric analysis of antagonistic stiffness in redundantly actuated parallel mechanisms," *Journal of Robotic Systems*, vol. 10, pp. 581–603, July 1993.

[10] S.-F. Chen and I. Kao, "Conservative congruence transformation for joint and cartesian stiffness matrices of robotic hands and fingers," *The International Journal of Robotics Research*, vol. 19, no. 9, pp. 835–847, 2000.

[11] N. Simaan and M. Shoham, "Geometric interpretation of the derivatives of parallel robots' jacobian matrix with application to stiffness control," *Journal of Mechanical Design*, vol. 125, no. 1, p. 33, 2003.

[12] N. Simaan and M. Shoham, "Stiffness synthesis of a variable geometry six-degrees-of-freedom double planar parallel robot," *The International Journal of Robotics Research*, vol. 22, pp. 757–775, Sept. 2003.

[13] S. Abdolshah, D. Zanotto, G. Rosati, and S. K. Agrawal, "Optimizing stiffness and dexterity of planar adaptive cable-driven parallel robots," *Journal of Mechanisms and Robotics*, vol. 9, no. 3, p. 031004, 2017.

[14] M. Anson, A. Alamdari, and V. Krovi, "Orientation workspace and stiffness optimization of cable-driven parallel manipulators with base mobility," *Journal of Mechanisms and Robotics*, vol. 9, p. 031011, Mar. 2017.

[15] W.-K. Kim, J.-Y. Lee, and B.-J. Yi, "Analysis for a planar 3 degree-of-freedom

parallel mechanism with actively adjustable stiffness characteristics," *KSME International Journal*, vol. 11, p. 408, July 1997.

[16] X. Zhou, S. Jun, and V. Krovi, "Planar cable robot with variable stiffness," in *Experimental Robotics*, pp. 391–403, Springer, 2016.

[17] G. L. H. Johnston, A. L. Orekhov, and N. Simaan, "Kinematic modeling and compliance modulation of redundant manipulators under bracing constraints," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4709–4716, IEEE, 2020.

[18] Y. L. Park, S. Elayaperumal, B. Daniel, S. C. Ryu, M. Shin, J. Savall, R. J. Black, B. Moslehi, and M. R. Cutkosky, "Real-time estimation of 3-D needle shape and deflection for MRI-guided interventions," *IEEE/ASME Transactions on Mechatronics*, vol. 15, pp. 906–915, Dec. 2010.

[19] R. Xu, A. Yurkewich, and R. V. Patel, "Curvature, torsion, and force sensing in continuum robots using helically wrapped FBG sensors," *IEEE Robotics and Automation Letters*, vol. 1, pp. 1052–1059, July 2016.

[20] A. Bajo, R. E. Goldman, and N. Simaan, "Configuration and joint feedback for enhanced performance of multi-segment continuum robots," in *2011 IEEE International Conference on Robotics and Automation*, pp. 2905–2912, May 2011.

[21] A. W. Mahoney, T. L. Bruns, P. J. Swaney, and R. J. Webster, "On the inseparable nature of sensor selection, sensor placement, and state estimation for continuum robots or "where to put your sensors and how to use them"," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4472–4478, 2016.

[22] S. Song, Z. Li, H. Yu, and H. Ren, “Shape reconstruction for wire-driven flexible robots based on Bézier curve and electromagnetic positioning,” *Mechatronics*, vol. 29, pp. 28–35, Aug. 2015.

[23] D. B. Camarillo, K. E. Loewke, C. R. Carlson, and J. K. Salisbury, “Vision based 3-D shape sensing of flexible manipulators,” in *2008 IEEE International Conference on Robotics and Automation*, pp. 2940–2947, May 2008.

[24] K. Xu and N. Simaan, “Intrinsic wrench estimation and its performance index for multisegment continuum robots,” *IEEE Transactions on Robotics*, vol. 26, pp. 555–561, June 2010.

[25] C. B. Black, J. Till, and D. C. Rucker, “Parallel continuum robots: Modeling, analysis, and actuation-based force sensing,” *IEEE Transactions on Robotics*, vol. 34, no. 1, pp. 29–47, 2017.

[26] C. G. Frazelle, A. D. Kapadia, and I. D. Walker, “A haptic continuum interface for the teleoperation of extensible continuum manipulators,” *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1875–1882, 2020.

[27] W. S. Rone and P. Ben-Tzvi, “Multi-segment continuum robot shape estimation using passive cable displacement,” in *2013 IEEE International Symposium on Robotic and Sensors Environments (ROSE)*, pp. 37–42, IEEE, 2013.

[28] K. Xu, Y. Chen, Z. Zhang, S. Zhang, N. Xing, and X. Zhu, “An insertable low-cost continuum tool for shape sensing,” in *2017 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 2044–2049, IEEE, Dec. 2017.

[29] T. Morales Bieze, F. Largilliere, A. Kruszewski, Z. Zhang, R. Merzouki, and C. Duriez, “FEM-based kinematics and closed-loop control of soft, continuum manipulators,” *Soft Robotics*, 2018.

