5-2019

On the Statics, Dynamics, and Stability of Continuum Robots: Model Formulations and Efficient Computational Schemes

John Daniel Till
University of Tennessee, jtill@vols.utk.edu

On the Statics, Dynamics, and Stability of Continuum Robots: Model Formulations and Efficient Computational Schemes

A Dissertation Presented for the

Doctor of Philosophy

Degree

The University of Tennessee, Knoxville

John Daniel Till

May 2019

Acknowledgments

I have benefited enormously from my apprenticeship to Caleb Rucker. Shortly after being shown to my desk on my first day of work, he pointed me to (2.15) in his dissertation [93] and said: "It's a system of nonlinear ODEs. You'll want to numerically approximate it with the fourth-order Runge-Kutta method." At the time that sounded like a lot of big words, but he was supportive, and his passion for teaching and mathematics have allowed me to gain a grasp of differential equations well beyond what I could have achieved on my own.

I am thankful to my committee- Reza Abedi, Michael Berry, and Bill Hamel, for their input to determine the scope of my dissertation and ensure its technical merit. The excellent teaching of the MABE faculty at the University of Tennessee, and more broadly my classroom experiences on the hill have been invaluable. I am also indebted to my previous instructors at Tennessee Technological University and Chattanooga State Community College. Particularly I would like to thank Rita Barnes, Mark Boshart, Steve Canfield, Marsha Schoonover, Jennifer Smith, Theresa Underwood-Lemons, and Chris Wilson. Finally, my mother, Lynn Till, left a successful career at Oak Ridge National Laboratory to teach and care for her four children, and my father, David Till, frequently turned the dinner table discussion to power systems engineering (particularly distribution), and his diligence kept the lights on and freed me to focus on my studies.

I would like to thank my coworkers- Vince Aloi, Caroline Black, Brett Brownlee, Kaitlin Oliver-Butler, Jonathan Carlton, Jake Childs, Scotty Chung, Adam Daniel, James Ferguson, Zachariah Nelson, Ryan Ponten, and Andrew Orekhov. In addition to the obvious fact that we supported each other through various co-authorships, their involvement made the office enjoyable, and there was no shortage of exciting projects to see when I needed a break from my own work.

The co-authorship of Katy Riojas and Bob Webster at Vanderbilt University was vital to perform the concentric tube modeling work in Chapter 3.5, and it was a pleasure working as part of that team. I was constantly impressed by Katy's work ethic, and fortunately if my co-authors were impressed by my work ethic, they were too gracious to say anything. The high-speed camera used for the experiment in Chapter 3.1.5 was generously provided and operated by Christopher Combs, Phillip Kreth, and John Schmisseur of the University of Tennessee Space Institute. Caroline Black, Andrew Orekhov, and Caleb Rucker completed most of the design and fabrication work to build the robots in Chapter 2.2, and Kaitlin Oliver-Butler lent her design expertise to the construction of the tendon-driven robot in Chapter 3.3.4.

This material is based upon work supported by the National Science Foundation under CMMI-1427122 as part of the NSF/NASA/NIH/USDA/DOD National Robotics Initiative and under NSF CAREER Award IIS-1652588. Any opinion, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation.

Abstract

This dissertation presents advances in continuum-robotic mathematical-modeling techniques. Specifically, problems of statics, dynamics, and stability are studied for robots with slender elastic links. The general procedure within each topic is to develop a continuous theory describing robot behavior, develop a discretization strategy to enable simulation and control, and to validate simulation predictions against experimental results.

Chapter 1 introduces the basic concept of continuum robotics and reviews progress in the field. It also introduces the mathematical modeling used to describe continuum robots and explains some notation used throughout the dissertation.

The derivation of Cosserat rod statics, the coupling of rods to form a parallel continuum robot (PCR), and solution of the kinematics problem are reviewed in Chapter 2. With this foundation, soft real-time teleoperation of a PCR is demonstrated and a miniature prototype robot with a grasper is controlled.

Chapter 3 reviews the derivation of Cosserat rod dynamics and presents a discretization strategy having several desirable features, such as generality, accuracy, and potential for good computational efficiency. The discretized rod model is validated experimentally using high speed camera footage of a cantilevered rod. The discretization strategy is then applied to simulate continuum robot dynamics for several classes of robot, including PCRs, tendon-driven robots, fluidic actuators, and concentric tube robots.

In Chapter 4, the stability of a PCR is analyzed using optimal control theory. Conditions of stability are gradually developed starting from a single planar rod and finally arriving at a stability test for parallel continuum robots. The approach is experimentally validated using a camera tracking system.

Chapter 5 provides closing discussion and proposes potential future work.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Continuum Robots

"Continuum" robotics is a subfield distinct from conventional rigid-link robotics because elastic links are used to achieve movement through controlled deformation. Figure 1.1 illustrates several representative designs of continuum robots. Various advantages to continuum robots have been highlighted, such as flexibility, light weight, inherent safety, scalability, and potential for low-cost parts. Relevant applications have been researched, such as material handling, exploration, and minimally invasive surgery [90, 121, 117, 17].

There is particular interest in continuum robots for minimally-invasive surgical applications owing to their flexibility. Compliant manipulators such as colonoscopes and catheters have become cornerstones of surgery, and researchers aspire to develop continuum devices with even greater reach and articulation. One of the most researched designs is the concentric tube robot, which is composed of concentrically aligned pre-curved tubes [118, 27, 32]. The tubes may be pre-curved by plastic deformation, by thermal "shape setting" [33], or even 3D printing [68]. When inserted concentrically, the collection of tubes will take some shape to reconcile opposing pre-curvatures. In this way translating or rotating the base of a tube will result in a change to the entire robot shape, and 6-DOF control of the tip is possible for a three-tube robot. The pre-curved tube shapes may be chosen to optimize the robot behavior for specific tasks [5, 43, 16, 38]. Another popular design is tendon-driven manipulators,

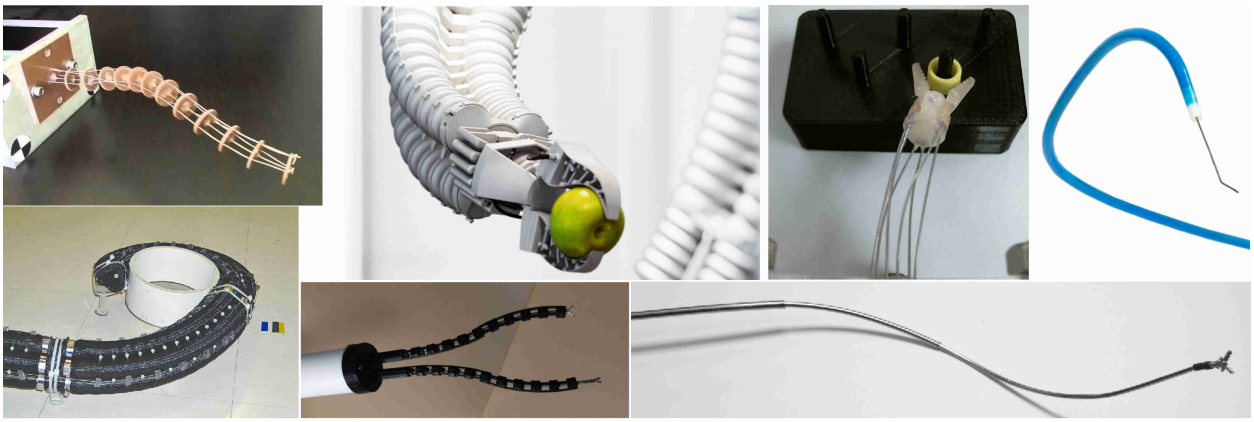


Figure 1.1: A sample of continuum robot designs is shown to illustrate the diverse design space. From left to right then top to bottom, these designs are: tendon-driven robots [97], the Festo bionic handling assistant [61], parallel continuum robots [77], Hansen Medical's Magellan robotic catheter, the OctArm [65], concentric notched-tube robots [72], and concentric tube robots [32].

which like colonoscopes and many catheters, rely on cables routed along the robot at an offset from the centerline to generate bending motion [18, 97]. Although tendons are often routed at a constant offset from the backbone, the path of the tendon routing channel may be chosen to alter the end-effector kinematic mapping [97] and stiffness to applied loads [73]. Continuum robots may also be used to supplement existing medical tools, such as increasing the articulation of tools deployed through a colonoscope using concentric tubes [81, 80] or concentric notched-tube arms [72].

Although many continuum robots have a serial design with a shape described by a central curve, often referred to as the "backbone", parallel designs have also been investigated. Parallel continuum robots use multiple elastic members connected in a parallel arrangement, which can provide increased precision and stiffness compared to slenderer, single-member continuum robots [13, 9, 8]. These designs not only have potential for large-scale compliant interaction with humans, but also small-scale surgical tasks because their scalability enables operation with many articulated DOF in short, confined spaces [78, 77]. Examples of parallel continuum robot architectures include the multi-backbone snake-like robots of Simaan et al. [25, 103], the Festo bionic tripod manipulator, soft parallel robots [88, 47], and continuum Stewart-Gough designs [13, 9, 76].

Outside of medical robotics, the compliant grasping and inherent safety of continuum robots has also resulted in applications of pneumatic robots for general manipulation tasks. The pneumatically actuated arm "OctArm" was an influential project in which robot motion was achieved by McKibben muscles, which shorten their length under internal pressure [36, 65]. Tubes were aligned in parallel against a flexible backbone to create a multi-section arm. This arm was attached to a mobile platform and the research team demonstrated the robot's ability to manipulate objects.

In addition to truly continuous robots, hyper-redundant robots having many discrete links are also commonly studied and regarded as continuum robots. Some would pinpoint the origin of continuum robotics to 1967 when Anderson and Horn published a journal article on the hyper-redundant discrete-link "Tensor Arm" [1], which was comprised of a series of stacked plates which could be rotated by pulling on tendons. Another notable hyper-redundant robot is the CardioARM [23], a highly redundant snake-like arm intended for cardiac ablation, where portions of the heart wall are cauterized to achieve insulation against irregular electrical rhythms.

There have been applications of continuum robots in industry. Hyper-redundant snake robots are marketed by OC Robotics [14] for industrial inspection and exploration. Medrobotics targets transoral surgery with the FLEX snake robot [89], which uses a backbone with adjustable stiffness so that the surgeon can make the main section rigid while manipulating instruments at the tip. Robotic catheter systems are sold by Hansen Medical and Stereotaxis. German industrial control and automation company Festo has created several novel continuum robots such as the bionic tripod and bionic handling assistant [61]. Olympus has researched and designed the "EndoSAMURAI" robotic colonoscope [99]. Samsung has developed a single-port access surgical robot [57] with a guide tube comprised of discrete disks with routing holes for tendons similar to the original tensor arm. NASA has researched tendon-driven continuum robots [66] such as the "tendril" robot, a tendon-actuated arm designed for "minimally invasive inspection" situations such as exploring crevices.

1.2 Elastic Rod Modeling

Generally continuum robots use slender elastic elements which may be idealized as one-dimensional objects. Slender elastic objects exhibiting large deflections are becoming increasingly prevalent not only in continuum robotics, but also for other applications such as soft robotics [98] and in interactions with objects such as ropes, sutures, needles, and catheters [104]. Flexible robot dynamics have been researched for decades with applications to spacecraft arms, energy-saving lightweight robots, and collaborative robots [3]. While tractable solutions can often be found with Euler-Bernoulli beam theory, this relies on the assumption of small deflections. In applications with large deflections, classical rod theories in nonlinear elasticity are needed.

Many models assume a continuum arm is comprised of sections having constant curvature[1] [121], which results in relatively simple mathematical models. This is particularly appealing since the resulting transformations can be represented by the Denavit-Hartenberg parameters which are used to describe rigid robots [121, 117, 41]. This idea facilitated the growth of continuum robotics, but the approach is fundamentally limited. Even the subset of robots that have a constant curvature shape in an unstressed resting configuration[2] will have variable curvature when the robot experiences loading or inertial dynamics. The ability to deform is, after all, a main feature of continuum robotics.

The approximate nature of constant curvature representations prompted the use of generalized elastic models such as planar Bernoulli-Euler elastica theory [35] and spatial Cosserat rod theory [115, 94, 97, 27, 85]. These theories describe the equations of motions of a rod, and are derived from first principles under mild assumptions such as idealizing the rod as a slender object as reviewed in Chapters 2.1.1 and 3.1.1. Although the Cosserat brothers derived their theory of elasticity at the turn of the 1900s [2], there has been recent interest in applying the equations to describe continuum robots, and in finding discretization schemes for simulation. The modeling efforts of this dissertation are mainly focused on these goals.

[1] "Constant curvature" is a phrase understood to mean a curvature which is constant with respect to a curve's arc length, although the curvature may be non-constant in time.

[2] e.g. the Bionic Handling Assistant and OctArm in Figure 1.1.

Many continuum robots may be modeled as one or more elastic rods subject to constraints arising from the robot design. The constraints may be discrete, as is the case with parallel continuum robots [13, 8, 9] and some rod manipulation tasks such as two grippers handling a rod [12]. Other scenarios impose continuous constraints on elastic rods, for example concentric tubes [94], tendon robots [97], soft robots [84, 83], and rod manipulation in a confined environment [108]. Imposing continuous constraints tends to give rise to differential algebraic equations (DAEs) describing the robot statics, and a main success of past works has been solving for an explicit form of the static ODEs. An overview of the Cosserat rod equations and their application to tendon and concentric-tube robots is provided in [93].

Generally there are not known analytical solutions to the equations of motion, or even the ODEs describing robot statics. Regarding numerical solutions, finite-element models (FEM) for large-deformation 3D nonlinear elasticity usually entail unnecessary computational expense when describing long, slender objects like rods and rod-based robots because general deformations of the rod cross sections are included. Classical Cosserat rod models assume no cross-section deformation and gain a great amount of efficiency due to this simplification (e.g. two orders of magnitude in [116] for an extended Cosserat model with cross-section inflation). Methods for simulating rod deformation range from non-physical techniques that achieve an aesthetic goal to accurate mechanics-based models [30]. Accurate models can be based on finite element methods [104, 26], finite differences [55, 86, 97], and differential algebraic equation solvers [40]. Many rod dynamics implementations first discretize the rod geometry in the spatial dimension and then derive equations of motion defining the accelerations of the generalized coordinates, which are typically integrated numerically with an explicit time-marching method. Obtaining the solution involves enforcing inextensibility and unshearability constraints (via minimal coordinates [7, 91], projection [6], or Lagrange multipliers with post-stabilization [106]), or explicitly modeling the stiff dynamics of shear and extension [97, 85] (equivalent to enforcement via a penalty method [107, 104]). Many explicit methods exhibit good computational efficiency and can run in real time. However, they require work-intensive derivations dependent on particular choices of model effects/assumptions and spatial discretization, which has a cost in terms of implementation effort and modularity. Furthermore, for all methods using explicit time

integration, the maximum time step is limited by stability conditions, which is especially limiting if the shear and extension are modeled or enforced via a penalty method.

Some desirable characteristics of a modeling and computational approach for the dynamics of slender elastic objects undergoing large deflections are as follows:

- Numerical consistency with the continuous theory
- Real-time computation for simulation and control
- Good scalability with respect to spatial resolution
- Stability under large time steps
- High order of accuracy in steady-state / static cases
- Low numerical damping

The method presented in Chapter 3 satisfies the above criteria. The approach directly solves the nonlinear, hyperbolic, partial-differential equations (PDE) for 3D, large-deflection elastic rods using implicit discretization of time derivatives and a shooting method in the spatial dimension. For the simpler problem of robot statics, the time semi-discretization is unnecessary and the spatial discretization is the same, so the description of static problems in Chapter 2 provides a good lead-in to dynamics. A shooting-method approach to solving the statics BVP or semi-discretized dynamics BVP is attractive because the size of the nonlinear system to be solved always remains the same, even as the spatial resolution increases. One argument against shooting methods is that the shooting Jacobian is prone to ill-conditioning. This is indeed a potential issue, but shooting methods can be quite reliable depending on the underlying problem being solved, particularly when the initial guess is close to the solution, which is often the case for successive iterations in a simulation.

The time semi-discretization in Chapter 3 is inspired by a relatively unexplored method suggested by [29] for modeling the planar motion of fly-fishing lines, and subsequently developed further by Lan and Lee in [53, 54] for planar compliant mechanisms. Starting with the continuous PDE, the time derivatives are discretized using a chosen implicit differentiation formula. This creates a continuous ordinary boundary value problem in the spatial dimension that can be solved to obtain the rod state at each time step. This implicit temporal-discretization approach provides consistency, stability under large time steps, and

the potential for high-order accuracy, scalability, and efficiency. Any shear and extension behavior (including inextensibility) is automatically satisfied by the spatial integration of the strains. Current efforts using this type of approach have been limited to planar dynamics, and computation of solutions at interactive rates has not been demonstrated. In this dissertation the approach is extended to the 3D spatial case, applied with higher-order numerical schemes in both the space and time dimensions, and demonstrated to achieve soft real-time performance.

A principle advantage of this method is that only simple changes are required to explore various model assumptions (neglecting shear, viscosity, etc.), external force terms such as tendon actuation [97], and discretization schemes in space or time, in contrast to methods which symbolically solve for acceleration terms. We believe that this modularity is worth the slight increase in run time, especially given the potential for greater accuracy by using high-order schemes.

Implementing a continuum robot simulation at interactive rates can be difficult due to the computational burden and inherent complexity of Cosserat rod models, but there is precedent. The concentric tube model introduced in [94] represents each robot section as a nonlinear system of ordinary differential equations (ODEs) which includes computationally expensive trigonometric functions arising from the relative rotation between tubes. Despite this complexity, it was shown in [15] that the model can be solved in software at rates enabling teleoperation (the robot Jacobian can be calculated at a rate of at least 200Hz on a modest 2.5GHz processor). One of the main factors in increasing the model's computational efficiency was exploiting mathematical relationships between the forward kinematics shooting problem and the robot's Jacobian [95]. Higher bandwidths may be achieved with model simplifications, e.g. linearization of the ODEs [123] or approximation based on precomputed solutions [27]. The efforts required in these works illustrate that solving systems of interacting Cosserat rods is a non-trivial task for current computing platforms to perform fast enough for control or rapid simulation. Results from C++ implementations of the models in this dissertation reflect favorably on the chosen discretization strategies.

1.3 Elastic Stability

As continuum robots store elastic energy, there is a potential for an elastic instability resulting in buckling or snapping behavior of the robot. Maintaining elastic stability has been recognized as a concern for many continuum robots, and prior research has investigated stability questions related to design and control. For cable-driven continuum robots, Li and Rahn [58] demonstrated buckling of the central backbone between two cable supports. Concentric tubes of sufficient curvature may exhibit a bifurcation due to relative rotation of the tubes. The elastic stability of concentric-tube robots been studied extensively with analyses based on energy [120, 119, 96], monotonicity and slope of an input-output "S-curve" mapping [27, 4, 5, 38], variational calculus [43, 31], and optimal control [39]. Aside from robotics applications, the stability of elastic rods is important in fields such as DNA modeling [114, 44] and computer graphics simulation [104]. Many rod stability problems have been studied in the continuum mechanics literature. Approaches include the use of variational calculus [60, 105], dynamic lumped parameter models [42], and finite element methods [56]. This field has also included studies of special cases such as branched rods [75] and rods with intrinsic curvature [63].

Although instability is generally regarded as unwanted behavior, Riojas et al have recently shown that concentric tube snapping may be useful to accomplish tasks such as propelling a suture through tissue [87], and Mochiyama et al considered the potential of snap-through plate buckling to generate a jumping motion [67]. However, there has been little research to model dynamic continuum robot behavior resulting from instability; previous work has mainly focused on detecting when an instability occurs. In Chapter 3 the dynamics of parallel continuum robot snapping [109] and concentric tube snapping [111] are simulated.

Chapter 4 presents a way to determine if a parallel continuum robot static solution (given by a numerical solution to our mechanics model) is elastically stable and thus physically realizable. This can be used to assess stability in a real-time simulation in order to avoid actuator commands that would cause unstable dynamic transitions during robot teleoperation. Stability assessment can also be used as a criteria in offline motion planning and simulation-based design optimization.