[30] R. J. Webster and B. A. Jones, “Design and kinematic modeling of constant curvature continuum robots: A review,” *The International Journal of Robotics Research*, vol. 29, pp. 1661–1683, Nov. 2010.

[31] S. H. Sadati, S. E. Naghibi, A. Shiva, B. Michael, L. Renson, M. Howard, C. D. Rucker, K. Althoefer, T. Nanayakkara, S. Zschaler, C. Bergeles, H. Hauser, and I. D. Walker, “TMTDyn: A Matlab package for modeling and control of hybrid rigid–continuum robots based on discretized lumped systems and reduced-order models,” *The International Journal of Robotics Research*, Jan. 2020.

[32] J. Till, V. Aloi, and C. Rucker, “Real-time dynamics of soft and continuum robots based on Cosserat rod models,” *The International Journal of Robotics Research*, vol. 38, no. 6, pp. 723–746, 2019.

[33] S. M. H. Sadati, S. E. Naghibi, I. D. Walker, K. Althoefer, and T. Nanayakkara, “Control space reduction and real-time accurate modeling of continuum manipulators using Ritz and Ritz-Galerkin methods,” *IEEE Robotics and Automation Letters*, vol. 3, pp. 328–335, Jan. 2018.

[34] O. Weeger, S.-K. Yeung, and M. L. Dunn, “Isogeometric collocation methods for Cosserat rods and rod structures,” *Computer Methods in Applied Mechanics and Engineering*, vol. 316, pp. 100–122, 2017.

[35] Y. Liu, M. Wang, and L. Meng, “Spectral collocation method in the large deformation analysis of flexible beam,” *IAENG International Journal of Applied Mathematics*, vol. 48, no. 4, pp. 1–8, 2018.

[36] P. K. Masjedi and H. Ovesy, “Chebyshev collocation method for static intrinsic equations of geometrically exact beams,” *International Journal of Solids and Structures*, vol. 54, pp. 183–191, Feb. 2015.

[37] D. Zupan and M. Saje, “Finite-element formulation of geometrically exact three-dimensional beam theories based on interpolation of strain measures,” *Computer Methods in Applied Mechanics and Engineering*, vol. 192, pp. 5209–5248, Dec. 2003.

[38] P. Češarek, M. Saje, and D. Zupan, “Dynamics of flexible beams: Finite-element formulation based on interpolation of strain measures,” *Finite Elements in Analysis and Design*, vol. 72, pp. 47–63, 2013.

[39] F. Renda, F. Boyer, J. Dias, and L. Seneviratne, “Discrete Cosserat approach for multisection soft manipulator dynamics,” *IEEE Transactions on Robotics*, pp. 1–16, 2018.

[40] S. Grazioso, G. Di Gironimo, and B. Siciliano, “A geometrically exact model for soft continuum robots: The finite element deformation space formulation,” *Soft Robotics*, Nov. 2018.

[41] F. Renda, C. Armanini, V. Lebastard, F. Candelier, and F. Boyer, “A geometric variable-strain approach for static modeling of soft manipulators with tendon and fluidic actuation,” *IEEE Robotics and Automation Letters*, vol. 5, pp. 4006–4013, July 2020.

[42] F. Boyer, V. Lebastard, F. Candelier, and F. Renda, “Dynamics of continuum and soft robots: A strain parameterization based approach,” *IEEE Transactions on Robotics*, vol. 37, no. 3, pp. 847–863, 2020.

[43] P. E. Dupont, J. Lock, B. Itkowitz, and E. Butler, “Design and control of concentric-tube robots,” *IEEE Transactions on Robotics*, vol. 26, no. 2, pp. 209–225, 2009.

[44] D. C. Rucker, B. A. Jones, and R. J. W. III, “A geometrically exact model

for externally loaded concentric-tube continuum robots," *IEEE Transactions on Robotics*, vol. 26, pp. 769–780, Oct. 2010.

[45] D. C. Rucker and R. J. Webster, "Statics and dynamics of continuum robots with general tendon routing and external loading," *IEEE Transactions on Robotics*, vol. 27, pp. 1033–1044, Dec. 2011.

[46] J. Till, C. E. Bryson, S. Chung, A. Orekhov, and D. C. 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)*, pp. 5067–5074, May 2015.

[47] M. T. Chikhaoui, S. Lilge, S. Kleinschmidt, and J. Burgner-Kahrs, "Comparison of modeling approaches for a tendon actuated continuum robot with three extensible segments," *IEEE Robotics and Automation Letters*, vol. 4, pp. 989–996, Apr. 2019.

[48] A. L. Orekhov, V. A. Aloi, and D. C. Rucker, "Modeling parallel continuum robots with general intermediate constraints," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6142–6149, May 2017.

[49] J. Wang, J. Ha, and P. E. Dupont, "Steering a multi-armed robotic sheath using eccentric precurved tubes," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 9834–9840, IEEE, 2019.

[50] R. E. Goldman, A. Bajo, and N. Simaan, "Compliant motion control for multisegment continuum robots with actuation force sensing," *IEEE Transactions on Robotics*, vol. 30, no. 4, pp. 890–902, 2014.

[51] D. C. Rucker and R. J. Webster, "Computing Jacobians and compliance matrices for externally loaded continuum robots," in *2011 IEEE International Conference on Robotics and Automation*, pp. 945–950, IEEE, 2011.

[52] L. Wang and N. Simaan, “Geometric calibration of continuum robots: Joint space and equilibrium shape deviations,” *IEEE Transactions on Robotics*, vol. 35, no. 2, pp. 387–402, 2019.

[53] A. L. Orekhov and N. Simaan, “Directional stiffness modulation of parallel robots with kinematic redundancy and variable stiffness joints,” *Journal of Mechanisms and Robotics*, vol. 11, July 2019.

[54] M. Luces, J. K. Mills, and B. Benhabib, “A review of redundant parallel kinematic mechanisms,” *Journal of Intelligent & Robotic Systems*, vol. 86, pp. 175–198, May 2017.

[55] C. Gosselin and L.-T. Schreiber, “Redundancy in parallel mechanisms: A review,” *Applied Mechanics Reviews*, vol. 70, no. 1, p. 010802, 2018.

[56] S. Kim, “Operational quality analysis of parallel manipulators with actuation redundancy,” in *Robotics and Automation, 1997. Proceedings., 1997 IEEE International Conference on*, vol. 3, pp. 2651–2656, IEEE, 1997.

[57] J. Radojicic, D. Surdilovic, and G. Schreck, “Modular hybrid robots for safe human-robot interaction,” *World Academy of Science, Engineering and Technology (WCSAT 2009)*, 2009.

[58] J. K. Salisbury, “Active stiffness control of a manipulator in cartesian coordinates,” in *1980 19th IEEE Conference on Decision and Control including the Symposium on Adaptive Processes*, pp. 95–100, Dec. 1980.

[59] N. Hogan, “Impedance control: An approach to manipulation, parts I, II, III,” *ASME Journal of Dynamic Systems, Measurement, and Control*, vol. 107, no. 1, 1985.

[60] J. Loncaric, “Normal forms of stiffness and compliance matrices,” *IEEE Journal on Robotics and Automation*, vol. 3, no. 6, pp. 567–572, 1987.

[61] S. Huang and J. M. Schimmels, “The bounds and realization of spatial stiffnesses achieved with simple springs connected in parallel,” *IEEE Transactions on Robotics and Automation*, vol. 14, pp. 466–475, Jun 1998.

[62] H. Jamshidifar, A. Khajepour, B. Fidan, and M. Rushton, “Kinematically-constrained redundant cable-driven parallel robots: Modeling, redundancy analysis, and stiffness optimization,” *IEEE/ASME Transactions on Mechatronics*, vol. 22, pp. 921–930, Apr. 2017.

[63] S. Kock and W. Schumacher, “A parallel x-y manipulator with actuation redundancy for high-speed and active-stiffness applications,” in *Proceedings. 1998 IEEE International Conference on Robotics and Automation*, vol. 3, pp. 2295–2300 vol.3, May 1998.

[64] D. Chakarov, “Study of the antagonistic stiffness of parallel manipulators with actuation redundancy,” *Mechanism and Machine Theory*, vol. 39, pp. 583–601, June 2004.

[65] A. Müller, “Stiffness control of redundantly actuated parallel manipulators,” in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, pp. 1153–1158, IEEE, 2006.

[66] K. Yu, L. F. Lee, C. P. Tang, and V. N. Krovi, “Enhanced trajectory tracking control with active lower bounded stiffness control for cable robot,” in *2010 IEEE International Conference on Robotics and Automation*, pp. 669–674, May 2010.

[67] E. B. Pitt, N. Simaan, and E. J. Barth, “An investigation of stiffness modulation limits in a pneumatically actuated parallel robot with actuation

redundancy," in *ASME/BATH 2015 Symposium on Fluid Power and Motion Control*, p. V001T01A063, American Society of Mechanical Engineers, 2015.

[68] B.-J. Yi and R. A. Freeman, "Synthesis of actively adjustable springs by antagonistic redundant actuation," *Journal of dynamic systems, measurement, and control*, vol. 114, no. 3, pp. 454–461, 1992.

[69] A. Müller, "Consequences of geometric imperfections for the control of redundantly actuated parallel manipulators," *IEEE Transactions on Robotics*, vol. 26, pp. 21–31, Feb 2010.

[70] R. Van Ham, B. Vanderborght, M. Van Damme, B. Verrelst, and D. Lefeber, "MACCEPA, the mechanically adjustable compliance and controllable equilibrium position actuator: Design and implementation in a biped robot," *Robotics and Autonomous Systems*, vol. 55, no. 10, pp. 761–768, 2007.

[71] A. Jafari, N. G. Tsagarakis, and D. G. Caldwell, "AwAS-II: A new actuator with adjustable stiffness based on the novel principle of adaptable pivot point and variable lever ratio," in *2011 IEEE International Conference on Robotics and Automation*, pp. 4638–4643, IEEE, 2011.

[72] K. W. Hollander, T. G. Sugar, and D. E. Herring, "Adjustable robotic tendon using a 'Jack Spring'," in *9th International Conference on Rehabilitation Robotics, 2005.*, pp. 113–118, IEEE, 2005.