The recent concentric-tube work of Ha et al. [39] is similar to the efforts here in that it applies established results in optimal control to formulate a numerical test for the stability of a concentric-tube robot model solution. Bretl, et al. have also recently studied stability for robotic manipulation of a single elastic Kirchhoff rod (a special case of Cosserat rods with no transverse shear or axial strain) [64, 11, 12, 10]. Their approach rigorously uses geometric optimal control theory for problems defined on manifolds (since the state variable is a member of the group SE(3)) and considers the case of a fully constrained terminal state (pose) with no external loading. They have shown all static equilibrium configurations form a path-connected smooth manifold with a global chart.

The optimal control approach is elegant, rigorous, and is minimally affected by discretization issues, in contrast to the lumped-parameter and finite element approaches. Thus Chapter 4 constructs an approach based on optimal control which builds on the work above and provides some distinct contributions. First, in contrast to prior work, the general problem of one or more elastic rods under loading and subject to *arbitrary* terminal constraints is considered. Whereas [39] considers no terminal constraints (a free end), and [12] considers a fully constrained terminal pose (a fixed end), the approach here can be used to assess the stability of elastic rods with partial constraints (e.g. constraints made by pinned joints, sliding joints, etc.), or a geometric coupling to another elastic system (as in the case of parallel continuum robots). The stability of planar, tree-like rod structures studied in [75] is a related problem, but the connectivity graph of a parallel continuum robot can contain a closed cycle, which requires general terminal conditions not considered in [75]. Second, the approach here examines the full Cosserat rod model in addition to the more restricted Kirchhoff model (no shear or axial strains) studied in [39, 12]. While the differences are largely negligible for slender rods, removing the Kirchhoff restrictions expands the generality of the approach, making it suitable for soft parallel robots with non-negligible shear and extension strains, such as [88, 47], and in general for rods with a low slenderness ratio. Third, the approach here provides a unique way of dealing with the spatial case (where the rod state variable is an element of the Euclidean group SE(3)) by using Euler-Poincaré reduction following Holm's treatment in [45], resulting in a minimal set of Lagrange multipliers, which simplifies the conditions for equilibrium and stability. This

approach is perhaps more accessible than the geometric optimal control formalism in [12] while still obtaining a minimal model representation that takes advantage of group symmetry. Contrasted against achievements of prior literature, the main contribution of this work is the consideration of general boundary conditions rather than a fixed end or a free end with an applied force. Although used here to couple rods, such a derivation would be useful in various scenarios such as touching a surface that applies a normal reaction force.

1.4 Notation and Conventions

To make reading equations easier, any vector and matrix variables will be typed in bold and scalar quantities will not. When taking a partial derivative with respect to a vector, numerator layout notation is used, meaning a matrix of partial derivatives $\boldsymbol{J} = \partial \boldsymbol{y}/\partial \boldsymbol{x}$ has elements $J_{i,j} = \partial y_i/\partial x_j$. For example a 2x2 system has the form

$$\boldsymbol{J} = \begin{bmatrix} \dfrac{\partial y_1}{\partial x_1} & \dfrac{\partial y_1}{\partial x_2} \\ \dfrac{\partial y_2}{\partial x_1} & \dfrac{\partial y_2}{\partial x_2} \end{bmatrix}.$$

A shorthand for partial derivatives is used when convenient so that a subscripted variable denotes a partial derivative with respect to that variable, for example $\partial \boldsymbol{p}/\partial s$ is equivalent to $\boldsymbol{p}_s$.

For the purposes of model derivation, orientation is generally represented as a rotation matrix $\boldsymbol{R} \in \mathrm{SO}(3)$ (although model implementations may use a different representation, e.g. quaternions as in Chapter 2.1.3). Some variables are defined in the global frame, but may be expressed in the local frame as denoted by an upper-right subscript, for example global-frame internal force $\boldsymbol{n}$ and body-frame internal force $\boldsymbol{n}^b$ so that $\boldsymbol{n}^b := \boldsymbol{R}^\top \boldsymbol{n}$. As in

[69], the $\widehat{\ }$ symbol denotes a mapping from $\mathbb{R}^3$ to $\mathfrak{so}(3)$ as follows,

$$\widehat{\mathbf{a}} := \begin{bmatrix} 0 & -a_3 & a_2 \\ a_3 & 0 & -a_1 \\ -a_2 & a_1 & 0 \end{bmatrix}, \tag{1.1}$$

and $\vee$ denotes the inverse mapping of $\widehat{\ }$, i.e. $(\widehat{\boldsymbol{a}})^\vee = \boldsymbol{a}$. For a matrix $\boldsymbol{b} \in \mathfrak{so}(3)$, this has the effect

$$\boldsymbol{b}^\vee := \begin{bmatrix} b_{3,2} & b_{1,3} & b_{2,1} \end{bmatrix}^\top. \tag{1.2}$$

We will often need to find an error between two rotation matrices, which requires some function $\boldsymbol{E} : \mathrm{SO}(3) \times \mathrm{SO}(3) \to \mathbb{R}^3$. There are multiple functions which can fulfill this requirement, but the metric defined in [62] is especially straightforward. This is given by

$$\boldsymbol{E}(\boldsymbol{R}_1, \boldsymbol{R}_2) := \left(\boldsymbol{R}_1^\top \boldsymbol{R}_2 - \boldsymbol{R}_1 \boldsymbol{R}_2^\top\right)^\vee. \tag{1.3}$$

This metric has the limitation that it only has a local minimum when the two rotations are the same, as demonstrated by $\boldsymbol{E}(\boldsymbol{I}, \boldsymbol{R}_z(\pi)) = \mathbf{0}$, where $\boldsymbol{R}_z(x)$ is a rotation about the z-axis by an angle x. Thus for solvers using convex optimization routines it is important for the initial error to be in the correct basin of attraction. If this is of concern, then one may select an error metric as in [13] which uses the matrix logarithm:

$$\boldsymbol{E}(\boldsymbol{R}_1, \boldsymbol{R}_2) := \log(\boldsymbol{R}_1^\top \boldsymbol{R}_2)^\vee.$$

As described in [19], the matrix logarithm is given by

$$\log \boldsymbol{R} = \frac{\theta}{2\sin\theta}(\boldsymbol{R}^\top - \boldsymbol{R}), \text{ where } \theta = \arccos\left(\frac{\mathrm{trace}(\boldsymbol{R}) - 1}{2}\right).$$

This metric is undefined for some inputs but has a non-zero limit, e.g.

$$\lim_{x \to \pi} \boldsymbol{E}(\boldsymbol{I}, \boldsymbol{R}_z(x)) = \begin{bmatrix} 0 & 0 & \pi \end{bmatrix}^\top.$$

Chapter 2

Statics

Often the assumption of quasi-static motion is adequate to describe and control continuum robots. This chapter begins by reviewing the development of the static Cosserat rod equations, a set of nonlinear ordinary differential equations based on idealizing an elastic rod as a one-dimensional object, then considers the coupling of rods to form a parallel continuum robot and issues related to efficient model implementation for teleoperation.

2.1 Individual Rod

2.1.1 Derivation of Static Cosserat Rod Equations

The derivation of rod statics here follows that of Antman [2], although there are differences in notation. For the purposes of mechanics modeling, a slender rod can be reasonably approximated as a one-dimensional object. The single dimension is the arclength, which is denoted by s. Typically the arclength will vary from zero at the base of the rod to the rod length L, that is $s \in [0 \quad L] \subset \mathbb{R}$. The rod has the ability to elongate, so we make the distinction that s and L are both defined in a stress-free configuration. Position and orientation of a rod are shown in Figure 2.1. The rod's Cartesian centerline position is described by a function $\boldsymbol{p}(s) \in \mathbb{R}^3$. The centerline defines a tangent direction of the rod, but the rod is capable of building up torque about this centerline so that the twist angle is also

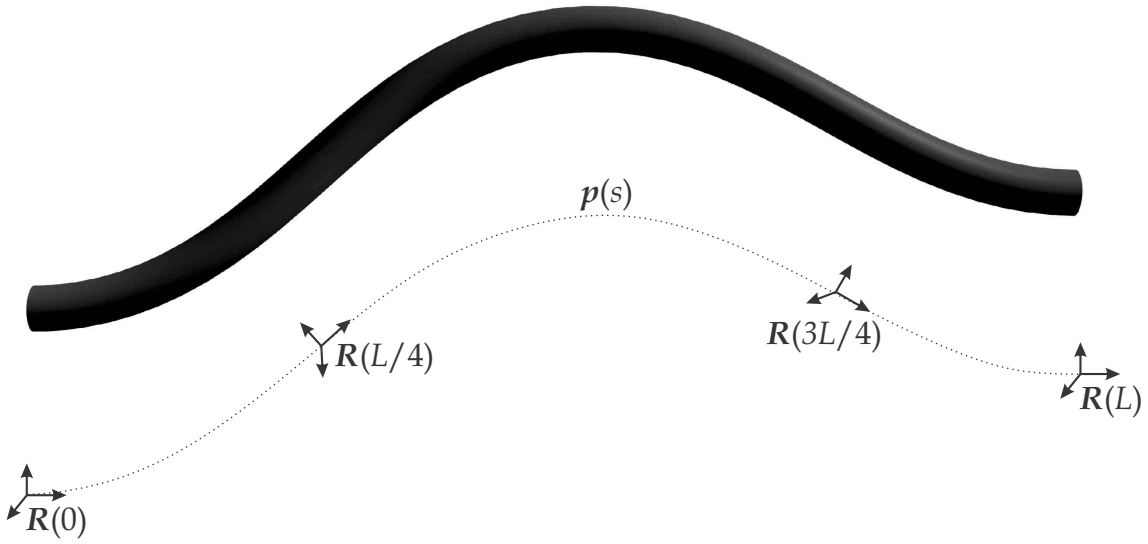


Figure 2.1: The rod is represented by a continuous centerline function $\boldsymbol{p}(s)$ and orientation so that each point along the rod has a six-DOF pose. The orientation will typically be implemented in rotation matrix form as $\boldsymbol{R}(s)$.

relevant. Thus the rod state includes 3 DOF orientation, which will typically be represented with a rotation matrix $\boldsymbol{R}(s) \in \mathrm{SO}(3)$.

A variable $\boldsymbol{v}$ is introduced as the first derivative of position in the local frame, that is $\boldsymbol{v} := \boldsymbol{R}^\top \boldsymbol{p}_s$. Another variable $\boldsymbol{u}$ is introduced as curvature in the local frame, that is $\boldsymbol{u} := \left(\boldsymbol{R}^\top \boldsymbol{R}_s\right)^\vee$. It may be helpful to realize this is analogous to body-frame angular velocity $\boldsymbol{\omega} := \left(\boldsymbol{R}^\top \boldsymbol{R}_t\right)^\vee$.

The internal force of the rod is described by $\boldsymbol{n}(s)$ and the internal moment by $\boldsymbol{m}(s)$. The sign convention is chosen so that the force $\boldsymbol{n}(s)$ is the force which the material at $s + \delta$ exerts on the material at $s - \delta$ for some infinitesimal δ, and likewise for the moment sign convention. Any distributed forces acting on the rod are integrated over the cross-section to describe the action on the rod centerline, which gives rise to a one-dimensional distributed force $\boldsymbol{f}(s)$ and distributed moment $\boldsymbol{l}(s)$ as shown in Figure 2.2.

A differential equation for $\boldsymbol{n}_s := \partial \boldsymbol{n}/\partial s$ can be derived by considering the force balance on some infinitesimal slice of the rod from s to $s + \delta$, as illustrated in Figure 2.3. Recalling

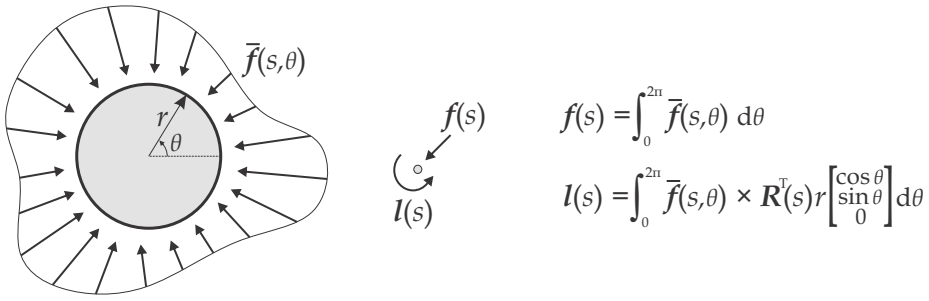


Figure 2.2: The rod has some force profile over its surface $\bar{\boldsymbol{f}}(s,\theta)$ which is idealized as one-dimensional distributed force $\boldsymbol{f}(s)$ and moment $\boldsymbol{l}(s)$ functions.

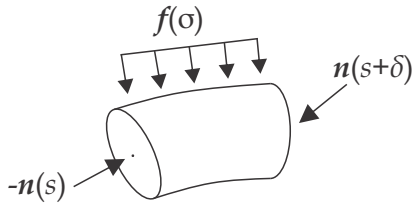


Figure 2.3: The force balance of an infinitesimal section of a rod is considered. The balance equation $\boldsymbol{n}(s+\delta) - \boldsymbol{n}(s) + \int_s^{s+\delta} \boldsymbol{f}(\sigma) d\sigma = \boldsymbol{0}$ is differentiated to obtain $\boldsymbol{n}_s = -\boldsymbol{f}$.

the sign convention, the static force balance equation is

$$\boldsymbol{n}(s+\delta) - \boldsymbol{n}(s) + \int_s^{s+\delta} \boldsymbol{f}(\sigma) d\sigma = \boldsymbol{0}.$$

The equation is differentiated to obtain

$$\boldsymbol{n}_s = -\boldsymbol{f}.$$

Finding a differential equation for $\boldsymbol{m}_s$ is similar, but requires a few more steps. Taking the moment balance of an infinitesimal section yields

$$\boldsymbol{m}(s+\delta) - \boldsymbol{m}(s) + \boldsymbol{p}(s+\delta) \times \boldsymbol{n}(s+\delta) - \boldsymbol{p}(s) \times \boldsymbol{n}(s) + \int_s^{s+\delta} \left[\boldsymbol{l}(\sigma) + \boldsymbol{p}(\sigma) \times \boldsymbol{f}(\sigma)\right] d\sigma = \boldsymbol{0}.$$

This is differentiated to obtain

$$\boldsymbol{m}_s = -\boldsymbol{l} - \boldsymbol{p} \times \boldsymbol{f} - \frac{\partial}{\partial s} \left(\boldsymbol{p} \times \boldsymbol{n}\right).$$

Differentiating the term on the right-hand side reveals a cancellation by

$$\begin{aligned}\boldsymbol{m}_s &= -\boldsymbol{l} - \boldsymbol{p} \times \boldsymbol{f} - \boldsymbol{p}_s \times \boldsymbol{n} - \boldsymbol{p} \times \boldsymbol{n}_s \\ &= -\boldsymbol{l} - \boldsymbol{p} \times \boldsymbol{f} - \boldsymbol{p}_s \times \boldsymbol{n} + \boldsymbol{p} \times \boldsymbol{f} \\ &= -\boldsymbol{l} - \boldsymbol{p}_s \times \boldsymbol{n}.\end{aligned}$$

Thus the set of differential equations describing the statics of an elastic rod is

$$\begin{aligned}\boldsymbol{p}_s &= \boldsymbol{R}\boldsymbol{v} \\ \boldsymbol{R}_s &= \boldsymbol{R}\widehat{\boldsymbol{u}} \\ \boldsymbol{n}_s &= -\boldsymbol{f} \\ \boldsymbol{m}_s &= -\boldsymbol{p}_s \times \boldsymbol{n} - \boldsymbol{l}.\end{aligned} \tag{2.1}$$

Some constitutive equation relating internal loading to strains must be specified to fully constrain the ODE system. Frequently a linear elastic relation is used:

$$\begin{aligned}\boldsymbol{n} &= \boldsymbol{R}\boldsymbol{K}_{se}(\boldsymbol{v} - \boldsymbol{v}^*) \\ \boldsymbol{m} &= \boldsymbol{R}\boldsymbol{K}_{bt}(\boldsymbol{u} - \boldsymbol{u}^*)\end{aligned} \tag{2.2}$$

The variables $\boldsymbol{v}^*$ and $\boldsymbol{u}^*$ indicate the shape of the rod in a stress-free situation. A typical rod which assumes a straight shape absent loading has $\boldsymbol{v}^* = \boldsymbol{e}_3$ and $\boldsymbol{u}^* = \boldsymbol{0}$, although in some cases it is useful for rods to have a curved stress-free shape. The "se" and "bt" subscripts of the stiffness matrices denote shear, extension, bending, and torsion. Typically the rod material is homogenous so that the stiffness matrices are

$$\boldsymbol{K}_{se} = \begin{bmatrix} GA & 0 & 0 \\ 0 & GA & 0 \\ 0 & 0 & EA \end{bmatrix}, \qquad \boldsymbol{K}_{bt} = \begin{bmatrix} EI_{xx} & 0 & 0 \\ 0 & EI_{yy} & 0 \\ 0 & 0 & GI_{zz} \end{bmatrix} \tag{2.3}$$

where $I_{xx} = I_{yy} = \pi r^4/4$ and $I_{zz} = I_{xx} + I_{yy}$ for a circular rod.

The system formed by (2.1) and (2.2) is a set of nonlinear ODEs in the arclength dimension.

$$\boldsymbol{v} = \boldsymbol{v}^* + \boldsymbol{K}_{se}^{-1}\boldsymbol{R}^\top \boldsymbol{n}$$
$$\boldsymbol{u} = \boldsymbol{u}^* + \boldsymbol{K}_{bt}^{-1}\boldsymbol{R}^\top \boldsymbol{m}$$

$$\begin{aligned} \boldsymbol{p}_s &= \boldsymbol{R}\boldsymbol{v} \\ \boldsymbol{R}_s &= \boldsymbol{R}\widehat{\boldsymbol{u}} \\ \boldsymbol{n}_s &= -\boldsymbol{f} \\ \boldsymbol{m}_s &= -\boldsymbol{p}_s \times \boldsymbol{n} - \boldsymbol{l}, \end{aligned} \tag{2.4}$$

2.1.2 Examples

With (2.4) the development has finally arrived at a complete system of ODEs which is fully constrained given some initial values. For example, consider a steel rod ($E = 207$GPa and $G = 80$GPa) with a radius of 1mm and a combination 6-DOF force sensor at the base to measure $\boldsymbol{n}(0)$ and $\boldsymbol{m}(0)$ arising from some unknown point wrench at the distal end. The coordinate frame is arbitrarily assigned so that $\boldsymbol{p}(0) = \boldsymbol{0}$ and $\boldsymbol{R}(0) = \boldsymbol{I}$. Distributed weight is considered so that $\boldsymbol{f} = \rho A \boldsymbol{g}$, where ρ is the material density (around 8000kg/m^3 for steel), A the cross-sectional area πr^2, and $\boldsymbol{g}$ the gravitational acceleration vector. The ODE system can be numerically solved to find the rod shape as a function of the measured 6-DOF force. The results of numerical solving the ODE with $\boldsymbol{n}(0) = [0\ 1\ 0]^\top$ and $\boldsymbol{m}(0) = \boldsymbol{0}$ are shown in Figure 2.4, and MATLAB code to numerical solve the IVP is in Appendix A.

In practice it is rare to have an initial value problem, and the situation usually includes unknown boundary conditions at both ends. For instance, the rod may be handled by robotic grippers at either end so that $\boldsymbol{p}(0)$, $\boldsymbol{R}(0)$, $\boldsymbol{p}(L)$, and $\boldsymbol{R}(L)$ are controlled. This problem can be solved with a shooting method where $\boldsymbol{n}(0)$ and $\boldsymbol{m}(0)$ are iteratively guessed and the rod is numerically integrated to evaluate the error between integrated and constrained values of $\boldsymbol{p}(L)$ and $\boldsymbol{R}(L)$. Figure 2.5 shows a solution when $\boldsymbol{p}(L) = \begin{bmatrix} 0 & -0.1L & 0.8L \end{bmatrix}^\top$ and $\boldsymbol{R}(L) = \boldsymbol{I}$.