[73] J. Lee, A. Ajoudani, E. M. Hoffman, A. Rocchi, A. Settimi, M. Ferrati, A. Bicchi, N. G. Tsagarakis, and D. G. Caldwell, "Upper-body impedance control with variable stiffness for a door opening task," in *2014 IEEE-RAS International Conference on Humanoid Robots*, pp. 713–719, Nov. 2014.

[74] W. B. Lim, S. H. Yeo, G. Yang, and I. M. Chen, “Design and analysis of a cable-driven manipulator with variable stiffness,” in *2013 IEEE International Conference on Robotics and Automation*, pp. 4519–4524, May 2013.

[75] X. Zhou, S.-k. Jun, and V. Krovi, “A cable based active variable stiffness module with decoupled tension,” *Journal of Mechanisms and Robotics*, vol. 7, p. 011005, Feb. 2015.

[76] J. J. Rice and J. M. Schimmels, “Passive compliance control of redundant serial manipulators,” *Journal of Mechanisms and Robotics*, vol. 10, no. 4, p. 041009, 2018.

[77] A. Alamdari, R. Haghighi, and V. Krovi, “Stiffness modulation in an elastic articulated-cable leg-orthosis emulator: Theory and experiment,” *IEEE Transactions on Robotics*, vol. 34, no. 5, pp. 1266–1279, 2018.

[78] R. I. Alizade, N. R. Tagiyev, and J. Duffy, “A forward and reverse displacement analysis of a 6-DOF in-parallel manipulator,” *Mechanism and Machine Theory*, vol. 29, pp. 115–124, Jan. 1994.

[79] L.-W. Tsai and F. Tahmasebi, “Synthesis and analysis of a new class of six-degree-of-freedom parallel minimanipulators,” *Journal of Robotic Systems*, vol. 10, pp. 561–580, July 1993.

[80] F. Behi, “Kinematic analysis for a six-degree-of-freedom 3-PRPS parallel mechanism,” *IEEE Journal on Robotics and Automation*, vol. 4, pp. 561–565, Oct. 1988.

[81] R. Ben-Horin, M. Shoham, and S. Djerassi, “Kinematics, dynamics and construction of a planarly actuated parallel robot,” *Robotics and Computer-Integrated Manufacturing*, vol. 14, pp. 163–172, Apr. 1998.

[82] I. A. Bonev and C. m. M. Gosselin, “Geometric algorithms for the computation of the constant-orientation workspace and singularity surfaces of a special 6-RUS parallel manipulator,” in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. 36533, pp. 505–514, 2002.

[83] G. Wu, “Multiobjective optimum design of a 3-rrr spherical parallel manipulator with kinematic and dynamic dexterities,” *Modeling, Identification and Control: A Norwegian Research Bulletin*, vol. 33, no. 3, pp. 111–121, 2012.

[84] D. R. Neill, R. Sneed, J. Dawson, J. Sebag, and W. Gressler, “Baseline design of the LSST hexapods and rotator,” in *Advances in Optical and Mechanical Technologies for Telescopes and Instrumentation*, vol. 9151, p. 91512B, International Society for Optics and Photonics, 2014.

[85] B. A. Sawyer, “Magnetic positioning device,” Apr. 2 1968. US Patent 3,376,578.

[86] M. Mohamed and J. Duffy, “A direct determination of the instantaneous kinematics of fully parallel robot manipulators,” *Journal of Mechanisms, Transmissions, and Automation in Design*, vol. 107, no. 2, pp. 226–229, 1985.

[87] N. Sima’an, D. Glozman, and M. Shoham, “Design considerations of new six degrees-of-freedom parallel robots,” in *Proceedings. 1998 IEEE International Conference on Robotics and Automation*, vol. 2, pp. 1327–1333, May 1998.

[88] L.-W. Tsai, *Robot Analysis: The Mechanics of Serial and Parallel Manipulators.* John Wiley & Sons, Feb. 1999.

[89] B. J. Yi, R. A. Freeman, and D. Tesar, “Open-loop stiffness control of overconstrained mechanisms/robotic linkage systems,” in *1989 International Conference on Robotics and Automation Proceedings*, pp. 1340–1345 vol.3, May 1989.

[90] Y. Li and I. Kao, "Stiffness control on redundant manipulators: a unique and kinematically consistent solution," in *Robotics and Automation, 2004. Proceedings. ICRA'04. 2004 IEEE International Conference on*, vol. 4, pp. 3956–3961, IEEE, 2004.

[91] S. Huang and J. M. Schimmels, "The bounds and realization of spatial stiffnesses achieved with simple springs connected in parallel," *IEEE Transactions on Robotics and Automation*, vol. 14, no. 3, pp. 466–475, 1998.

[92] A. Liegeois, "Automatic supervisory control of the configuration and behavior of multibody mechanisms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 7, no. 12, pp. 868–871, 1977.

[93] I. D. Walker and S. I. Marcus, "Subtask performance by redundancy resolution for redundant robot manipulators," *IEEE Journal on Robotics and Automation*, vol. 4, pp. 350–354, June 1988.

[94] L. Li, W. A. Gruver, Q. Zhang, and Z. Yang, "Kinematic control of redundant robots and the motion optimizability measure," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 31, pp. 155–160, Feb. 2001.

[95] Y. Liu, J. Zhao, and B. Xie, "Obstacle avoidance for redundant manipulators based on a novel gradient projection method with a functional scalar," in *2010 IEEE International Conference on Robotics and Biomimetics*, pp. 1704–1709, Dec. 2010.

[96] J. A. Euler, R. V. Dubey, and S. M. Babcock, "Self motion determination based on actuator velocity bounds for redundant manipulators," *Journal of Robotic Systems*, vol. 6, pp. 417–425, Aug. 1989.

[97] W. A. Khan and J. Angeles, "The kinetostatic optimization of robotic

manipulators: the inverse and the direct problems," *Journal of Mechanical Design*, vol. 128, no. 1, pp. 168–178, 2006.

[98] H. Zghal, R. V. Dubey, and J. A. Euler, "Efficient gradient projection optimization for manipulators with multiple degrees of redundancy," in *IEEE International Conference on Robotics and Automation Proceedings*, pp. 1006–1011 vol.2, May 1990.

[99] S. Agarwal, R. A. Srivatsan, and S. Bandyopadhyay, "Analytical determination of the proximity of two right-circular cylinders in space," *Journal of Mechanisms and Robotics*, vol. 8, no. 4, p. 041010, 2016.

[100] J. S. Ketchel and P. M. Larochelle, "Self-collision detection in spatial closed chains," *Journal of Mechanical Design*, vol. 130, p. 092305, Aug. 2008.

[101] J. P. Merlet and D. Daney, "Legs interference checking of parallel robots over a given workspace or trajectory," in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pp. 757–762, May 2006.

[102] S. Perreault, P. Cardou, C. M. Gosselin, and M. J.-D. Otis, "Geometric determination of the interference-free constant-orientation workspace of parallel cable-driven mechanisms," *Journal of Mechanisms and Robotics*, vol. 2, p. 031016, July 2010.

[103] G. Grioli, S. Wolf, M. Garabini, M. Catalano, E. Burdet, D. Caldwell, R. Carloni, W. Friedl, M. Grebenstein, M. Laffranchi, *et al.*, "Variable stiffness actuators: The user's point of view," *The International Journal of Robotics Research*, vol. 34, no. 6, pp. 727–743, 2015.

[104] J. W. Hurst, J. E. Chestnutt, and A. A. Rizzi, "The actuator with mechanically

adjustable series compliance," *IEEE Transactions on Robotics*, vol. 26, no. 4, pp. 597–606, 2010.

[105] S. Robla-Gómez, V. M. Becerra, J. R. Llata, E. Gonzalez-Sarabia, C. Torre-Ferrero, and J. Perez-Oria, "Working together: A review on safe human-robot collaboration in industrial environments," *IEEE Access*, vol. 5, pp. 26754–26773, 2017.

[106] International Federation of Robotics, "World Robotics 2020 Industrial Robots," Sept. 2020.

[107] U.S. Bureau of Labor Statistics, "Survey of occuptational injuries and illnesses data," May 2020.

[108] S. Haddadin, A. De Luca, and A. Albu-Schäffer, "Robot collisions: A survey on detection, isolation, and identification," *IEEE Transactions on Robotics*, vol. 33, no. 6, pp. 1292–1312, 2017.

[109] C. Fang, N. Kashiri, G. F. Rigano, A. Ajoudani, and N. G. Tsagarakis, "Exploitation of environment support contacts for manipulation effort reduction of a robot arm," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 9502–9508, IEEE, 2019.

[110] N. Delson and H. West, "Bracing to increase the natural frequency of a manipulator: Analysis and design," *The International journal of robotics research*, vol. 12, no. 6, pp. 560–571, 1993.

[111] J. D. Greer, L. H. Blumenschein, R. Alterovitz, E. W. Hawkes, and A. M. Okamura, "Robust navigation of a soft growing robot by exploiting contact with the environment," *The International Journal of Robotics Research*, p. 0278364920903774, 2020.

[112] M. Selvaggio, L. Ramirez, N. Naclerio, B. Siciliano, and E. Hawkes, "An obstacle-interaction planning method for navigation of actuated vine robots," in *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3227–3233, IEEE, 2020.

[113] S. Hirose, "Biologically inspired robots," *Snake-Like Locomotors and Manipulators*, 1993.

[114] J. Santoso and C. D. Onal, "An origami continuum robot capable of precise motion through torsionally stiff body and smooth inverse kinematics," *Soft Robotics*, pp. 371–386, July 2020.

[115] X. Chen, Y. Guo, D. Duanmu, J. Zhou, W. Zhang, and Z. Wang, "Design and modeling of an extensible soft robotic arm," *IEEE Robotics and Automation Letters*, vol. 4, pp. 4208–4215, Oct. 2019.

[116] A. Raisch, A. Mayer, D. Müller, A. Hildebrandt, and O. Sawodny, "A model-based cascaded control concept for the bionic motion robot," in *2020 American Control Conference (ACC)*, pp. 2049–2054, July 2020.