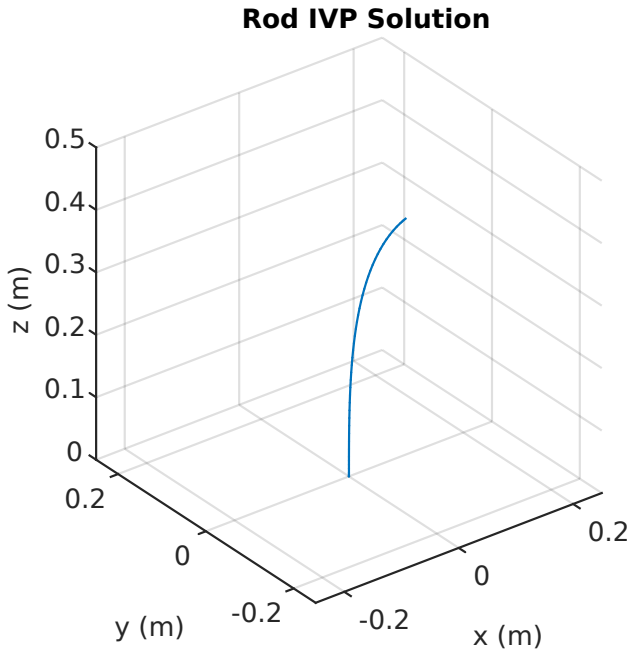


Figure 2.4: A rod IVP centerline solution is visualized for $\boldsymbol{n}(0) = [0\ 1\ 0]^\top$ and $\boldsymbol{m}(0) = \boldsymbol{0}$.

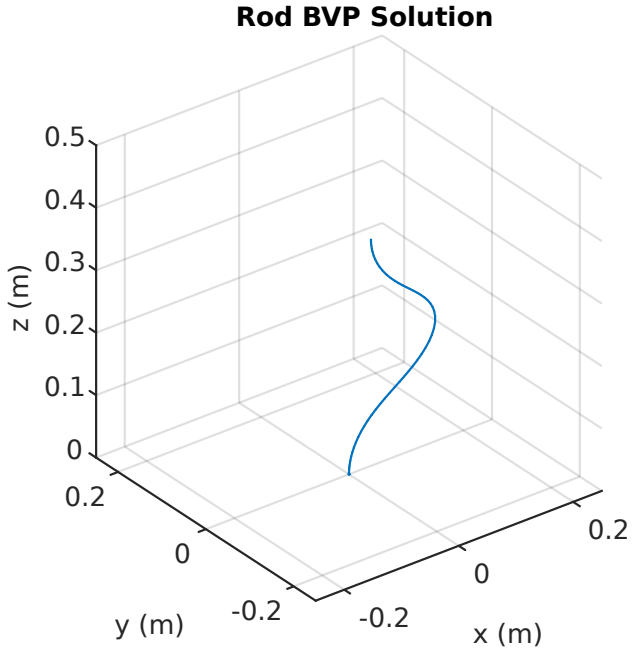


Figure 2.5: A rod BVP solution is visualized for $\boldsymbol{p}(L) = [0\ \ -0.1L\ \ 0.8]^\top$ and $\boldsymbol{R}(L) = \boldsymbol{I}$.

The code to solve the BVP scenario is shown in Appendix B. This specific boundary value problem is studied in greater detail by Bretl and McCarthy [12].

2.1.3 Orientation Represented by Quaternion

Truncation error from numerically integrating $\boldsymbol{R}(s) = \boldsymbol{R}(0) + \int_0^s \boldsymbol{R}_s(\sigma)d\sigma$ results in degeneracy so that $\boldsymbol{R}(s) \notin \mathrm{SO}(3)$. As discussed in [92], orientation may be represented using a quaternion $\boldsymbol{h} = h_1 + h_2 i + h_3 j + h_4 k$ which has a differential equation

$$\boldsymbol{h}_s = \frac{1}{2}\begin{bmatrix} 0 & -u_1 & -u_2 & -u_3 \\ u_1 & 0 & u_3 & -u_2 \\ u_2 & -u_3 & 0 & u_1 \\ u_3 & u_2 & -u_1 & 0 \end{bmatrix}\begin{bmatrix} h_1 \\ h_2 \\ h_3 \\ h_4 \end{bmatrix}. \tag{2.5}$$

The other equations involving $\boldsymbol{R}$ are unchanged; $\boldsymbol{R}(\boldsymbol{h})$ is found by

$$\boldsymbol{R}(\boldsymbol{h}) = \boldsymbol{I} + \frac{2}{\boldsymbol{h}^\top \boldsymbol{h}}\begin{bmatrix} -h_3^2 - h_4^2 & h_2h_3 - h_4h_1 & h_2h_4 + h_3h_1 \\ h_2h_3 + h_4h_1 & -h_2^2 - h_4^2 & h_3h_4 - h_2h_1 \\ h_2h_4 - h_3h_1 & h_3h_4 + h_2h_1 & -h_2^2 - h_3^2 \end{bmatrix}.$$

Integrating $\boldsymbol{h}_s$ ensures that $\boldsymbol{R}(\boldsymbol{h})$ conforms to SO(3). However, it is often the case that with a high-order integration method, the matrix $\boldsymbol{R}(L) = \boldsymbol{R}(0) + \int_0^L \boldsymbol{R}_s(s)ds$ is sufficiently near orthonormal. For example, the case in Appendix A with ODE integration using MATLAB's "ode45" implemented with a fourth-order Dormance-Prince scheme [100] has

$$\boldsymbol{R}(L)^\top \boldsymbol{R}(L) - \boldsymbol{I} = \begin{bmatrix} -0.47 & 3.92 & 4.23 \\ 3.92 & 6.19 & -0.06 \\ 4.23 & -0.06 & 5.40 \end{bmatrix} \times 10^{-9},$$

which is an acceptable deviation for many applications. The quaternion substitution has a tendency to further obfuscate the already complicated model, so the development here will typically rely on the rotation matrix differential equation while bearing in mind its potential

drawbacks, particularly if low-order integration methods are used. A MATLAB example of rod dynamics with orientation represented as a quaternion is shown in [109].

2.1.4 Local Frame Formulation

The rod ODEs (2.4) are often written with internal loading expressed in the local frame. To derive this system, local frame expressions of the internal loading are defined as $\boldsymbol{n}^b := \boldsymbol{R}^\top \boldsymbol{n}$ and $\boldsymbol{m}^b := \boldsymbol{R}^\top \boldsymbol{m}$. Recalling that skew-symmetric matrices satisfy $\widehat{\boldsymbol{u}}^\top = -\widehat{\boldsymbol{u}}$, the partial derivatives of the local-frame variables are obtained by

$$\begin{aligned}
(\boldsymbol{n}^b)_s &= \frac{\partial}{\partial s}(\boldsymbol{R}^\top \boldsymbol{n}) \\
&= \boldsymbol{R}_s^\top \boldsymbol{n} + \boldsymbol{R}^\top \boldsymbol{n}_s \\
&= (\boldsymbol{R}\widehat{\boldsymbol{u}})^\top \boldsymbol{n} - \boldsymbol{R}^\top \boldsymbol{f} \\
&= \widehat{\boldsymbol{u}}^\top \boldsymbol{R}^\top \boldsymbol{n} - \boldsymbol{R}^\top \boldsymbol{f} \\
&= -\widehat{\boldsymbol{u}}\boldsymbol{n}^b - \boldsymbol{R}^\top \boldsymbol{f}
\end{aligned}$$

and

$$\begin{aligned}
(\boldsymbol{m}^b)_s &= \frac{\partial}{\partial s}(\boldsymbol{R}^\top \boldsymbol{m}) \\
&= \boldsymbol{R}_s^\top \boldsymbol{m} + \boldsymbol{R}^\top \boldsymbol{m}_s \\
&= (\boldsymbol{R}\widehat{\boldsymbol{u}})^\top \boldsymbol{m} - \boldsymbol{R}^\top (\boldsymbol{p}_s \times \boldsymbol{n} + \boldsymbol{l}) \\
&= \widehat{\boldsymbol{u}}^\top \boldsymbol{R}^\top \boldsymbol{m} - \boldsymbol{R}^\top \boldsymbol{p}_s \times \boldsymbol{R}^\top \boldsymbol{n} - \boldsymbol{R}^\top \boldsymbol{l} \\
&= -\widehat{\boldsymbol{u}}\boldsymbol{m}^b - \boldsymbol{v} \times \boldsymbol{n}^b - \boldsymbol{R}^\top \boldsymbol{l}.
\end{aligned}$$

Thus the rod ODEs with internal loading in the local frame are

$$\boldsymbol{v} = \boldsymbol{v}^* + \boldsymbol{K}_{se}^{-1}\boldsymbol{n}^b$$
$$\boldsymbol{u} = \boldsymbol{u}^* + \boldsymbol{K}_{bt}^{-1}\boldsymbol{m}^b$$

$$\begin{aligned} \boldsymbol{p}_s &= \boldsymbol{R}\boldsymbol{v} \\ \boldsymbol{R}_s &= \boldsymbol{R}\widehat{\boldsymbol{u}} \\ (\boldsymbol{n}^b)_s &= -\widehat{\boldsymbol{u}}\boldsymbol{n}^b - \boldsymbol{R}^\top \boldsymbol{f} \\ (\boldsymbol{m}^b)_s &= -\widehat{\boldsymbol{u}}\boldsymbol{m}^b - \widehat{\boldsymbol{v}}\boldsymbol{n}^b - \boldsymbol{R}^\top \boldsymbol{l}. \end{aligned} \tag{2.6}$$

2.1.5 Effect of Shear and Extension

For slender rods, the shear and extensions strains are negligible. This results in $\boldsymbol{v} = \boldsymbol{e}_3$, and of course it is no longer necessary to use a constitutive equation to describe $\boldsymbol{v}$. Such a rod is known as a "Kirchhoff rod" because Gustav Kirchhoff formulated rod equations with these assumptions. For the spring steel rods described in this chapter, the difference between Cosserat and Kirchhoff rod solutions is slight, although the Cosserat model is used for generality.

2.2 Continuum Stewart-Gough Robot

The work presented in this section paraphrases the author's paper on the topic of continuum Stewart-Gough teleoperation [110] and summarizes subsequent developments to teleoperate a miniature robot [78, 77] and formulate a kinematic scheme which estimates an applied tip load using force measurements at the actuators [9]. Additional detail is provided along with new insights. Work prior to the author's involvement introduced the continuum Stewart-Gough (CSG) robot and formulated a kinematic model which treats the CSG as a set of multiple Cosserat rods with coupled boundary conditions [13]. The initial implementations of the model in [13] required several seconds to solve the forward or inverse kinematics with a shooting method. Given that the frequency range of voluntary hand motion in skilled

activities is roughly 4-7 hertz [28], inverse kinematics computation rates greater than a few hundred hertz should provide sufficient temporal resolution for human teleoperation. However, 1 kilohertz is a typical servo-loop rate for medical robots [52]. This section reviews the boundary value problem presented in [13] and details methods for the efficient numerical solution of this model at rates that enable real-time interactive simulation, motion planning, design optimization, and control. This fast implementation is used to teleoperate a prototype robot using real-time inverse kinematics solutions, and simulation tests show that inverse kinematics solutions are consistently computed at rates of several kilohertz using standard desktop computing hardware. The teleoperation scheme is subsequently applied to a miniature manipulator with a grasper.

2.2.1 Prototype Robot Design

To provide further context for the discussion of modeling and computational approaches, Fig. 2.6 shows a teleoperated parallel continuum robot system. The manipulator consists of six flexible rods (made of spring steel ASTM A228) connected in a parallel pattern similar to a traditional Stewart-Gough platform. The base frame is constructed from extruded aluminum beams (80/20® Inc.) and laser-cut acrylic plates. Linear actuators with potentiometer feedback (Firgelli Technologies Inc.) are attached to the aluminum beams and translate the base end of the legs such that their length between the platforms changes as each leg is extended or retracted through guide holes in the base platform. The actuators are attached to the rod bases with a shaft-collar connection that spins freely and cannot support torsion. The linear actuators are controlled by a custom PI control algorithm implemented on an Arduino Mega 2560 board which receives desired positions from a desktop computer through serial communication.

2.2.2 Boundary Conditions for Inverse Kinematics

The CSG is illustrated in Figure 2.7. Rods pass through holes in a base plate and join at an end effector. The rod variables are given a subscript i numbered from 1 to 6. The arclength

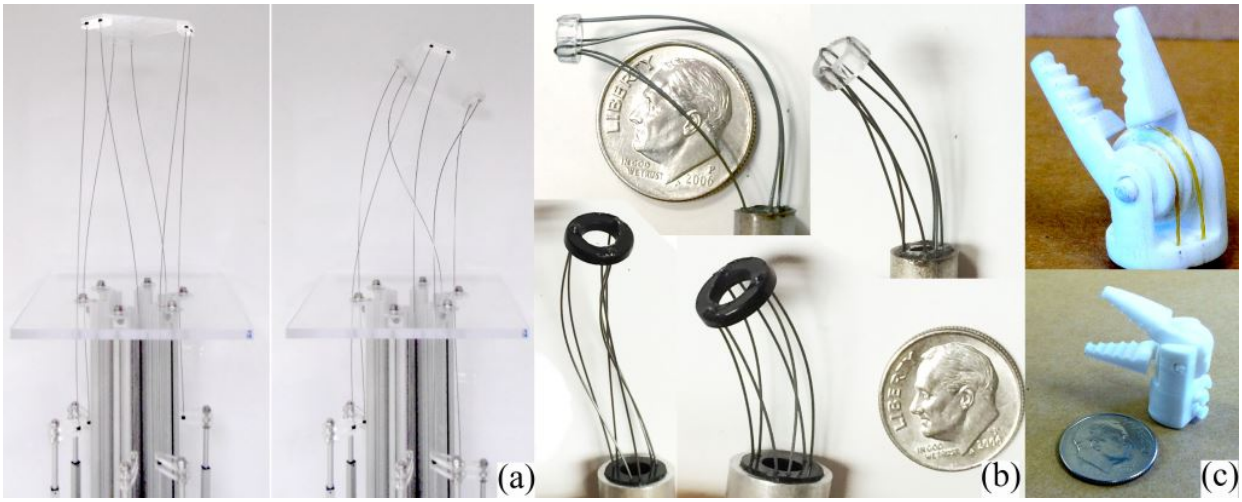


Figure 2.6: (a) A 400 mm tall benchtop prototype demonstrates 6 DOF motion. This system was used for model validation [13] and to implement the inverse kinematics algorithm for soft-real-time teleoperation [110]. (b) These small prototypes (10mm and 5mm diameter) show that parallel continuum manipulators can be scaled to sizes appropriate for manipulating endoscopic surgical tools. (c) These gripper designs for the smaller prototypes incorporate two tendon-driven jaws. The tendons are routed through hollow channels in the legs and are pulled by actuators outside the body of the robot.

of a rod from the base plate to the end effector is denoted by L_i. The portion of rod below the base plate is straight (absent buckling), and any elongation of rods below the base plate is neglected. Let L_i^* be the arclength L_i when the actuator is at its minimum stroke length $q_i = 0$, then L_i is found by

$$L_i = L_i^* + q_i. \tag{2.7}$$

Note that for the inverse kinematics problem, L_i is solved as part of the BVP then the motor displacement $q_i = L_i - L_i^*$ is calculated to control the motors.

The center of the base plate is arbitrarily assigned as the origin, and the lab-frame z-axis is normal to the base plate. The baseplate holes are specified in polar coordinates with a uniform radius $\mathfrak{R}$ from the center. The hexagonal Stewart-Gough pattern is described by a independent major angle α_1 and a minor angle $\alpha_2 = 120^\circ - \alpha_1$, so that the polar coordinate

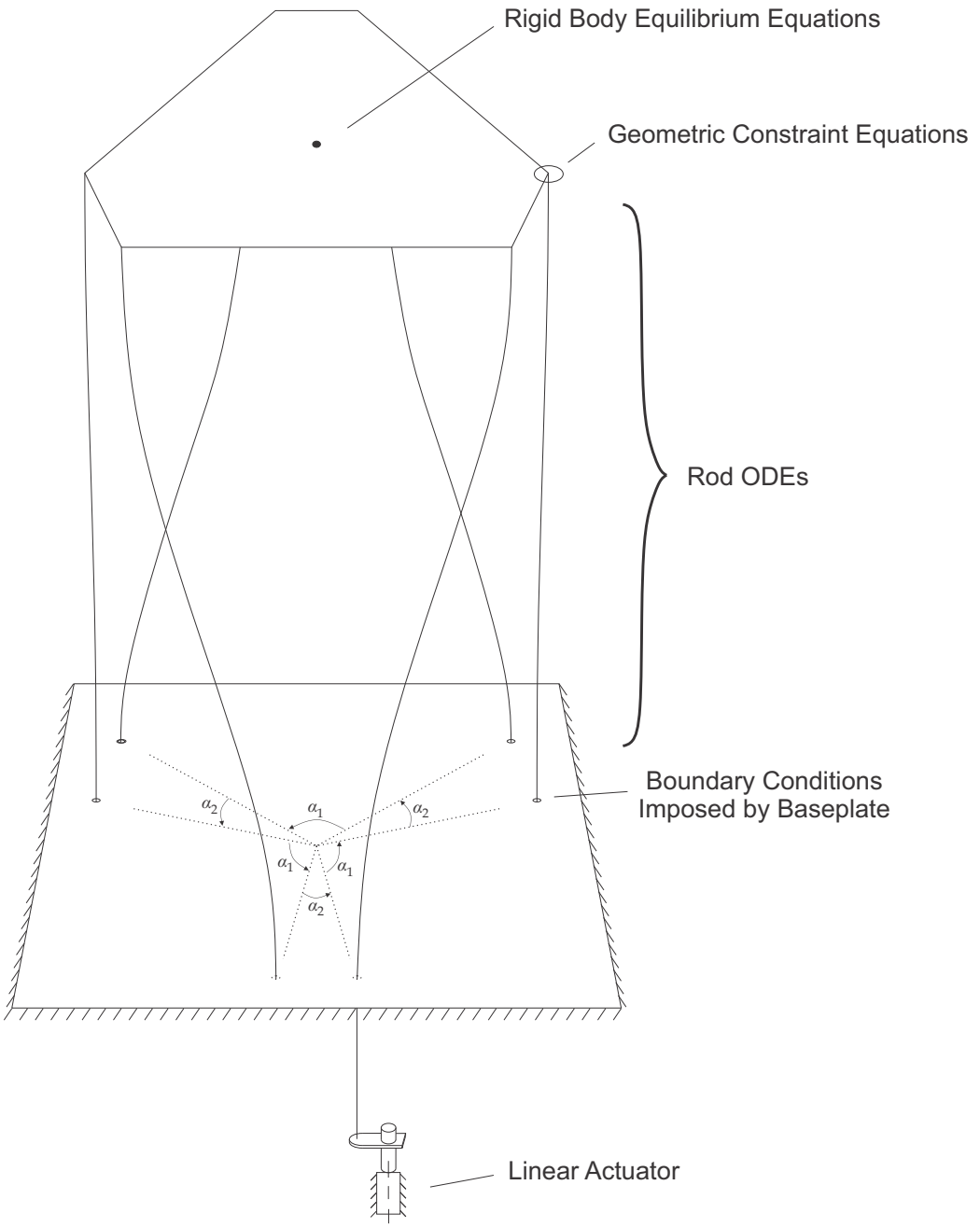


Figure 2.7: The continuum Stewart-Gough robot is shown with annotations indicating the boundary conditions. Each rod is modeled by the Cosserat rod equations with a linear-elastic constitutive equation (2.4). The end-effector joins the six rods resulting in attachment constraints (2.9) and (2.10), and static equilibrium equations (2.11). The hole pattern of the base plate imposes known pose boundary conditions (2.8) and unknown point loads on each rod. The length of a rod above the base plate is related to the linear actuator displacement (2.7), although only one actuator is illustrated here.

angles θ_i^B describing the location of holes in the baseplate are

$$\theta_i^B = \begin{cases} -\alpha_2/2, & i = 1 \\ \theta_{i-1}^B + \alpha_2, & i \in \{2, 4, 6\} \\ \theta_{i-1}^B + \alpha_1, & i \in \{3, 5\} \end{cases}.$$

This can be equivalently described by

$$\theta_i^B = -\alpha_2/2 + \frac{i - i\%2}{2}\alpha_2 + \frac{(i-1) - (i-1)\%2}{2}\alpha_1, \tag{2.8}$$

where % is the modulus operator. Figure 2.7 illustrates the base plate hole pattern with angles α_1 and α_2. The hole locations in Cartesian coordinates are

$$\boldsymbol{p}_i(0) = \mathfrak{R}\begin{bmatrix}\cos\theta_i^B & \sin\theta_i^B & 0\end{bmatrix}^\top.$$

The prototype has $\mathfrak{R} = 0.087$m and $\alpha_1 = 100°$. The end-effector has a similar attachment pattern, but the major and minor angle are switched, that is

$$\theta_i^E = -\alpha_1/2 + \frac{i - i\%2}{2}\alpha_1 + \frac{(i-1) - (i-1)\%2}{2}\alpha_2.$$