[117] T. Mahl, A. Hildebrandt, and O. Sawodny, "A variable curvature continuum kinematics for kinematic control of the bionic handling assistant," *IEEE Transactions on Robotics*, vol. 30, pp. 935–949, Aug. 2014.

[118] M. B. Pritts and C. D. Rahn, "Design of an artificial muscle continuum robot," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, vol. 5, pp. 4742–4746, IEEE, 2004.

[119] G. Immega and K. Antonelli, "The KSI tentacle manipulator," in *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, vol. 3, (Nagoya, Japan), pp. 3149–3154, IEEE, 1995.

[120] S. Sanan, M. H. Ornstein, and C. G. Atkeson, “Physical human interaction for an inflatable manipulator,” in *2011 Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, pp. 7401–7404, Aug. 2011.

[121] M. T. Gillespie, C. M. Best, and M. D. Killpack, “Simultaneous position and stiffness control for an inflatable soft robot,” in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1095–1101, May 2016.

[122] T. Aoki, H. Ohno, and S. Hirose, “Design of Slim Slime Robot II (SSR-II) with bridle bellows,” in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 1, pp. 835–840, Sept. 2002.

[123] J. A. Childs and C. Rucker, “Leveraging geometry to enable high-strength continuum robots,” *Frontiers in Robotics and AI*, vol. 8, 2021.

[124] H. Wei, Y. Chen, J. Tan, and T. Wang, “Sambot: A self-assembly modular robot system,” *IEEE/ASME Transactions on Mechatronics*, vol. 16, no. 4, pp. 745–757, 2010.

[125] D. Rollinson, Y. Bilgen, B. Brown, F. Enner, S. Ford, C. Layton, J. Rembisz, M. Schwerin, A. Willig, P. Velagapudi, *et al.*, “Design and architecture of a series elastic snake robot,” in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4630–4636, IEEE, 2014.

[126] T. Tosun, J. Davey, C. Liu, and M. Yim, “Design and characterization of the EP-face connector,” in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 45–51, IEEE, 2016.

[127] J. A. Childs and C. Rucker, “Concentric precurved bellows: new bending actuators for soft robots,” *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 1215–1222, 2020.

[128] C. Abah, A. L. Orekhov, G. L. H. Johnston, P. Yin, H. Choset, and N. Simaan, "A multi-modal sensor array for safe human-robot interaction and mapping," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 3768–3774, IEEE, May 2019.

[129] C. Abah, A. L. Orekhov, G. L. H. Johnston, and N. Simaan, "A multi-modal sensor array for human-robot interaction and confined spaces exploration using continuum robots," *IEEE Sensors Journal*, 2022 (in press).

[130] R. J. Roesthuis, M. Kemp, J. J. v. d. Dobbelsteen, and S. Misra, "Three-dimensional needle shape reconstruction using an array of fiber bragg grating sensors," *IEEE/ASME Transactions on Mechatronics*, vol. 19, pp. 1115–1126, Aug. 2014.

[131] J. H. B. Koh, T. Jeong, S. Han, W. Li, K. Rhode, and Y. Noh, "Optoelectronic sensor-based shape sensing approach for flexible manipulators," in *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 3199–3203, July 2019.

[132] D. Trivedi and C. D. Rahn, "Model-based shape estimation for soft robotic manipulators: The planar case," *Journal of Mechanisms and Robotics*, vol. 6, p. 021005, May 2014.

[133] Y. Shapiro, G. Kosa, and A. Wolf, "Shape tracking of planar hyper-flexible beams via embedded PVDF deflection sensors," *IEEE/ASME Transactions on Mechatronics*, vol. 19, pp. 1260–1267, Aug. 2014.

[134] R. Reilink, S. Stramigioli, and S. Misra, "3D position estimation of flexible instruments: marker-less and marker-based methods," *International journal of computer assisted radiology and surgery*, vol. 8, no. 3, pp. 407–417, 2013.

[135] C. Shi, X. Luo, P. Qi, T. Li, S. Song, Z. Najdovski, T. Fukuda, and H. Ren, "Shape sensing techniques for continuum robots in minimally invasive surgery: A survey," *IEEE Transactions on Biomedical Engineering*, vol. 64, pp. 1665–1678, Aug. 2017.

[136] P. H. Nguyen, S. Sridar, W. Zhang, and P. Polygerinos, "Design and control of a 3-chambered fiber reinforced soft actuator with off-the-shelf stretch sensors," *International Journal of Intelligent Robotics and Applications*, vol. 1, pp. 342–351, Sept. 2017.

[137] H. A. Wurdemann, S. Sareh, A. Shafti, Y. Noh, A. Faragasso, D. S. Chathuranga, H. Liu, S. Hirai, and K. Althoefer, "Embedded electro-conductive yarn for shape sensing of soft robotic manipulators," in *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 8026–8029, Aug. 2015.

[138] S. Cheng, Y. S. Narang, C. Yang, Z. Suo, and R. D. Howe, "Stick-on large-strain sensors for soft robots," *Advanced Materials Interfaces*, vol. 6, no. 20, p. 1900985, 2019.