This defines a constant vector in the local end-effector frame from the platform centroid to a rod attachment point,

$$\boldsymbol{r}_i := \mathfrak{R}\begin{bmatrix}\cos\theta_i^E & \sin\theta_i^E & 0\end{bmatrix}^\top.$$

Thus given the end-effector centroid in global coordinates $\boldsymbol{p}_e$ and the rotation $\boldsymbol{R}_e$, there are position constraint equations

$$\boldsymbol{p}_i(L_i) = \boldsymbol{p}_e + \boldsymbol{R}_e\boldsymbol{r}_i. \tag{2.9}$$

For the prototype robot, the rod ends are constrained via shaft collars which allow rotation about the tangent axis, so the orientations are constrained by

$$\boldsymbol{R}_i(L_i)\boldsymbol{e}_3 = \boldsymbol{R}_e\boldsymbol{e}_3. \tag{2.10}$$

In addition to the geometric constraints, the end-effector is subject to rigid-body static equilibrium equations, which are given by

$$\begin{aligned} \boldsymbol{F} - \sum_{i=1}^{6} \boldsymbol{n}_i(L_i) &= \mathbf{0} \\ \boldsymbol{M} - \sum_{i=1}^{6} [\boldsymbol{m}_i(L_i) + \boldsymbol{R}_e^\top \boldsymbol{r}_i \times \boldsymbol{n}_i(L_i)] &= \mathbf{0}, \end{aligned} \tag{2.11}$$

where $\boldsymbol{F}$ and $\boldsymbol{M}$ are external force and moment vectors applied at the platform centroid. The weight of the platform can be accounted for by $\boldsymbol{F} = m\mathbf{g}$, but for the prototype robot the platform mass is negligible. The shaft collar attachments do not exert torque, that is

$$m_{i,z}^b(0) = m_{i,z}^b(L_i) = 0.$$

2.2.3 Solution with Shooting Method

The inverse kinematics problem has a known desired pose $\boldsymbol{p}_e$ and $\boldsymbol{R}_e$ and unknown arc lengths L_i which must be solved. Due to the geometric and equilibrium constraints imposed on the system, the inverse kinematics is a boundary value problem. It is possible to formulate a shooting method to solve the inverse kinematics BVP. There is not a unique choice of guessed variables and residual equations, but one sensible approach is to define the guess as the 30 unknown forces at the base and the 6 unknown arc lengths. The guess is expressed as a vector

$$\boldsymbol{G} := [\boldsymbol{n}_1^\top(0) \quad \boldsymbol{m}_{1_{xy}}^\top(0) \quad \boldsymbol{n}_2^\top(0) \quad \boldsymbol{m}_{2_{xy}}^\top(0) \quad ... \quad \boldsymbol{n}_6^\top(0) \quad \boldsymbol{m}_{6_{xy}}^\top(0) \quad L_1 \quad L_2 \quad ... \quad L_6]^\top. \tag{2.12}$$

The objective function error is

$$\boldsymbol{E}(\boldsymbol{G}) = [(\boldsymbol{E}_1^p)^\top \quad (\boldsymbol{E}_1^R)^\top \quad (\boldsymbol{E}_2^p)^\top \quad (\boldsymbol{E}_2^R)^\top \quad ... \quad (\boldsymbol{E}_6^p)^\top \quad (\boldsymbol{E}_6^R)^\top \quad (\boldsymbol{E}^F)^\top \quad (\boldsymbol{E}^M)^\top]^\top,$$

where error terms are defined based on (2.9), (2.10), and (2.11) so that the equations

$$\begin{aligned}
\boldsymbol{E}_i^p &= \boldsymbol{p}_e + \boldsymbol{R}_e \boldsymbol{r}_i - \boldsymbol{p}_i(L_i) \\
\boldsymbol{E}_i^R &= \begin{bmatrix} \boldsymbol{e}_1 & \boldsymbol{e}_2 \end{bmatrix}^\top \left[\boldsymbol{R}_e^\top \boldsymbol{R}_i(L_i) - \boldsymbol{R}_e \boldsymbol{R}_i^\top(L_i) \right]^\vee \\
\boldsymbol{E}^F &= \boldsymbol{F}_e - \sum_{i=1}^{6} \boldsymbol{n}_i(L_i) \\
\boldsymbol{E}^M &= \boldsymbol{M}_e - \sum_{i=1}^{6} \left[\boldsymbol{m}_i(L_i) + \boldsymbol{R}_e \boldsymbol{r}_i \times \boldsymbol{n}_i(L_i) \right]
\end{aligned} \tag{2.13}$$

provide contributions to the total error vector.

Thus a shooting method evaluation function taking 36 guesses and returning 36 errors has been formulated. An example MATLAB implementation of this shooting method is presented in Appendix C, which is used to find the inverse kinematics solution for a robot bending about the y-axis as shown in Figure 2.8.

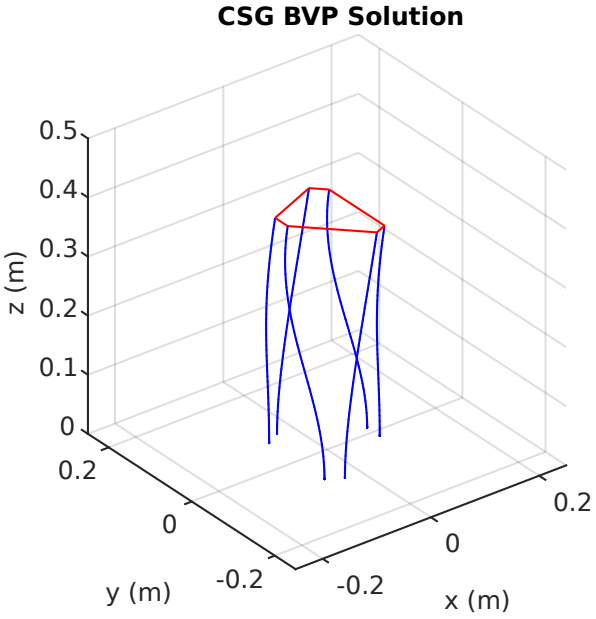


Figure 2.8: The solution of a boundary value problem for the continuum Stewart-Gough robot is visualized. The MATLAB code to create this plot is included in Appendix C.

2.2.4 Mathematical Considerations for Efficient Implementation

MATLAB is good for writing concise code because of its dynamic type system and monolithic design, but the MATLAB inverse kinematics function requires a few seconds to execute on current hardware. For teleoperation applications the model must be solved iteratively at a fast rate. There are three thrusts to reformulate the shooting method for real-time control. The example codes have relied on MATLAB functions for numerical integration using "ode45" and optimization with "fsolve", but the code should be rewritten in a compiled language[1]. Thus the first thrust is to decide how to implement the shooting method apart from the MATLAB interpreter. Next, a large part of the computational effort is spent on numerically integrating the rod equations. Thus the second thrust is to decrease the number of times a rod is numerically integrated, and the third is to decrease the computational effort of an individual rod integration.

Numerical Methods for the Shooting Problem

The high level logic of the shooting method is illustrated by Figure 2.9. Calculating the error evaluation function $\boldsymbol{E}(\boldsymbol{G})$ requires the integration of all rods from base to tip. The Cosserat rod equations are smooth ODEs, so it is appropriate to use a fixed-step high-order method, specifically the standard fourth-order-accurate Runge-Kutta scheme is chosen.

There is a significant implementation decision regarding how to calculate the shooting problem Jacobian $\boldsymbol{J} = \partial \boldsymbol{E}/\partial \boldsymbol{G}$. An initial implementation can rely on a naïve Jacobian calculation via first-order finite differences, that is

$$\boldsymbol{J}\boldsymbol{e}_i = \frac{\boldsymbol{E}(\boldsymbol{G} + \boldsymbol{e}_i\Delta) - \boldsymbol{E}(\boldsymbol{G})}{\Delta} + \mathcal{O}(\Delta) \quad \text{for } i = 1...36.$$

This approach is not optimally efficient, but it is a simple starting point. The nonlinear optimization routine is implemented by the Trust-Region-Dogleg algorithm [71], for which the Hessian is approximated by $\boldsymbol{B} = \boldsymbol{J}^\top \boldsymbol{J}$.

[1] MATLAB currently has limited support for compiling, but is mainly interpreted.

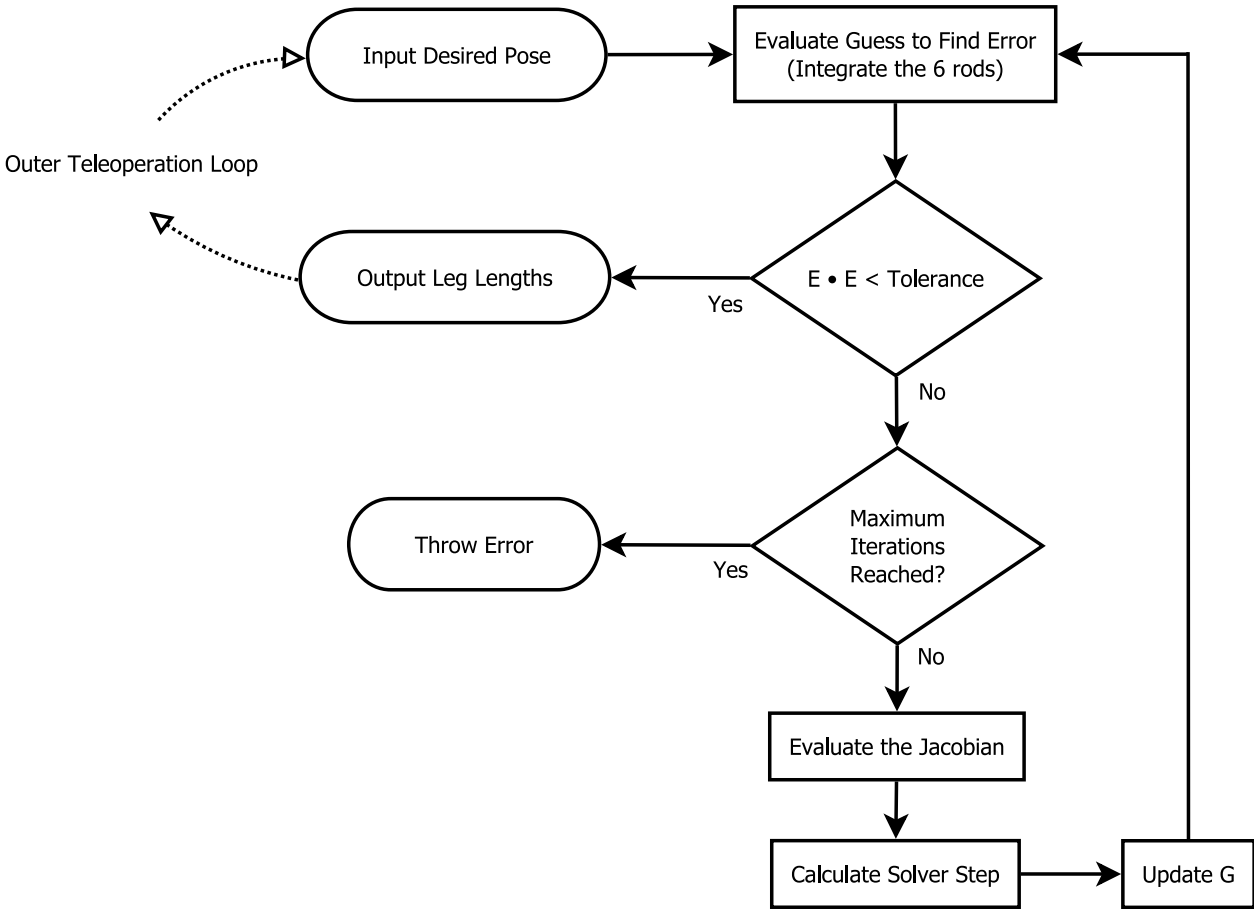


Figure 2.9: A shooting method is employed to solve the boundary value problem for the inverse kinematics. For teleoperation, the desired end-effector pose changes incrementally, so a good initial guess of the unknowns is available from the previous solution.

Decreasing Number of Rod Integrations

Exploitation of the model structure enables a significant reduction in the number of rod integrations required to evaluate the shooting method Jacobian. While the boundary conditions (and thus the eventual solutions) of the individual rod models are coupled through the static equilibrium condition in (2.11), changing a guessed variable in a rod i will not effect the behavior of a rod j for the shooting method. Variables are defined for the guessed proximal wrench of a rod $\boldsymbol{w}_i := [\boldsymbol{n}_i^\top(0) \quad \boldsymbol{m}_{i_{xy}}^\top(0)]^\top$, the geometric error $\boldsymbol{E}_i^G := [(\boldsymbol{E}_i^p)^\top \ (\boldsymbol{E}_i^R)^\top]^\top$, and the equilibrium error $\boldsymbol{E}^{eq} := [(\boldsymbol{E}^F)^\top \ (\boldsymbol{E}^M)^\top]^\top$. The Jacobian has the following form:

$$\boldsymbol{J} = \begin{bmatrix} \frac{\partial \boldsymbol{E}_1^G}{\partial \boldsymbol{w}_1} & 0 & & 0 & \frac{\partial \boldsymbol{E}_1^G}{\partial L_1} & 0 & & 0 \\ 0 & \frac{\partial \boldsymbol{E}_2^G}{\partial \boldsymbol{w}_2} & & 0 & 0 & \frac{\partial \boldsymbol{E}_2^G}{\partial L_2} & & 0 \\ 0 & 0 & \ddots & 0 & 0 & 0 & \ddots & 0 \\ 0 & 0 & & \frac{\partial \boldsymbol{E}_6^G}{\partial \boldsymbol{w}_6} & 0 & 0 & & \frac{\partial \boldsymbol{E}_6^G}{\partial L_6} \\ \frac{\partial \boldsymbol{E}^{eq}}{\partial \boldsymbol{w}_1} & \frac{\partial \boldsymbol{E}^{eq}}{\partial \boldsymbol{w}_2} & \cdots & \frac{\partial \boldsymbol{E}^{eq}}{\partial \boldsymbol{w}_6} & \frac{\partial \boldsymbol{E}^{eq}}{\partial L_1} & \frac{\partial \boldsymbol{E}^{eq}}{\partial L_2} & \cdots & \frac{\partial \boldsymbol{E}^{eq}}{\partial L_6} \end{bmatrix} \tag{2.14}$$

For the purposes of the objective function, a change to the guessed variables pertaining to a rod i impact only the distal states of the rod i. Thus only five integrations of rod 1 are needed to find columns 1-5, five integrations of rod 2 for columns 6-10, and so forth to column 30. The partial derivatives with respect to arc length in columns 31-36 may be solved without additional integrations by referring to the rod ODEs. For example, the partial derivative of a position error is

$$\frac{\partial \boldsymbol{E}_1^p}{\partial L_1} = \frac{\partial}{\partial L_1}[\boldsymbol{p}_e + \boldsymbol{R}_e \boldsymbol{r}_1 - \boldsymbol{p}_1(L_1)] = -\boldsymbol{p}_{s,1}(L_1) = -\boldsymbol{R}_1(L_1)\boldsymbol{v}_1(L_1).$$

The numerically integrated values of $\boldsymbol{R}_1(L_1)$ and $\boldsymbol{v}_1(L_1)$ are calculated while evaluating $\boldsymbol{E}(\boldsymbol{G})$ prior to the Jacobian calculation. The entire Jacobian then requires only 30 rod integrations. This is in contrast with the naïve approach which required 218 integrations (36 evaluations of $\boldsymbol{E}(\boldsymbol{G})$ with 6 rod integrations each).

In addition to improving the Jacobian calculation, seeding the shooting method initial guess with a previous solution means that the first evaluation of $\boldsymbol{E}(\boldsymbol{G})$ does not require any rod integrations. Changes in $\boldsymbol{p}_e$ and $\boldsymbol{R}_e$ between model solutions will lead to a change in $\boldsymbol{E}$, but re-integrating the rods is only necessary when $\boldsymbol{G}$ changes, assuming the distal rod states are persistently stored. The number of rod integrations can also be reduced by improving the solver convergence rate. This is the main motivation for using the Trust-Region-Dogleg algorithm here, whereas the original conference paper [110] used a Levenberg-Marquardt method. Of course multithreading the rod integrations will also improve the solution speed by decreasing the number of rod integrations per processor core, but from a mathematical standpoint this is rather straightforward as the integrations are embarrassingly parallel.

Improving Speed of Rod Integrations

It can be shown that the robot design results in an absence of torsion throughout each rod, which results in mathematical simplifications to the rod ODEs. The rods are straight absent applied loads so that $\boldsymbol{v}^* = \boldsymbol{e}_3$ and $\boldsymbol{u}^* = \boldsymbol{0}$. The constitutive equation is reduced to

$$\begin{aligned}\boldsymbol{v} &= \boldsymbol{K}_{se}^{-1}\boldsymbol{n}^b + \boldsymbol{e}_3\\ \boldsymbol{u} &= \boldsymbol{K}_{bt}^{-1}\boldsymbol{m}^b.\end{aligned}$$

There is no distributed moment, $\boldsymbol{l} = \boldsymbol{0}$, so the differential equation for moment is given by

$$(\boldsymbol{m}^b)_s = -\widehat{\boldsymbol{u}}\boldsymbol{m}^b - \widehat{\boldsymbol{v}}\boldsymbol{n}^b.$$

The first term is expanded by

$$-\widehat{\boldsymbol{u}}\boldsymbol{m}^b = \begin{bmatrix} m_2^b m_3^b \left(GI_{zz} - EI_{yy}\right)^{-1} \\ m_1^b m_3^b \left(EI_{xx} - GI_{zz}\right)^{-1} \\ m_1^b m_2^b \left(EI_{yy} - EI_{xx}\right)^{-1} \end{bmatrix} = \begin{bmatrix} -m_2^b \left(EI_{xx} - GI_{zz}\right)^{-1} m_3^b \\ m_1^b \left(EI_{xx} - GI_{zz}\right)^{-1} m_3^b \\ 0 \end{bmatrix}_{|I_{xx}=I_{yy}}.$$

The third component is zero since the rods are circular with homogenous material properties. For the second term, $\boldsymbol{v}$ is split into its components by the constitutive equation,

$$-\boldsymbol{v} \times \boldsymbol{n}^b = -(\boldsymbol{K}_{se}^{-1}\boldsymbol{n}^b) \times \boldsymbol{n}^b - \boldsymbol{e}_3 \times \boldsymbol{n}^b,$$

then expand so that

$$-\boldsymbol{v} \times \boldsymbol{n}^b = \begin{bmatrix} n_2^b \left(EA - GA\right)^{-1} n_3^b \\ -n_1^b \left(EA - GA\right)^{-1} n_3^b \\ 0 \end{bmatrix} + \begin{bmatrix} n_2^b \\ -n_1^b \\ 0 \end{bmatrix}.$$

Thus the partial derivative of torsion is zero, and since the shaft collars do not support torsion, there is no torsion along the whole rod, that is $m_3^b(s) = 0$ and also $u_3(s) = 0 \ \ \forall s$. The reduced system without torsion is

$$\begin{aligned} \boldsymbol{v} &= \boldsymbol{K}_{se}^{-1}\boldsymbol{n}^b + \boldsymbol{e}_3 \\ \boldsymbol{u}_{xy} &= \boldsymbol{m}_{xy}^b/(EI) \end{aligned}$$

$$\begin{aligned} \boldsymbol{p}_s &= \boldsymbol{R}\boldsymbol{v} \\ \boldsymbol{R}_s &= \begin{bmatrix} -\boldsymbol{R}\boldsymbol{e}_3 u_2 & \boldsymbol{R}\boldsymbol{e}_3 u_1 & \boldsymbol{R}\boldsymbol{e}_1 u_2 - \boldsymbol{R}\boldsymbol{e}_2 u_1 \end{bmatrix} \\ (\boldsymbol{n}^b)_s &= \begin{bmatrix} -u_2 n_3^b & u_1 n_3^b & u_2 n_1^b - u_1 n_2^b \end{bmatrix}^\top - \boldsymbol{R}^\top \rho A \boldsymbol{g} \\ (\boldsymbol{m}_{xy}^b)_s &= [1 + (EA - GA)^{-1} n_3^b] \begin{bmatrix} n_2^b & -n_1^b \end{bmatrix}^\top. \end{aligned} \tag{2.15}$$

This statement of the ODEs requires fewer operations and less memory.