[139] C. G. Frazelle, A. Kapadia, and I. Walker, "Developing a kinematically similar master device for extensible continuum robot manipulators," *Journal of Mechanisms and Robotics*, vol. 10, p. 025005, Feb. 2018.

[140] C. Escande, T. Chettibi, R. Merzouki, V. Coelen, and P. M. Pathak, "Kinematic calibration of a multisection bionic manipulator," *IEEE/ASME Transactions on Mechatronics*, vol. 20, pp. 663–674, Apr. 2015.

[141] J. Bosman, T. M. Bieze, O. Lakhal, M. Sanz, R. Merzouki, and C. Duriez, "Domain decomposition approach for FEM quasistatic modeling and control

of Continuum Robots with rigid vertebras," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 4373–4378, May 2015.

[142] O. Lakhal, A. Melingui, T. M. Bieze, C. Escande, B. Conrard, and R. Merzouki, "On the kinematic modeling of a class of continuum manipulators," in *2014 IEEE International Conference on Robotics and Biomimetics (ROBIO 2014)*, pp. 368–373, IEEE, 2014.

[143] C. Wang, C. G. Frazelle, J. R. Wagner, and I. Walker, "Dynamic control of multi-section three-dimensional continuum manipulators based on virtual discrete-jointed robot models," *IEEE/ASME Transactions on Mechatronics*, pp. 777–788, 2020.

[144] A. Melingui, O. Lakhal, B. Daachi, J. B. Mbede, and R. Merzouki, "Adaptive neural network control of a compact bionic handling arm," *IEEE/ASME Transactions on Mechatronics*, vol. 20, pp. 2862–2875, Dec. 2015.

[145] O. Lakhal, A. Melingui, and R. Merzouki, "Hybrid approach for modeling and solving of kinematics of a compact bionic handling assistant manipulator," *IEEE/ASME Transactions on Mechatronics*, vol. 21, pp. 1326–1335, June 2016.

[146] K. Oliver-Butler, J. Till, and C. Rucker, "Continuum robot stiffness under external loads and prescribed tendon displacements," *IEEE Transactions on Robotics*, vol. 35, no. 2, pp. 403–419, 2019.

[147] C. D. Santina and D. Rus, "Control oriented modeling of soft robots: The polynomial curvature case," *IEEE Robotics and Automation Letters*, vol. 5, pp. 290–298, Apr. 2020.

[148] G. S. Chirikjian and J. W. Burdick, "A modal approach to hyper-redundant manipulator kinematics," *IEEE Transactions on Robotics and Automation*, vol. 10, no. 3, pp. 343–354, 1994.

[149] K. E. Zanganeh and J. Angeles, “The inverse kinematics of hyper-redundant manipulators using splines,” in *Proceedings of 1995 IEEE International Conference on Robotics and Automation*, vol. 3, pp. 2797–2802, IEEE, 1995.

[150] J. Zhang, J. Roland, J. Thomas, S. Manolidis, and N. Simaan, “Optimal Path Planning for Robotic Insertion of Steerable Electrode Arrays in Cochlear Implant Surgery,” *Journal of Medical Devices*, vol. 3, Dec. 2008.

[151] I. S. Godage, G. A. Medrano-Cerda, D. T. Branson, E. Guglielmino, and D. G. Caldwell, “Modal kinematics for multisection continuum arms,” *Bioinspiration & Biomimetics*, vol. 10, no. 3, p. 035002, 2015.

[152] A. W. Mahoney, T. L. Bruns, R. Alterovitz, and R. J. Webster III, “Design, sensing, and planning: Fundamentally coupled problems for continuum robots,” in *Robotics Research*, pp. 267–282, Springer, 2018.

[153] A. Nahvi and J. M. Hollerbach, “The noise amplification index for optimal pose selection in robot calibration,” in *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 1, pp. 647–654, 1996.

[154] K. M. Lynch and F. C. Park, *Modern Robotics*. Cambridge University Press, 2017.

[155] A. Iserles, H. Z. Munthe-Kaas, S. P. Nørsett, and A. Zanna, “Lie-group methods,” *Acta numerica*, vol. 9, pp. 215–365, 2000.

[156] A. L. Orekhov and N. Simaan, “Solving Cosserat rod models via collocation and the Magnus expansion,” in *IEEE/RSJ International Conference on Robots and Systems (IROS)*, Oct. 2020.

[157] P. S. Gonthina, A. D. Kapadia, I. S. Godage, and I. D. Walker, “Modeling variable curvature parallel continuum robots using Euler curves,” in *2019*

International Conference on Robotics and Automation (ICRA), pp. 1679–1685, IEEE, 2019.

[158] P. Rao, Q. Peyron, and J. Burgner-Kahrs, "Using Euler curves to model continuum robots," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1402–1408, 2021.

[159] J. C. Mason and D. C. Handscomb, *Chebyshev polynomials.* Chapman and Hall/CRC, 2002.

[160] W. Rossmann, *Lie groups: An introduction through linear groups*, vol. 5. Oxford University Press on Demand, 2006.