If the weight is neglected, there is actually a known analytical solution to the ODE involving elliptic integrals [101], and this solution is significantly simpler when combined with the assumption of negligible shear and extension strains [102, 70]. Here weight is considered and the ODE system (2.15) is used to model the rods.

2.2.5 Implementation of Teleoperable System

The prototype robot was successfully teleoperated as captured on video, a still shot of which is shown in Figure 2.10. The model prediction qualitatively matches the prototype robot shape, although we note that a more rigorous validation of model accuracy is presented later in Section 2.2.8.

The initial shooting method implementation in MATLAB (Appendix C) typically solves the model in about five seconds using the "ode45" and "fsolve" functions and the naïve Jacobian computation. For teleoperation the shooting method was implemented in C++ using the Eigen matrix library [37]. A rod integration routine solves (2.15) as an initial value problem using RK4. The Trust-Region-Dogleg routine relies heavily on Eigen for matrix multiplications and decompositions. Although the Jacobian has many zero elements, its dimensions are small compared to typical sparse problems, and $\boldsymbol{J}^\top \boldsymbol{J}$ is dense, so the matrix data structures are dense.

A MATLAB teleoperation loop calls the C++ shooting method code. This MATLAB script also performs serial communication to command the motors, reads input from an Xbox 360 controller, and visualizes the model solution for comparison to the actual robot. If the joint commands are within an achievable range, MATLAB sends the commands to an Arduino MEGA 2560 which performs low-level PI control of linear actuators using potentiometer feedback. With this system users are able to manipulate the end effector to a desired pose effectively.

2.2.6 Real-Time Modeling Results

In order to evaluate the run-time efficiency of the model, there were two sets of simulations which measured the computation time required to obtain inverse kinematics solutions. These time trials are not "hardware-in-the-loop" simulations; they are intended to assess the shooting method's efficiency. The design parameters of the simulated robot match the prototype. The first simulation investigates the model's ability to solve at real-time rates for teleoperation, where the changes in position and orientation of the end effector will be

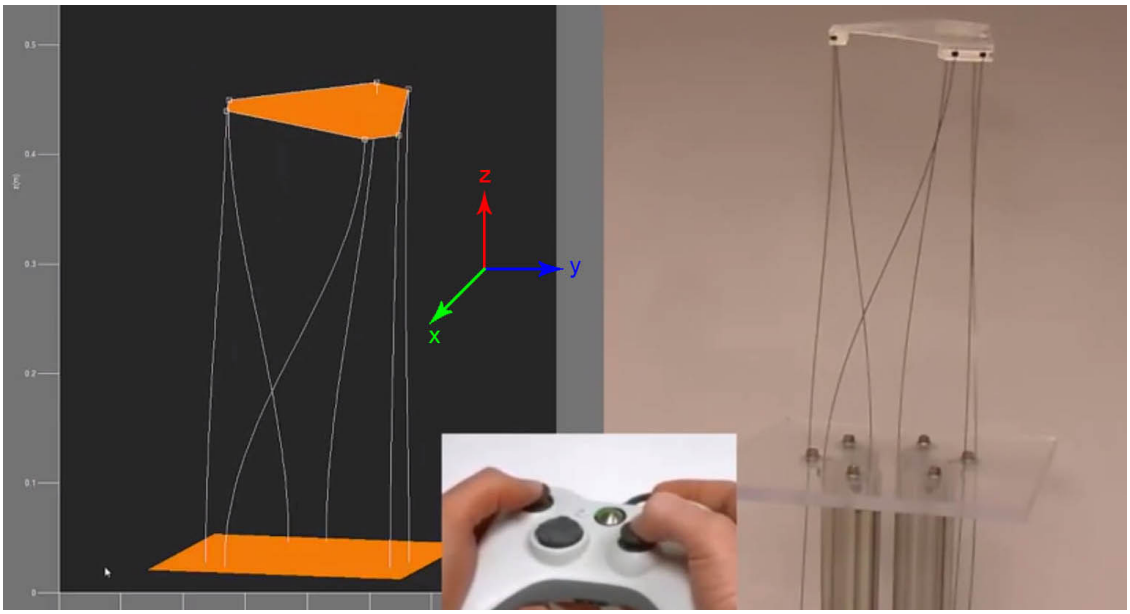


Figure 2.10: A visualization of the numerical solution is shown on the left and the physical manipulator on the right. The input device in the middle gives the user 6 DOF control of the manipulator in real-time.

small between solutions and the previous solution can be used as the initial guess for the unknown parameters. The second simulation set measures the time required for the model to solve when the initial guess is far from the actual solution. This task may be necessary for motion planning and design optimization. In both simulations, the rods were integrated using forty steps, and the run-time for a single inverse kinematics solution was calculated as the average time over the course of many runs.

Solution Speed for Teleoperation

In the case of teleoperation, the preceding pose of the end effector is close to the current pose. Our simulation mimics this with a simple translation movement of the desired pose in y and z directions. The end-effector begins at [0 0.02 0.48] m, moves to [0 0.12 0.58] m over the course of 100 solves, and returns to [0 0.02 0.48] m in 100 solves, thus moving 1.4 mm between solves. The cycle continues until one million solves are reached. The termination criteria for the Trust-Region-Dogleg scheme was that the error sum of squares $\|\boldsymbol{E}\|^2$ be less than 1e-6. The resulting timing statistics for this simulation set are presented in Figure 2.11.

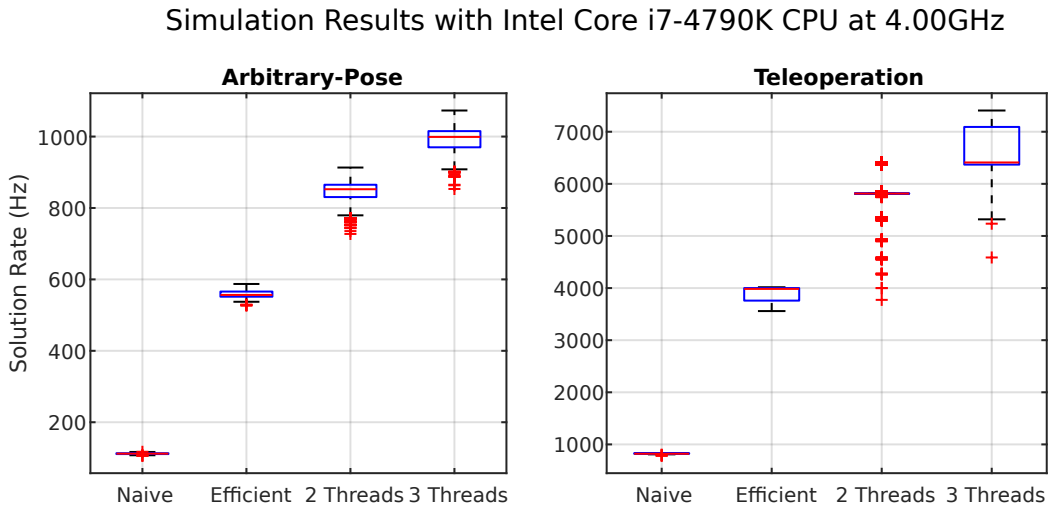


Figure 2.11: The solution speeds for the teleoperation and arbitrary-pose scenarios are shown. Because a single model solution is a short-lived process, each boxplot data point is the average solution frequency of 1000 solves, and each boxplot has 1000 data points, a total of one million solves per simulation.

Solution Speed for an Arbitrary Pose

In the case of an arbitrary desired pose of the end effector, the model solved for a position and orientation chosen randomly from a uniform distribution within the manipulator's workspace. The desired position was limited to a cylindrical volume around the nominal end effector position with 20 mm radius and a height of 200 mm. For the desired orientation, ZYZ Euler angles were sampled randomly from $-10°$ to $10°$. These random numbers were obtained using the standard C++ "rand()" function. One-thousand average timing results were measured, each one averaged over one-thousand model solutions. Results are presented in Figure 2.11.

Discussion

The results demonstrate that the shooting method is an effective discretization strategy, particularly when attention is paid to the implementation details. The ability to solve the inverse kinematics BVP efficiently enables model-based control. Contrasting the teleoperation simulation to the arbitrary-pose simulation, there is a significant difference in the effort required to reach a solution for these different tasks, which indicates that the

performance depends on the nearness of the initial guess to the actual solution. Although there are immediately diminishing returns from multithreading, the benefit from adding a thread is significant.

Continuum robot kinematics have typically been solved on desktop machines, but the approach here is amenable to smaller form factors. The single-threaded teleoperation simulation also solved at a rate of about 350Hz on a Raspberry Pi 3B, a credit-card sized single-board computer with a mass of just 42 grams and cost of $30.

2.2.7 Forward Kinematics and Force Sensing Models[2]

The forward kinematics problem differs from the inverse kinematics because the leg lengths L_i are known and the end-effector pose variables $\boldsymbol{p}_e$ and $\boldsymbol{R}_e$ are unknown. The shooting method in [13] defines a guess

$$\boldsymbol{G} = [\boldsymbol{n}_1^\top(0) \quad \boldsymbol{m}_{1_{xy}}^\top(0) \quad \boldsymbol{n}_2^\top(0) \quad \boldsymbol{m}_{2_{xy}}^\top(0) \quad ... \quad \boldsymbol{n}_6^\top(0) \quad \boldsymbol{m}_{6_{xy}}^\top(0)]^\top,$$

with an error

$$\boldsymbol{E}(\boldsymbol{G}) = [(\boldsymbol{E}_2^p)^\top \quad (\boldsymbol{E}_2^R)^\top \quad ... \quad (\boldsymbol{E}_6^p)^\top \quad (\boldsymbol{E}_6^R)^\top \quad (\boldsymbol{E}^F)^\top \quad (\boldsymbol{E}^M)^\top]^\top,$$

where the subterms were defined in (2.13). The geometric constraints on the first rod are not part of the error, but rather are embedded into the objective function by specifying the end-effector pose, that is

$$\begin{aligned} \boldsymbol{p}_e &= \boldsymbol{p}_1(L_1) - \boldsymbol{R}_e \boldsymbol{r}_1 \\ \boldsymbol{R}_e &= \boldsymbol{R}_1(L_1). \end{aligned}$$

This results in a square 30x30 optimization problem. One can also include the end-effector pose in the guess and include the first rod's geometric constraints in the error as in [9] for a

[2]Caroline B. Black is lead author of related publications. The author of this dissertation was a supporting author.

36x36 optimization problem with a sparser Jacobian, but in either case $\boldsymbol{J}$ can be obtained by finite differences with 30 rod integrations.

If the linear actuators are equipped with axial force sensors, it is possible to determine unknown end-effector loads [122, 9]. The axial forces $n_{i,z}(0)$ are measured so that the shooting method guess only contains transverse forces, that is

$$\boldsymbol{G} = [\boldsymbol{n}_{1,xy}^\top(0) \quad \boldsymbol{m}_{1,xy}^\top(0) \quad \boldsymbol{n}_{2,xy}^\top(0) \quad \boldsymbol{m}_{2,xy}^\top(0) \quad ... \quad \boldsymbol{n}_{6,xy}^\top(0) \quad \boldsymbol{m}_{6,xy}^\top(0)]^\top.$$

The error is then

$$\boldsymbol{E}(\boldsymbol{G}) = [(\boldsymbol{E}_2^p)^\top \quad (\boldsymbol{E}_2^R)^\top \quad ... \quad (\boldsymbol{E}_6^p)^\top \quad (\boldsymbol{E}_6^R)^\top]^\top,$$

where the equilibrium constraints $\boldsymbol{E}^F$ and $\boldsymbol{E}^M$ are no longer included because the end-effector loads are solved by

$$\begin{aligned}\boldsymbol{F}_e &= \sum_{i=1}^{6} \boldsymbol{n}_i(L_i) \\ \boldsymbol{M}_e &= \sum_{i=1}^{6} \boldsymbol{m}_i(L_i) + \boldsymbol{R}_e \boldsymbol{r}_i \times \boldsymbol{n}_i(L_i).\end{aligned}$$

Such force sensing schemes have been experimentally validated for continuum robots [122]. Force sensing specifically for CSGs was considered in [9].

2.2.8 Miniature Manipulator and Accuracy Validation[3]

The real-time kinematics implementation facilitated teleoperation and experimental model validation of a surgical-scale prototype robot with a grasper [78, 77]. The robot system is shown in Figure 2.12. Rods were attached by epoxy rather than torsion shafts so that the joints support torsion, which is trivial to account for in the shooting method by guessing the proximal torques $m_{i,z}(0)$ and including the full orientation error for each rod $\boldsymbol{E}_i^R = \left[\boldsymbol{R}_e^\top \boldsymbol{R}_i(L_i) - \boldsymbol{R}_e \boldsymbol{R}_i^\top(L_i)\right]^\vee$. The rod material was Nitinol, a nickel-titanium alloy commonly

[3] Andrew Orekhov is lead author of related publications. The author of this dissertation was a supporting author.

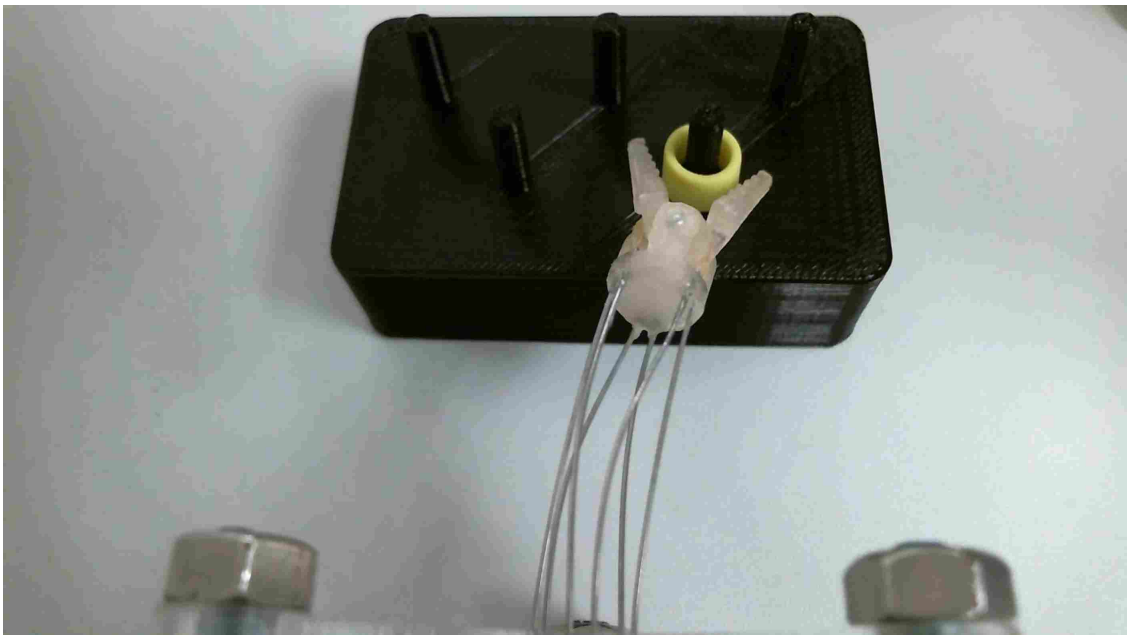


Figure 2.12: A webcam feed was used during the pick-and-place task to simulate a teleoperated training procedure.

used in medical applications due to its biocompability. The stress-strain relationship for Nitinol is nonlinear and involves hysteresis [125], but is often approximated by a linear elastic relation [94], which was the approach used for the model analysis here.

The accuracy validation used a "MicronTracker H3-60" stereoscopic camera system to measure the end-effector pose. The model-predicted position was compared to the measured position by the metric

$$\%\text{Error} = \frac{\|\boldsymbol{p}_{\text{model}} - \boldsymbol{p}_{\text{measured}}\|}{42\text{mm}} \times 100\%,$$

where 42mm is the length of the robot in the home configuration. The average position error was 2.83% over 37 tested poses, with a maximum error of 7.26% and a minimum of 0.31% [77]. This indicates that the model prediction is acceptably close to the actual robot behavior. Factors affecting the model accuracy include construction tolerances and model approximation errors. There are two sources of approximation error from the shooting method– the $\mathcal{O}(ds^4)$ global error of a 4^{th}-order Runge-Kutta routine and the error allowed

from the optimization routine's cutoff tolerance. Other model approximations include the linear-elastic constitutive equation and the one-dimensional idealization of the rods. Overall the results are reasonable and demonstrate that Cosserat rod theory provides a fitting description of parallel continuum robots.

2.3 Conclusions

This chapter derived the static Cosserat rod equations and extensively considered their application to continuum Stewart-Gough robots. Although the CSG BVP is complicated, it can be efficiently solved by the shooting method with standard desktop computing hardware. The developed CSG model was applied to teleoperate two prototype robots and to generate simulation results.

The teleoperated robots moved reasonably quickly without violating the assumption of quasi-static motion. There are important scenarios where the quasi-static assumption is invalid, for example if there were significant inertia at the CSG end-effector. In the next chapter, the statement of Cosserat rod equations is extended to include dynamic effects.

Chapter 3

Dynamics

The equations of motion for continuum robots are useful to simulate dynamic behavior, which allows one to verify the suitability of quasi-static control schemes, and is a prerequisite for dynamic control. In this chapter we review the derivation of the dynamic Cosserat rod partial differential equations, present discretization strategies, and simulate various continuum robots, including continuum Stewart Gough robots, tendon-driven robots, fluidic actuators, and concentric tubes.

The developments in Sections 3.1.2 - 3.4 are based on a conference paper describing an implicit time semi-discretization for elastic rod dynamics [112] and an accepted paper describing continuum robot dynamics with co-author Vince Aloi [109]. The tendon-robot boundary conditions presented in Section 3.3.2 include results from Kaitlin Oliver-Butler's paper describing the effect of tendon routing paths on robot tip stiffness [73], of which the author of this dissertation was a supporting author. The model of concentric tube dynamics in Section 3.5 is based on a submitted paper [111] in collaboration with Katy Riojas and Bob Webster of Vanderbilt University.

3.1 Individual Rod

3.1.1 Derivation of Dynamic Cosserat Rod Equations

Whereas variables were previously functions of arc length only, we now recognize dependence on time. For example, the centerline is described by $\boldsymbol{p}(t, s)$ as shown in Figure 3.1. We set out to derive the dynamic equations of motion. As in the static case, a similar derivation can be found in Antman [2]. We define variables for the local-frame velocity and angular velocity

$$\begin{aligned} \boldsymbol{q}(t, s) &:= \boldsymbol{R}^{\top} \boldsymbol{p}_t \\ \boldsymbol{\omega}(t, s) &:= \left(\boldsymbol{R}^{\top} \boldsymbol{R}_t\right)^{\vee}, \end{aligned}$$

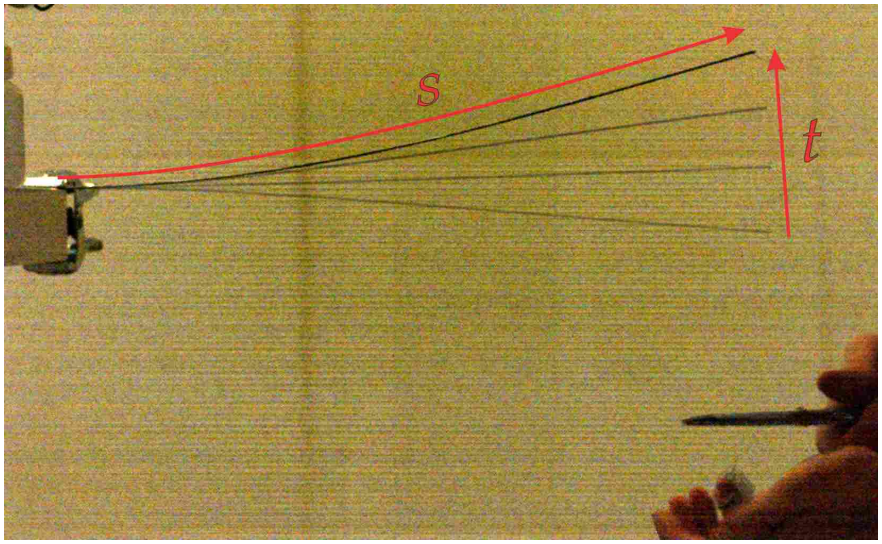


Figure 3.1: The dynamic rod description includes dimensions for time and arclength so that the centerline is a function $\boldsymbol{p}(t, s)$, as illustrated by this time-lapse photo in which removing a tip weight results in dynamic motion.

which is analogous to how $\boldsymbol{v}$ and $\boldsymbol{u}$ were defined as

$$\begin{aligned}\boldsymbol{v}(t,s) &:= \boldsymbol{R}^\top \boldsymbol{p}_s \\ \boldsymbol{u}(t,s) &:= \left(\boldsymbol{R}^\top \boldsymbol{R}_s\right)^\vee.\end{aligned}$$

As in [97], we derive a compatibility equation for $\boldsymbol{q}_s$ by taking the partial derivative

$$\boldsymbol{q}_s = \boldsymbol{R}_s^\top \boldsymbol{p}_t + \boldsymbol{R}^\top \boldsymbol{p}_{ts}$$

and obtain $\boldsymbol{p}_{ts}$ by taking the partial derivative of $\boldsymbol{p}_s = \boldsymbol{R}\boldsymbol{v}$, that is

$$\boldsymbol{p}_{ts} = \boldsymbol{R}_t \boldsymbol{v} + \boldsymbol{R}\boldsymbol{v}_t.$$

We substitute this expression for $\boldsymbol{p}_{ts}$, then also substitute for $\boldsymbol{p}_t = \boldsymbol{R}\boldsymbol{q}$, $\boldsymbol{R}_t = \boldsymbol{R}\widehat{\boldsymbol{\omega}}$, and $\boldsymbol{R}_s = \boldsymbol{R}\widehat{\boldsymbol{u}}$ to find

$$\begin{aligned}\boldsymbol{q}_s &= \boldsymbol{R}_s^\top \boldsymbol{p}_t + \boldsymbol{R}^\top \left(\boldsymbol{R}_t \boldsymbol{v} + \boldsymbol{R}\boldsymbol{v}_t\right) \\ &= \left(\boldsymbol{R}\widehat{\boldsymbol{u}}\right)^\top \boldsymbol{R}\boldsymbol{q} + \boldsymbol{R}^\top \left(\boldsymbol{R}\widehat{\boldsymbol{\omega}}\boldsymbol{v} + \boldsymbol{R}\boldsymbol{v}_t\right) \\ &= -\widehat{\boldsymbol{u}}\boldsymbol{q} + \widehat{\boldsymbol{\omega}}\boldsymbol{v} + \boldsymbol{v}_t.\end{aligned}$$

Solving for $\boldsymbol{\omega}_s$ requires a similar process where we take the partial derivative of angular velocity. We work with the skew-symmetric representation to simplify the solution process so that

$$\begin{aligned}\widehat{\boldsymbol{\omega}}_s &= \frac{\partial}{\partial s}\boldsymbol{R}^\top \boldsymbol{R}_t \\ &= \boldsymbol{R}_s^\top \boldsymbol{R}_t + \boldsymbol{R}^\top \boldsymbol{R}_{ts},\end{aligned}$$

then solve the mixed partial derivative term by differentiating $\boldsymbol{R}_s = \boldsymbol{R}\widehat{\boldsymbol{u}}$ to obtain

$$\boldsymbol{R}_{ts} = \boldsymbol{R}_t \widehat{\boldsymbol{u}} + \boldsymbol{R}\widehat{\boldsymbol{u}}_t.$$

We obtain a reduced expression for $\widehat{\boldsymbol{\omega}}_s$ by substituting and simplifying:

$$\begin{aligned}\widehat{\boldsymbol{\omega}}_s &= \boldsymbol{R}_s^\top \boldsymbol{R}_t + \boldsymbol{R}^\top \left(\boldsymbol{R}_t \widehat{\boldsymbol{u}} + \boldsymbol{R}\widehat{\boldsymbol{u}}_t\right) \\ &= \left(\boldsymbol{R}\widehat{\boldsymbol{u}}\right)^\top \boldsymbol{R}\widehat{\boldsymbol{\omega}} + \boldsymbol{R}^\top \left(\boldsymbol{R}\widehat{\boldsymbol{\omega}}\widehat{\boldsymbol{u}} + \boldsymbol{R}\widehat{\boldsymbol{u}}_t\right) \\ &= -\widehat{\boldsymbol{u}}\widehat{\boldsymbol{\omega}} + \widehat{\boldsymbol{\omega}}\widehat{\boldsymbol{u}} + \widehat{\boldsymbol{u}}_t.\end{aligned}$$

It can be verified that $\left(-\widehat{\boldsymbol{u}}\widehat{\boldsymbol{\omega}} + \widehat{\boldsymbol{\omega}}\widehat{\boldsymbol{u}}\right)^\vee = -\widehat{\boldsymbol{u}}\boldsymbol{\omega}$, so we finally obtain

$$\boldsymbol{\omega}_s = -\widehat{\boldsymbol{u}}\boldsymbol{\omega} + \boldsymbol{u}_t.$$

Now we need to revisit the equilibrium equations we derived in Section 2.1.1 to include dynamic terms. The dynamic force balance for a slice of rod is

$$\boldsymbol{n}(s+\delta) - \boldsymbol{n}(s) + \int_s^{s+\delta} \boldsymbol{f}(\sigma) d\sigma = \int_s^{s+\delta} \rho A \boldsymbol{p}_{tt} d\sigma,$$

where every variable in the kernel on the right-hand side is a function of σ and dependence on t has been omitted for clarity. We differentiate to obtain

$$\boldsymbol{n}_s = \rho A \boldsymbol{p}_{tt} - \boldsymbol{f}.$$

Now consider the dynamic moment balance equation for a section of rod. The rate of change of angular momentum is $\frac{\partial}{\partial t}\left(\boldsymbol{R}\rho\boldsymbol{J}\boldsymbol{\omega}\right)$, so that the moment balance equation is

$$\begin{aligned}&\boldsymbol{m}(s+\delta) - \boldsymbol{m}(s) + \boldsymbol{p}(s+\delta) \times \boldsymbol{n}(s+\delta) - \boldsymbol{p}(s) \times \boldsymbol{n}(s) + \int_s^{s+\delta} \left[\boldsymbol{l}(\sigma) + \boldsymbol{p}(\sigma) \times \boldsymbol{f}(\sigma)\right] d\sigma \\ &= \int_s^{s+\delta} \frac{\partial}{\partial t}\left(\boldsymbol{R}\rho\boldsymbol{J}\boldsymbol{\omega}\right) d\sigma,\end{aligned}$$

once again omitting the σ argument. Differentiating and canceling terms as in Section 2.1.1 leads to an equation for $\boldsymbol{m}_s$, that is

$$\boldsymbol{m}_s = \frac{\partial}{\partial t}\left(\boldsymbol{R}\rho\boldsymbol{J}\boldsymbol{\omega}\right) - \widehat{\boldsymbol{p}}_s \boldsymbol{n} - \boldsymbol{l}.$$

Finally we have derived all the terms of the dynamic Cosserat rod equations. The whole system of PDEs is

$$\begin{aligned}
\boldsymbol{p}_s &= \boldsymbol{R}\boldsymbol{v} \\
\boldsymbol{R}_s &= \boldsymbol{R}\widehat{\boldsymbol{u}} \\
\boldsymbol{n}_s &= \rho A \boldsymbol{R}\left(\widehat{\boldsymbol{\omega}}\boldsymbol{q} + \boldsymbol{q}_t\right) - \boldsymbol{f} \\
\boldsymbol{m}_s &= \rho \boldsymbol{R}\left(\widehat{\boldsymbol{\omega}}\boldsymbol{J}\boldsymbol{\omega} + \boldsymbol{J}\boldsymbol{\omega}_t\right) - \widehat{\boldsymbol{p}}_s\boldsymbol{n} - \boldsymbol{l} \\
\boldsymbol{q}_s &= \boldsymbol{v}_t - \widehat{\boldsymbol{u}}\boldsymbol{q} + \widehat{\boldsymbol{\omega}}\boldsymbol{v} \\
\boldsymbol{\omega}_s &= \boldsymbol{u}_t - \widehat{\boldsymbol{u}}\boldsymbol{\omega},
\end{aligned} \tag{3.1}$$

where we have written the acceleration $\boldsymbol{p}_{tt}$ in terms of the local frame velocity $\boldsymbol{q}$ so that

$$\boldsymbol{p}_{tt} = \frac{\partial}{\partial t}(\boldsymbol{R}\boldsymbol{q}) = \boldsymbol{R}_t\boldsymbol{q} + \boldsymbol{R}\boldsymbol{q}_t = \boldsymbol{R}\left(\widehat{\boldsymbol{\omega}}\boldsymbol{q} + \boldsymbol{q}_t\right)$$

and we have assumed that cross-sectional properties ρ and $\boldsymbol{J}$ do not vary with time so that the rate of change of angular momentum is

$$\frac{\partial}{\partial t}\left(\boldsymbol{R}\rho\boldsymbol{J}\boldsymbol{\omega}\right) = \boldsymbol{R}_t\rho\boldsymbol{J}\boldsymbol{\omega} + \boldsymbol{R}\rho\boldsymbol{J}\boldsymbol{\omega}_t = \rho\boldsymbol{R}\left(\widehat{\boldsymbol{\omega}}\boldsymbol{J}\boldsymbol{\omega} + \boldsymbol{J}\boldsymbol{\omega}_t\right).$$

An appropriate material constitutive equation must be chosen to relate $\boldsymbol{m}$ to $\boldsymbol{u}$ and $\boldsymbol{n}$ to $\boldsymbol{v}$. Here we use a linear elastic relation with material damping

$$\begin{aligned}
\boldsymbol{n} &= \boldsymbol{R}[\boldsymbol{K}_{se}(\boldsymbol{v} - \boldsymbol{v}^*) + \boldsymbol{B}_{se}\boldsymbol{v}_t] \\
\boldsymbol{m} &= \boldsymbol{R}[\boldsymbol{K}_{bt}(\boldsymbol{u} - \boldsymbol{u}^*) + \boldsymbol{B}_{bt}\boldsymbol{u}_t].
\end{aligned} \tag{3.2}$$

This equation includes Kelvin-Voigt type viscous damping as described in [59]. Note that these are differential equations, so we cannot explicitly solve for $\boldsymbol{v}$ and $\boldsymbol{u}$ prior to discretizing time.

We may define the distributed force term $\boldsymbol{f}$ to explicitly consider terms for weight and square law drag air resistance so that

$$\boldsymbol{f} := -\boldsymbol{R}\boldsymbol{C}\boldsymbol{q} \odot |\boldsymbol{q}| + \rho A\boldsymbol{g}, \tag{3.3}$$

where the Hadamard product $\odot$ performs element-wise multiplication so that $\boldsymbol{q} \odot |\boldsymbol{q}| = \begin{bmatrix} q_1^2 \operatorname{sgn}(q_1) & q_2^2 \operatorname{sgn}(q_2) & q_3^2 \operatorname{sgn}(q_3) \end{bmatrix}^\top$.

3.1.2 Semi-discretization in Time

For a PDE with one spatial dimension and a time dimension, an ODE can be obtained by discretizing the time partial derivatives. Many implicit finite difference schemes fit the form

$$\boldsymbol{y}_t(t_i) \approx c_0 \boldsymbol{y}(t_i) + \sum_{j=1}^{\infty} c_j \boldsymbol{y}(t_{i-j}) + d_j \boldsymbol{y}_t(t_{i-j}).$$

For example, backward Euler has $c_0 = dt^{-1}$, $c_1 = -dt^{-1}$, $c_j = 0 \ \ \forall j > 1$, and $d_j = 0 \ \ \forall j$. We can abstract the details of the specific scheme by using a single variable to represent all history dependent terms, that is

$$\boldsymbol{y}_t(t_i) \approx c_0 \boldsymbol{y}(t_i) + \overset{\hbar}{\boldsymbol{y}}(t_i). \tag{3.4}$$

Thus the only term of $\boldsymbol{y}_t(t_i)$ corresponding to time t_i is $c_0 \boldsymbol{y}(t_i)$, and $\overset{\hbar}{\boldsymbol{y}}(t_i)$ is defined as the sum of all remaining terms which rely on the past history of $\boldsymbol{y}$. In general we will use this notation for the history-dependent part of a variable's derivative approximation. Lumping all the history-dependent terms together is useful because: 1) the separate terms for current and previous state allow one to decouple the details of the implicit time discretization from the ODEs, and 2) each step in an iterative solver has a new value for the current state $\boldsymbol{y}$, but the history term $\overset{\hbar}{\boldsymbol{y}}$ is common for all steps taken at a single t value.

To implement the method in simulation, we will use the BDF-α method [20], which is $O(\delta t^2)$ accurate. This is described by

$$\begin{aligned} \boldsymbol{y}_t(t_i) &= c_0 \boldsymbol{y}(t_i) + c_1 \boldsymbol{y}(t_{i-1}) + c_2 \boldsymbol{y}(t_{i-2}) + d_1 \boldsymbol{y}_t(t_{i-1}) \\ &:= c_0 \boldsymbol{y}(t_i) + \overset{\hbar}{\boldsymbol{y}}(t_i) \end{aligned}$$

where

$$
\begin{aligned}
c_0 &= (1.5+\alpha)/[\delta t(1+\alpha)] \\
c_1 &= -2/\delta t \\
c_2 &= (0.5+\alpha)/[\delta t(1+\alpha)] \\
d_1 &= \alpha/(1+\alpha).
\end{aligned}
$$

The variable α is an independent parameter in the range $-0.5 \leq \alpha \leq 0$. The trapezoidal method is obtained for $\alpha = -0.5$ and the second-order backward differentiation formula BDF2 is obtained for $\alpha = 0$. Using this approximation, the solution to the ODE in s at t_i is dependent on the solutions to previous ODEs at times t_{i-1} and t_{i-2}.

With the time discretized, the strains $\boldsymbol{v}$ and $\boldsymbol{u}$ can be solved as a function of the internal forces $\boldsymbol{n}$ and $\boldsymbol{m}$. Recall from (3.2) that the internal force is described by

$$
\boldsymbol{n} = \boldsymbol{R}\left[\boldsymbol{K}_{se}(\boldsymbol{v}-\boldsymbol{v}^*) + \boldsymbol{B}_{se}\boldsymbol{v}_t\right].
$$

We express the internal force in the local frame and apply the time discretization $\boldsymbol{v}_t \approx c_0\boldsymbol{v} + \mathring{\boldsymbol{v}}$ so that

$$
\boldsymbol{n}^b = \boldsymbol{K}_{se}(\boldsymbol{v}-\boldsymbol{v}^*) + \boldsymbol{B}_{se}(c_0\boldsymbol{v} + \mathring{\boldsymbol{v}}).
$$

We solve for $\boldsymbol{v}$ to find

$$
\boldsymbol{v} = (\boldsymbol{K}_{se} + c_0\boldsymbol{B}_{se})^{-1}\left(\boldsymbol{n}^b + \boldsymbol{K}_{se}\boldsymbol{v}^* - \boldsymbol{B}_{se}\mathring{\boldsymbol{v}}\right),
$$

and by an identical process we obtain $\boldsymbol{u}$ as

$$
\boldsymbol{u} = (\boldsymbol{K}_{bt} + c_0\boldsymbol{B}_{bt})^{-1}\left(\boldsymbol{m}^b + \boldsymbol{K}_{bt}\boldsymbol{u}^* - \boldsymbol{B}_{bt}\mathring{\boldsymbol{u}}\right).
$$

Finally, combining this result with the Cosserat rod PDE (3.1), the definition of distributed force (3.3), and the time discretization (3.4), we have arrived at an ODE system. The final

dynamic ODE system is:

$$
\begin{aligned}
\boldsymbol{v} &= (\boldsymbol{K}_{se} + c_0\boldsymbol{B}_{se})^{-1}\left(\boldsymbol{R}^\top\boldsymbol{n} + \boldsymbol{K}_{se}\boldsymbol{v}^* - \boldsymbol{B}_{se}\overset{\natural}{\boldsymbol{v}}\right)\\
\boldsymbol{u} &= (\boldsymbol{K}_{bt} + c_0\boldsymbol{B}_{bt})^{-1}\left(\boldsymbol{R}^\top\boldsymbol{m} + \boldsymbol{K}_{bt}\boldsymbol{u}^* - \boldsymbol{B}_{bt}\overset{\natural}{\boldsymbol{u}}\right)\\
\boldsymbol{v}_t &= c_0\boldsymbol{v} + \overset{\natural}{\boldsymbol{v}}\\
\boldsymbol{u}_t &= c_0\boldsymbol{u} + \overset{\natural}{\boldsymbol{u}}\\
\boldsymbol{q}_t &= c_0\boldsymbol{q} + \overset{\natural}{\boldsymbol{q}}\\
\boldsymbol{\omega}_t &= c_0\boldsymbol{\omega} + \overset{\natural}{\boldsymbol{\omega}}\\
\boldsymbol{f} &= -\boldsymbol{R}\boldsymbol{C}\boldsymbol{q}\odot|\boldsymbol{q}| + \rho A\boldsymbol{g}
\end{aligned}
\tag{3.5}
$$

$$
\begin{aligned}
\boldsymbol{p}_s &= \boldsymbol{R}\boldsymbol{v}\\
\boldsymbol{R}_s &= \boldsymbol{R}\widehat{\boldsymbol{u}}\\
\boldsymbol{n}_s &= \rho A\boldsymbol{R}\left(\widehat{\boldsymbol{\omega}}\boldsymbol{q} + \boldsymbol{q}_t\right) - \boldsymbol{f}\\
\boldsymbol{m}_s &= \rho\boldsymbol{R}\left(\widehat{\boldsymbol{\omega}}\boldsymbol{J}\boldsymbol{\omega} + \boldsymbol{J}\boldsymbol{\omega}_t\right) - \widehat{\boldsymbol{p}}_s\boldsymbol{n} - \boldsymbol{l}\\
\boldsymbol{q}_s &= \boldsymbol{v}_t - \widehat{\boldsymbol{u}}\boldsymbol{q} + \widehat{\boldsymbol{\omega}}\boldsymbol{v}\\
\boldsymbol{\omega}_s &= \boldsymbol{u}_t - \widehat{\boldsymbol{u}}\boldsymbol{\omega}
\end{aligned}
$$

Implicit Midpoint Time Discretization

Although the above approach is amenable to many discretizations, the implicit midpoint method does not fit the form of (3.4). This method has the benefit that it is energy conservative and stable, whereas the trapezoidal method borders on unstable. For a problem with $\boldsymbol{y}_t = \boldsymbol{g}(t, \boldsymbol{y})$, the implicit midpoint method is given by

$$
\boldsymbol{y}(t_i) \approx \boldsymbol{y}(t_{i-1}) + \delta t\ \boldsymbol{g}\left(\frac{t_i + t_{i-1}}{2}, \frac{\boldsymbol{y}(t_i) + \boldsymbol{y}(t_{i-1})}{2}\right).
$$

This is an implicit approach, but for a PDE system with $\boldsymbol{y}_s = \boldsymbol{f}(t, \boldsymbol{y}, \boldsymbol{y}_t)$ and $\boldsymbol{y}_t = \boldsymbol{g}(t, \boldsymbol{y}, \boldsymbol{y}_s)$ it can be solved explicitly for $\boldsymbol{y}_s$. The PDEs may also have an explicit dependence on s, but

it is omitted for brevity. The implicit midpoint method is

$$\frac{\boldsymbol{y}(t_i) - \boldsymbol{y}(t_{i-1})}{\delta t} = \boldsymbol{g}(\frac{t_i + t_{i-1}}{2}, \frac{\boldsymbol{y}(t_i) + \boldsymbol{y}(t_{i-1})}{2}, \frac{\boldsymbol{y}_s(t_i) + \boldsymbol{y}_s(t_{i-1})}{2}),$$

which implies that the equation for $\boldsymbol{f}$ must also hold:

$$\frac{\boldsymbol{y}_s(t_i) + \boldsymbol{y}_s(t_{i-1})}{2} = \boldsymbol{f}(\frac{t_i + t_{i-1}}{2}, \frac{\boldsymbol{y}(t_i) + \boldsymbol{y}(t_{i-1})}{2}, \frac{\boldsymbol{y}(t_i) - \boldsymbol{y}(t_{i-1})}{\delta t}).$$

This can be solved for

$$\boldsymbol{y}_s(t_i) = -\boldsymbol{y}_s(t_{i-1}) + 2\boldsymbol{f}(\frac{t_i + t_{i-1}}{2}, \frac{\boldsymbol{y}(t_i) + \boldsymbol{y}(t_{i-1})}{2}, \frac{\boldsymbol{y}(t_i) - \boldsymbol{y}(t_{i-1})}{\delta t}). \tag{3.6}$$

Now the PDE is semi-discretized as $\boldsymbol{y}_s(t_i)$ defines an ODE which depends on the solution to an ODE at the previous time step.