[161] J. M. Selig, *Geometric fundamentals of robotics.* Springer Science & Business Media, 2004.

[162] 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*, vol. 30, no. 5, pp. 1109–1122, 2014.

[163] N. Simaan, R. Taylor, and P. Flint, "A dexterous system for laryngeal surgery," in *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, vol. 1, pp. 351–357, 2004.

[164] X. Dong, M. Raffles, S. Cobos-Guzman, D. Axinte, and J. Kell, "A novel continuum robot using twin-pivot compliant joints: design, modeling, and validation," *Journal of Mechanisms and Robotics*, vol. 8, no. 2, p. 021010, 2016.

[165] J. P. Merlet, "Jacobian, manipulability, condition number, and accuracy of parallel robots," *Journal of Mechanical Design*, vol. 128, pp. 199–206, June 2005.

[166] F. Renda, M. Cianchetti, H. Abidi, J. Dias, and L. Seneviratne, "Screw-based modeling of soft manipulators with tendon and fluidic actuation," *Journal of Mechanisms and Robotics*, Apr. 2017.

[167] N. Simaan, "Snake-like units using flexible backbones and actuation redundancy for enhanced miniaturization," in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pp. 3012–3017, IEEE, 2005.

[168] L. C. Young, "Orthogonal collocation revisited," *Computer Methods in Applied Mechanics and Engineering*, vol. 345, pp. 1033–1076, 2019.

[169] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang, *Spectral methods: fundamentals in single domains.* Springer Science & Business Media, 2007.

[170] J. Villadsen and M. L. Michelsen, *Solution of differential equation models by polynomial approximation*, vol. 7. Prentice-Hall Englewood Cliffs, NJ, 1978.

[171] C. Rucker, "Integrating rotations using nonunit quaternions," *IEEE Robotics and Automation Letters*, vol. 3, no. 4, pp. 2979–2986, 2018.

[172] J. Park and W.-K. Chung, "Geometric integration on euclidean group with application to articulated multibody systems," *IEEE Transactions on Robotics*, vol. 21, no. 5, pp. 850–863, 2005.

[173] A. Müller, "Approximation of finite rigid body motions from velocity fields," *ZAMM - Journal of Applied Mathematics and Mechanics*, vol. 90, pp. 514–521, June 2010.

[174] W. Magnus, "On the exponential solution of differential equations for a linear operator," *Communications on pure and applied mathematics*, vol. 7, no. 4, pp. 649–673, 1954.

[175] H. Munthe-Kaas and B. Owren, “Computations in a free lie algebra,” *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, vol. 357, no. 1754, pp. 957–981, 1999.

[176] P. C. Moan and J. Niesen, “Convergence of the Magnus series,” *Foundations of Computational Mathematics*, vol. 8, no. 3, pp. 291–301, 2008.

[177] M. G. de Azevedo Bahia, R. F. Dias, and V. T. L. Buono, “The influence of high amplitude cyclic straining on the behaviour of superelastic NiTi,” *International journal of fatigue*, vol. 28, no. 9, pp. 1087–1091, 2006.

[178] L. L. Howell and A. Midha, “Parametric deflection approximations for end-loaded, large-deflection beams in compliant mechanisms,” *Journal of Mechanical Design*, vol. 117, pp. 156–165, March 1995.

[179] K. Xu and N. Simaan, “Analytic formulation for kinematics, statics, and shape restoration of multibackbone continuum robots via elliptic integrals,” *Journal of Mechanisms and Robotics*, vol. 2, no. 1, 2010.

[180] W. Wei and N. Simaan, “Modeling, force sensing, and control of flexible cannulas for microstent delivery,” *Journal of Dynamic Systems, Measurement, and Control*, vol. 134, p. 041004, Apr. 2012.

[181] M. Mahvash and P. E. Dupont, “Stiffness control of surgical continuum manipulators,” *IEEE Transactions on Robotics*, vol. 27, no. 2, pp. 334–345, 2011.

[182] G. Jelenić and M. Saje, “A kinematically exact space finite strain beam model—finite element formulation by generalized virtual work principle,” *Computer Methods in Applied Mechanics and Engineering*, vol. 120, no. 1-2, pp. 131–161, 1995.

[183] J. Simo, “The (symmetric) hessian for geometrically nonlinear models in solid mechanics: intrinsic definition and geometric interpretation,” *Computer Methods in Applied Mechanics and Engineering*, vol. 96, no. 2, pp. 189–200, 1992.

[184] I. Gravagne and I. Walker, “Manipulability, force, and compliance analysis for planar continuum manipulators,” *IEEE Transactions on Robotics and Automation*, vol. 18, no. 3, pp. 263–273, 2002.

[185] G. Alici and B. Shirinzadeh, “Enhanced stiffness modeling, identification and characterization for robot manipulators,” *IEEE Transactions on Robotics*, vol. 21, no. 4, pp. 554–564, 2005.

[186] A. Zanna, “Collocation and relaxed collocation for the Fer and the Magnus expansions,” *SIAM journal on numerical analysis*, vol. 36, no. 4, pp. 1145–1182, 1999.