This approach becomes more complicated when there are some variables having a partial derivative in time but not in arc length, which is the case for $\boldsymbol{v}$ and $\boldsymbol{u}$. Let there be functions $\boldsymbol{y}_s = \boldsymbol{f}(t, \boldsymbol{y}, \boldsymbol{y}_t, \boldsymbol{z}, \boldsymbol{z}_t)$ and $\boldsymbol{y}_t = \boldsymbol{g}(t, \boldsymbol{y}, \boldsymbol{y}_s, \boldsymbol{z}, \boldsymbol{z}_t)$. Then

$$\begin{aligned}\boldsymbol{y}_s(t_i) &= -\boldsymbol{y}_s(t_{i-1}) \\ &+ 2\boldsymbol{f}(\frac{t_i + t_{i-1}}{2}, \frac{\boldsymbol{y}(t_i) + \boldsymbol{y}(t_{i-1})}{2}, \frac{\boldsymbol{y}(t_i) - \boldsymbol{y}(t_{i-1})}{\delta t}, \frac{\boldsymbol{z}(t_i) + \boldsymbol{z}(t_{i-1})}{2}, \frac{\boldsymbol{z}(t_i) - \boldsymbol{z}(t_{i-1})}{\delta t}).\end{aligned} \tag{3.7}$$

Once again, the discretized form of the constitutive equation with material damping (3.2) can be solved explicitly. The discretized equation is

$$\bar{\boldsymbol{n}} = \bar{\boldsymbol{R}}\left[\boldsymbol{K}_{se}(\frac{\boldsymbol{v}(t_i) + \boldsymbol{v}(t_{i-1})}{2} - \boldsymbol{v}^*) + \boldsymbol{B}_{se}\frac{\boldsymbol{v}(t_i) - \boldsymbol{v}(t_{i-1})}{\delta t}\right]$$

where we have defined

$$\bar{\boldsymbol{n}} := \frac{\boldsymbol{n}(t_i) + \boldsymbol{n}(t_{i-1})}{2}, \qquad \bar{\boldsymbol{R}} := \frac{\boldsymbol{R}(t_i) + \boldsymbol{R}(t_{i-1})}{2}.$$

This is solved as

$$\boldsymbol{v}(t_i) = (\frac{\boldsymbol{K}_{se}}{2} + \frac{\boldsymbol{B}_{se}}{\delta t})^{-1} \left[(-\frac{\boldsymbol{K}_{se}}{2} + \frac{\boldsymbol{B}_{se}}{\delta t})\boldsymbol{v}(t_{i-1}) + \bar{\boldsymbol{R}}^{\top}\bar{\boldsymbol{n}} + \boldsymbol{K}_{se}\boldsymbol{v}^{*} \right], \tag{3.8}$$

and likewise

$$\boldsymbol{u}(t_i) = (\frac{\boldsymbol{K}_{bt}}{2} + \frac{\boldsymbol{B}_{bt}}{\delta t})^{-1} \left[(-\frac{\boldsymbol{K}_{bt}}{2} + \frac{\boldsymbol{B}_{bt}}{\delta t})\boldsymbol{u}(t_{i-1}) + \bar{\boldsymbol{R}}^{\top}\bar{\boldsymbol{m}} + \boldsymbol{K}_{bt}\boldsymbol{u}^{*} \right].$$

With these terms solved, the ODE $\boldsymbol{y}_s$ may be evaluated to solve a BVP in arc length.

3.1.3 Spatial Discretization

Shooting Method

With the problem semi-discretized in time, we must now discretize the arc length dimension. We will once again use the shooting method as one approach. Multistage ODE integrators require some interpolation to estimate history dependent terms since they lie between the grid points of previous ODE solutions. For example, the RK4 scheme requires a midpoint value for $\overset{\text{h}}{\boldsymbol{y}}(s_{j+0.5})$. We use linear interpolation.

Although arbitrarily large time steps are possible by using an asymptotically-stable finite difference scheme, poor conditioning is encountered at small time steps. The rod state is continuous almost everywhere in both time and space so that

$$\lim_{\delta t \to 0} \boldsymbol{y}(t + \delta t) - \boldsymbol{y}(t) = \boldsymbol{0}.$$

Thus, performing the calculation $\boldsymbol{y}(t+\delta t)-\boldsymbol{y}(t)$ at small time steps results in a "catastrophic cancellation," where subtracting two nearly equal numbers causes a loss of significant digits [34]. This is an inherent property of the implicit time-discretization, and this effect seems to be exacerbated by the numerical integration of shooting. While the breakdown of the time-discretized scheme is inevitable with decreasing time steps, the simulations of a small steel rod described later are able to run with a time step as small as 2-5ms using just a simple shooting method. There are also modifications of the shooting method that can

improve convergence, such as the modified simple shooting method [46]. Thus shooting is often effective, and has the benefit of good computational efficiency.

Finite Difference System

To study particularly stiff dynamics such as concentric tube snapping, it may be necessary to use a small time step beyond what is feasible with shooting. In this case we can solve the ODE using a finite difference system.

Consider a general state variable $\bar{\boldsymbol{y}}(s) \in \mathbb{R}^M$ with an ODE $\bar{\boldsymbol{y}}_s = f(s, \bar{\boldsymbol{y}})$. Let the domain be discretized into N grid points s_1 through s_N, and let the approximate state at these points be denoted by $\boldsymbol{y}_i := \boldsymbol{y}(s_i)$. Using the midpoint rule one can write a finite difference equation

$$\frac{\boldsymbol{y}_{i+1} - \boldsymbol{y}_i}{s_{i+1} - s_i} \approx f\left(\frac{s_i + s_{i+1}}{2}, \frac{\boldsymbol{y}_i + \boldsymbol{y}_{i+1}}{2}\right)$$

Let the states over the whole grid be joined in a state vector $\boldsymbol{Y} := \begin{bmatrix} \boldsymbol{y}_1^\top & \boldsymbol{y}_2^\top & \dots & \boldsymbol{y}_N^\top \end{bmatrix}^\top$, then a vector containing the residual errors of the differential equations may be expressed as

$$\boldsymbol{E}(\boldsymbol{Y}) = \begin{bmatrix} \frac{\boldsymbol{y}_2 - \boldsymbol{y}_1}{s_2 - s_1} - f\left(\frac{s_1+s_2}{2}, \frac{\boldsymbol{y}_1+\boldsymbol{y}_2}{2}\right) \\ \frac{\boldsymbol{y}_3 - \boldsymbol{y}_2}{s_3 - s_2} - f\left(\frac{s_2+s_3}{2}, \frac{\boldsymbol{y}_2+\boldsymbol{y}_3}{2}\right) \\ \vdots \\ \frac{\boldsymbol{y}_N - \boldsymbol{y}_{N-1}}{s_N - s_{N-1}} - f\left(\frac{s_{N-1}+s_N}{2}, \frac{\boldsymbol{y}_{N-1}+\boldsymbol{y}_N}{2}\right) \end{bmatrix}. \tag{3.9}$$

The boundary conditions are strongly satisfied so that some elements of $\boldsymbol{y}$ are specified. The unknown elements are constructed from a guess vector $\boldsymbol{G}$ so that $\boldsymbol{Y} = \boldsymbol{Y}(\boldsymbol{G})$. For a square system $\boldsymbol{E}$ and $\boldsymbol{G}$ both have $(N-1) \times M$ elements where M is the number of elements in the state vector $\bar{\boldsymbol{y}}(s)$, and $\boldsymbol{G}$ may be solved so that $\boldsymbol{E} = \boldsymbol{0}$ by iterative methods.

This approach discretizes *one* system of ODEs. Several problems we will consider have coupled PDE systems, such as a rod with a discontinuous internal force due to an applied impulse, or a concentric tube robot. Extending the above approach to account for coupled systems is tedious, but relatively straightforward.

3.1.4 Examples

A cantilevered steel rod has length $L = 0.4$m, radius $r = 1.2$mm, and negligible damping coefficients. The world frame is arbitrarily assigned so that $\boldsymbol{p}(t,0) = \boldsymbol{0}$ and $\boldsymbol{R}(t,0) = \boldsymbol{I}$. The nature of the cantilever attachment requires $\boldsymbol{q}(t,0) = \boldsymbol{\omega}(t,0) = \boldsymbol{0}$. The scenario starts at static equilibrium with a 0.2kg mass tied to the tip so that the distal boundary conditions are

$$\boldsymbol{n}(0,L) = 0.2kg * \boldsymbol{g}, \quad \boldsymbol{m}(0,L) = \boldsymbol{0},$$

where gravity acts in the negative x-direction so that $\boldsymbol{g} = -9.81\boldsymbol{e}_1 \text{m/s}^2$. The string holding the mass is suddenly cut so that there are no forces at the free end, that is

$$\boldsymbol{n}(t,L) = \boldsymbol{m}(t,L) = \boldsymbol{0}$$

for all $t > 0$.

This could be treated as a planar problem, but we are developing a method for the general spatial case, so we will guess the entire proximal wrench and have an error for the entire distal wrench. Thus the shooting method is defined by

$$\boldsymbol{G} = \begin{bmatrix} \boldsymbol{n}^\top(0) & \boldsymbol{m}^\top(0) \end{bmatrix}^\top, \quad \boldsymbol{E}(\boldsymbol{G}) = \begin{bmatrix} \boldsymbol{n}^\top(L) & \boldsymbol{m}^\top(L) \end{bmatrix}^\top.$$

A MATLAB code implementing this solution with the BDF-α time discretization and shooting arc length discretization is included in Appendix D. The first few frames of the simulation are shown in Figure 3.2.

This problem is also solved by discretizing time with the implicit midpoint method and shooting in space in Appendix E, and by using the BDF-α method in time with a finite difference system in space in Appendix F. The MATLAB example of the finite difference system is overly slow since it builds a dense Jacobian for a sparse problem, but it nonetheless illustrates the nature of the objective function. A separate MATLAB example of rod dynamics with orientation represented as a quaternion is shown in [109].

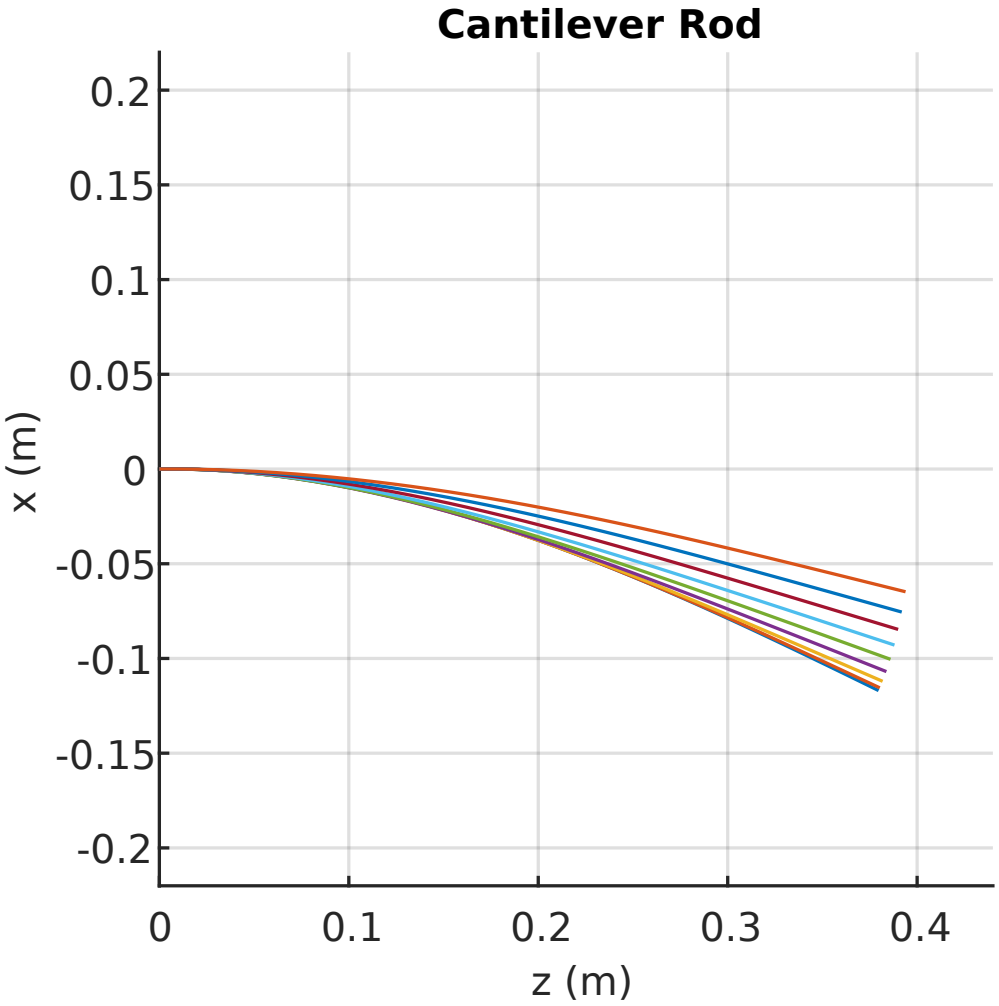


Figure 3.2: In an example problem, a cantilever rod with an applied tip force is released to spring upward. The different lines correspond to timesteps of the simulation. The simulation is implemented in MATLAB as shown in Appendix D using the BDF-α method for the time semi-discretization, and a shooting method to solve the spatial BVP.

3.1.5 Experimental Validation of a Cantilevered Case

The rod model was experimentally validated by comparing simulation results to high-speed footage of a cantilevered rod clamped to a table as described in [112, 109]. The simulations implement a full 3-dimensional model, but the experimental data was taken from planar cases to simplify the process of reconstructing the scene and evaluating the results. The model is validated using both the Cosserat equations and Kirchhoff equations (see Chapter 2.1.5 for a description). The rod was spring steel with a 1.42mm diameter. There were two scenarios. First, a 20g weight was hung by a string at the tip of a rod with a cantilevered length of 0.408m, and after equilibrium was reached, the string was cut. Second, the cantilevered length was increased to 0.517m to obtain larger vibrations, and the rod was hit with a rigid object near its base to excite high-frequency vibration modes. The BDF-α coefficient was $\alpha = -0.4$ for all simulations, which is close to the trapezoidal method and thus exhibits very little numerical damping.

The camera was placed about three meters from the rod with the viewing plane parallel to the rod's plane of motion. The camera recorded a frame every millisecond. The rod was darker than the background so that the experimental rod position could be easily extracted by comparing pixel brightness values, as shown in Figure 3.3.

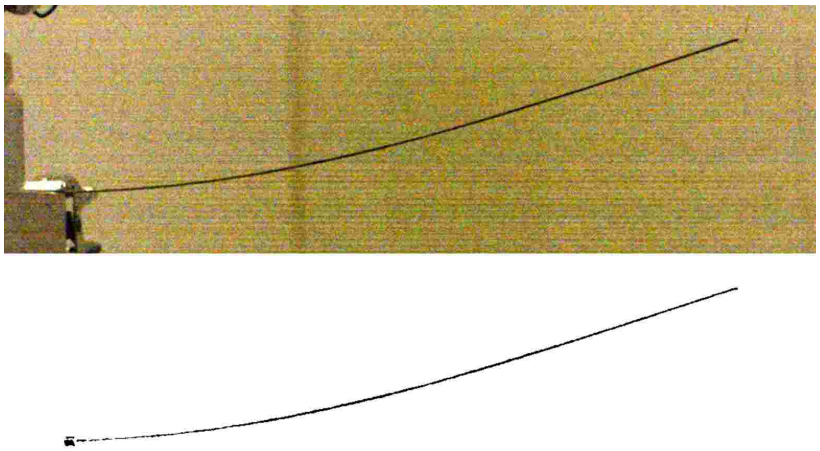


Figure 3.3: The experimental rod shape was quantified by obtaining binary data based on the aggregate brightness value for a 3x3 neighborhood around a pixel. The rightmost pixel is taken as the tip, specifically the top rightmost pixel when multiple rightmost pixels exist.

Weight Release

The weight release trial was used to calibrate the rod parameters EI, ρ, and C. This calibration is implemented in MATLAB using "fsolve" to minimize error between experimental data and model prediction. The weight release response is very nearly a decaying sine wave, as shown in Figure 3.4. The calibration objective function was evaluated by running the simulation for a set of parameters and evaluating the characteristics of the simulated response versus the experimental data. The magnitude of the first peak, magnitude of the first valley, magnitude of the final peak, and frequency are compared and combined to form the objective function residual. MATLAB's "findpeaks" command can easily detect peaks in the smooth simulation data. The experimental data has some noise, but since the experimental response only needs to be analyzed once, this was done manually. The calibrated values are shown in Table 3.1. For steel, ρ is typically around 7800 kg/m^3. With an assumed Young's modulus of 200GPa, the 1.42mm diameter rod would have a bending stiffness EI of 0.03992 Nm^2. Thus, the calibrated values are within reason.

Impulse Near Base

After calibration of the model parameters using the weight release dataset, we evaluated the model prediction versus data taken from the impulse response experiment. The impulse

Table 3.1: Calibrated Parameters

	Cosserat		**Kirchhoff**	
Parameter	**Euler, N=400**	**RK4, N=100**	**Euler, N=400**	**RK4, N=100**
EI (Nm^2)	0.0380	0.0380	0.0380	0.0380
ρ (kg/m^3)	7602	7602	7602	7603
C (g/m^2)	2.09	2.12	2.08	2.12

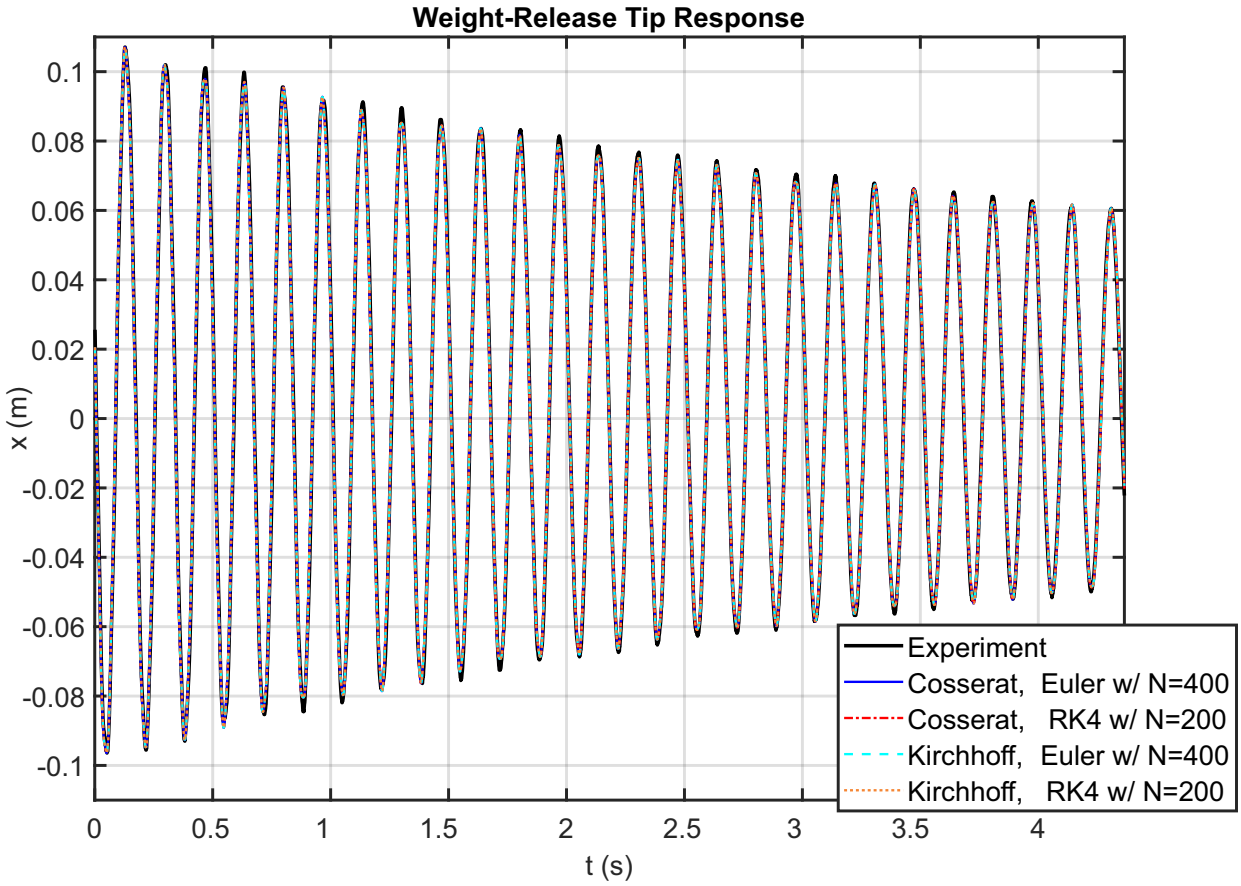


Figure 3.4: A weight was attached to the free end of a cantilevered rod by a string. After the weighted rod reached equilibrium, the string was cut. This scenario was simulated, and the simulation parameters were calibrated so that the simulation response visibly matches the experimental response. These calibrated values were used while validating the other impulse experiment.

point force was modeled as a hat function in time with

$$F(t) = \begin{cases} M\frac{t}{0.5d}, & t < 0.5d \\ M(2 - \frac{t}{0.5d}), & 0.5d \leq t \leq d \\ 0, & t > d. \end{cases}$$

Appropriate values for the impulse's peak magnitude and duration were found: $M = 5$N and $d = 0.016$s. The impulse point force is included in the simulation by performing piecewise integration of the ODEs in space and applying the point force at the transition. A still frame of the experimental footage is shown in Figure 3.5. The impulse response is shown in Figure 3.6. The simulation responses are similar for the Cosserat and Kirchhoff models, so we conclude that the shear and extension strains are indeed negligible for the experimental rod.

Real-Time Performance

To evaluate computational speed, we ran many simulations of both the weight release and impulse response scenarios using increasing values of δt (logarithmically spaced). Results are shown in the log-log plots of Figure 3.7. The real-time performance ratio is the amount of

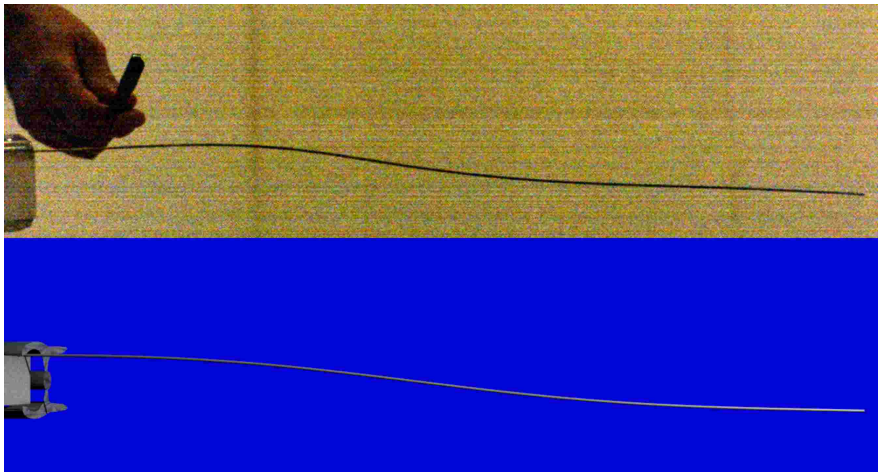


Figure 3.5: An impulse point force is applied near the base of a cantilevered rod, resulting in a variety of vibration modes. The model simulation is visualized with Blender.

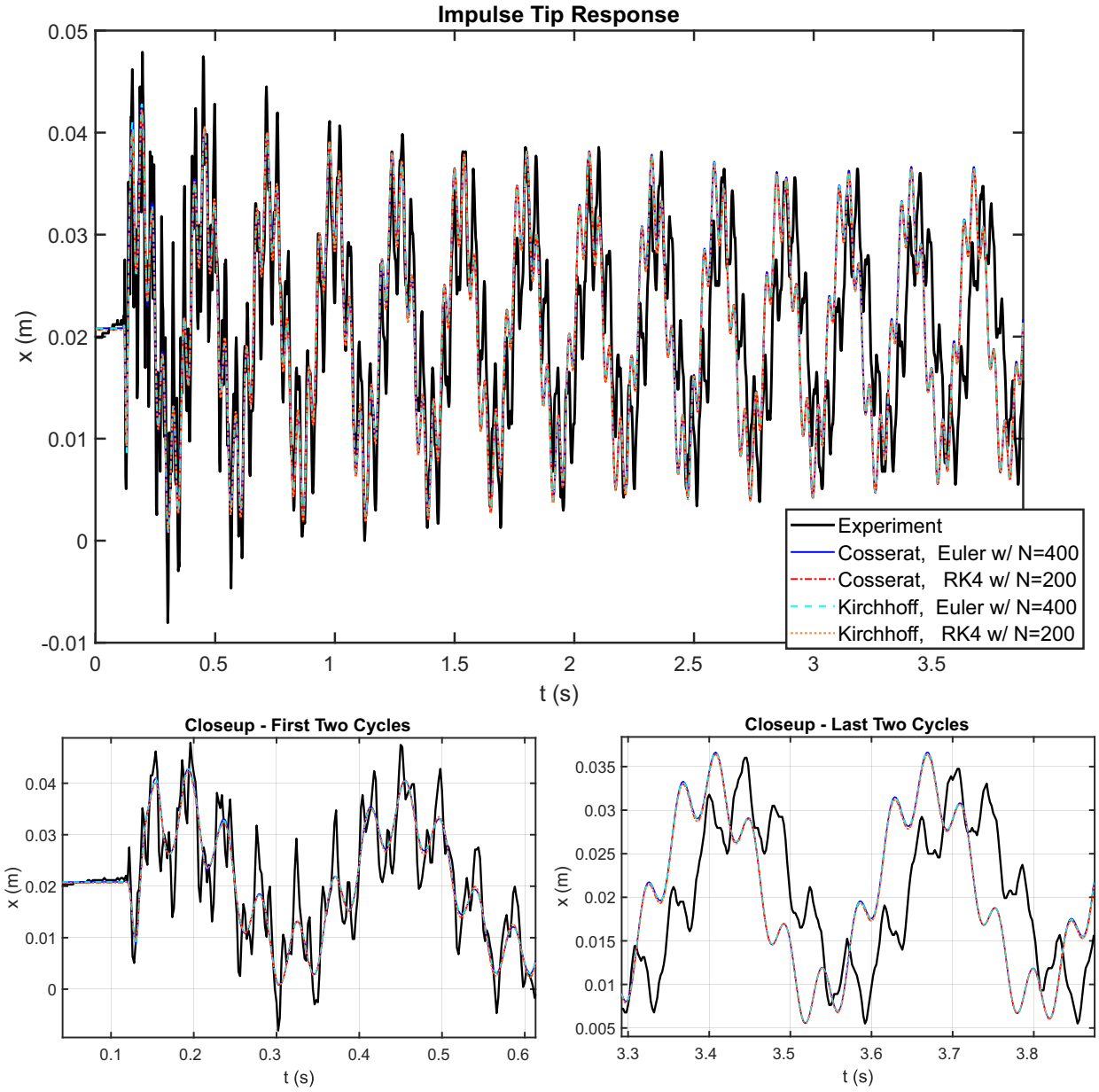


Figure 3.6: An impulse was applied near the base of a cantilevered rod. The experimental impulse response is compared to model predictions with both Cosserat and Kirchhoff rod theories. The simulated behavior matches experimental behavior closely.

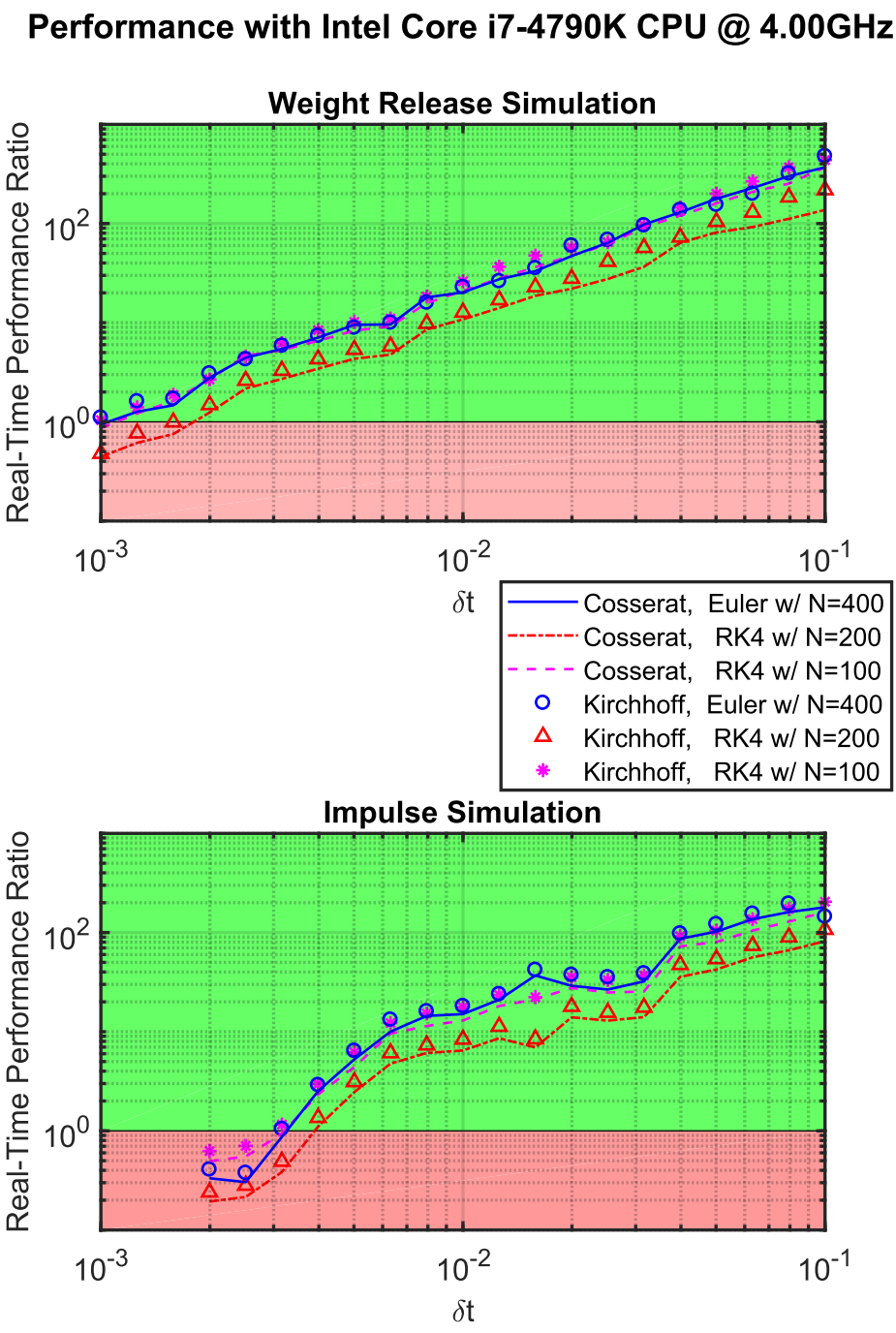


Figure 3.7: This plot shows the real-time performance ratio versus the time step on a log-log scale for various simulation datasets. The real-time performance ratio is the amount of time simulated divided by the wall-clock time spent running the simulation. A ratio greater than or equal to one indicates soft real-time performance, as shown by the green regions. The weight release scenario requires significantly less effort to solve than the impulse scenario. For the difficult impulse scenario, the simulation can run in soft real-time with $\delta t = 4$ms. The smallest time step for the impulse simulations is 2ms because of convergence issues with small time steps.

time simulated divided by the wall-clock time spent running the simulation, and the green regions indicate real-time performance. The plot confirms that most of the simulations ran in real-time. At higher time steps, the dependence of the run time on the time step is nearly linear. The run time also appears linear in the number of spatial steps, again confirming $O(N)$ computation time as we showed in the previous section.

Not surprisingly, the impulse response case requires higher computational times due to the increased presence of faster dynamic modes that require more solver iterations per time step. A time step of 2 milliseconds captured the high-frequency dynamics very accurately, but this simulation required more computation time than it simulated, and the speed is further reduced when the solver begins to encounter numerical ill-conditioning at smaller time steps.

3.2 Continuum Stewart-Gough Robot

Parallel continuum robot dynamics can be significant, especially for larger scale applications with potential human interaction. A particular feature that we would like our model to capture is the dynamic transition from one stable static state to another when the robot configuration becomes unstable due to actuation or external loading. In this section, we outline the equations necessary to solve this problem and demonstrate real-time simulation of a six-DOF parallel Stewart-Gough platform undergoing a dynamic stability transition.

The geometric constraints on the CSG are the same as those given in (2.9) and (2.10), except the states are now parameterized by time so that

$$\boldsymbol{p}_e(t) + \boldsymbol{R}_e(t)\boldsymbol{r}_i = \boldsymbol{p}_i(t, L_i) \text{ for i} = 1...6, \tag{3.10}$$

and instead of shaft collars connecting the rods to the end effector, we consider a fully rigid connection so that orientation is constrained by

$$\boldsymbol{R}_e(t) = \boldsymbol{R}_i(t, L_i) \text{ for i} = 1...6. \tag{3.11}$$

The rigid body equilibrium constraints originally given by (2.11) are updated to include inertial terms. The sum of forces has contributions from the rod attachments, gravity, and external loads as described by

$$\boldsymbol{F}_e(t) + m_e\boldsymbol{g} - \sum_{i=1}^{6} \boldsymbol{n}_i(t, L_i) = m_e\boldsymbol{a}_e(t), \tag{3.12}$$

where m_e is the end-effector mass, $\boldsymbol{a}_e$ is the end-effector acceleration in the global frame, and $\boldsymbol{F}_e$ is an external force acting on the end-effector center of mass. The moment balance equation, with end-effector angular velocity $\boldsymbol{\omega}_e$ defined in the global frame, is

$$\begin{aligned} &\boldsymbol{M}_e(t) - \sum_{i=1}^{6} \boldsymbol{m}_i(t, L_i) + [\boldsymbol{R}_e(t)\boldsymbol{r}_i] \times \boldsymbol{n}_i(t, L_i) = \\ &\boldsymbol{R}_e(t)\boldsymbol{J}_e\boldsymbol{R}_e^{\top}(t)\boldsymbol{\omega}_{te}(t) + \widehat{\boldsymbol{\omega}}_e(t)\boldsymbol{R}_e(t)\boldsymbol{J}_e\boldsymbol{R}_e^{\top}(t)\boldsymbol{\omega}_e(t). \end{aligned} \tag{3.13}$$

The constraint equations (3.10), (3.11), (3.12), and (3.13) define a boundary value problem subject to the integrated rod distal conditions.

Formulating a scheme to solve the forward dynamics BVP is an interesting subproblem, but in this case we arbitrarily rely on the shooting method with a guess

$$\boldsymbol{G} = [\boldsymbol{n}_1^{\top}(t,0) \quad \boldsymbol{m}_1^{\top}(t,0) \quad \boldsymbol{n}_2^{\top}(t,0) \quad \boldsymbol{m}_2^{\top}(t,0) \quad \ldots \quad \boldsymbol{n}_6^{\top}(t,0) \quad \boldsymbol{m}_6^{\top}(t,0) \quad \boldsymbol{p}_e^{\top} \quad \boldsymbol{k}^{\top}]^{\top},$$

where $\boldsymbol{k}$ is a 3x1 vector used to generate the end-effector rotation by

$$\boldsymbol{R}_e(\boldsymbol{k}) = \begin{cases} \boldsymbol{I} + \sin\|\boldsymbol{k}\| \frac{\widehat{\boldsymbol{k}}}{\|\boldsymbol{k}\|} + (1 - \cos\|\boldsymbol{k}\|) \frac{\widehat{\boldsymbol{k}}^2}{\|\boldsymbol{k}\|^2}, & \|\boldsymbol{k}\| > 0 \\ \boldsymbol{I}, & \|\boldsymbol{k}\| = 0 \end{cases}.$$

The dynamic terms for the end effector are found by

$$\begin{aligned}
\boldsymbol{v}_e &= c_0 \boldsymbol{p}_e + \overset{\natural}{\boldsymbol{p}}_e \\
\boldsymbol{a}_e &= c_0 \boldsymbol{v}_e + \overset{\natural}{\boldsymbol{v}}_e \\
\boldsymbol{R}_{e,t} &= c_0 \boldsymbol{R}_e + \overset{\natural}{\boldsymbol{R}}_e \\
\boldsymbol{\omega}_e &= \left(\boldsymbol{R}_e^\top \boldsymbol{R}_{e,t}\right)^\vee \\
\boldsymbol{\omega}_{e,t} &= c_0 \boldsymbol{\omega}_e + \overset{\natural}{\boldsymbol{\omega}}_e,
\end{aligned}$$

then errors may be defined by

$$\begin{aligned}
\boldsymbol{E}^F &= \boldsymbol{F}_e + m_e(\boldsymbol{g} - \boldsymbol{a}_e) - \sum_{i=1}^{6} \boldsymbol{n}_i \\
\boldsymbol{E}^M &= \boldsymbol{M}_e - \boldsymbol{R}_e \left(\boldsymbol{J}\boldsymbol{\omega}_{e,t} + \widehat{\boldsymbol{\omega}}_e \boldsymbol{J}\boldsymbol{\omega}_e\right) - \sum_{i=1}^{6} (\boldsymbol{m}_i + \boldsymbol{R}_e \boldsymbol{r}_i \times \boldsymbol{n}_i) \\
\boldsymbol{E}_i^p &= \boldsymbol{p}_e + \boldsymbol{R}_e \boldsymbol{r}_i - \boldsymbol{p}_i \\
\boldsymbol{E}_i^R &= \left(\boldsymbol{R}_e^\top \boldsymbol{R}_i - \boldsymbol{R}_e \boldsymbol{R}_i^\top\right)^\vee
\end{aligned}$$

so that the total error is

$$\boldsymbol{E}(\boldsymbol{G}) = [(\boldsymbol{E}_1^p)^\top \quad (\boldsymbol{E}_1^R)^\top \quad (\boldsymbol{E}_2^p)^\top \quad (\boldsymbol{E}_2^R)^\top \quad \ldots \quad (\boldsymbol{E}_6^p)^\top \quad (\boldsymbol{E}_6^R)^\top \quad (\boldsymbol{E}^F)^\top \quad (\boldsymbol{E}^M)^\top]^\top.$$

The guessed set $\boldsymbol{G}$ and residual $\boldsymbol{E}(\boldsymbol{G})$ form a square 42x42 system of nonlinear equations. There is some interesting flexibility in forming the sets $\boldsymbol{G}$ and $\boldsymbol{E}$, but we have found this particular choice to be concise and simple to understand.

There is a potential for instability of parallel continuum robots that we have observed experimentally, and we have established a method based on optimal control theory to assess the stability of any solution to the static model equations as described in Chapter 4. However, this method does not provide information about how the robot dynamically transitions to a new stable equilibrium elsewhere in the workspace. We demonstrate the ability of our dynamic modeling framework to capture this behavior by simulating the forward dynamics of a robot which is actuated to a statically unstable configuration. The robot is shown in

Figure 3.8 and the dynamic trajectory of the end effector is plotted in Figure 3.9. In this scenario, slow changes in the actuator positions translate the end effector in a straight line, until eventually there is a bifurcation leading to dynamic end-effector behavior.

This simulation had a real-time ratio of about 7-8 using three threads working in parallel for the integration of (3.5). The rods had identical parameters of $E = 207\text{GPa}$, $r = 1\text{mm}$, and $\rho = 8000\text{kg/m}^3$. The Stewart-Gough leg-spacing pattern used a major angle of 100° and a radius of 87mm. The end effector was modeled as a short acrylic ($\rho = 1180\text{kg/m}^3$) cylinder with 3mm depth and radius of 91mm, which results in a mass of 92.1g and a mass moment of inertia $\boldsymbol{J} = \text{diag}(1.91, 1.91, 3.81) \times 10^{-4}$ kg-m^2. The damping parameters were all zero. The discretization parameters were $\delta t = 1/120\text{s}$, $\alpha = -0.2$, and 200 points per rod with Euler's method. The shooting method solver was the Trust-Region-Dogleg scheme. We handle the insertion of rods through the baseplate by discretizing the portion of a rod above the baseplate into a constant number of points. We assume quasi-static insertion speed, that is $\partial ds/\partial t \approx 0$, and the initial velocity $q_{i,z}(t, 0)$ is the same as the actuator velocity (which assumes negligible extension below the base).

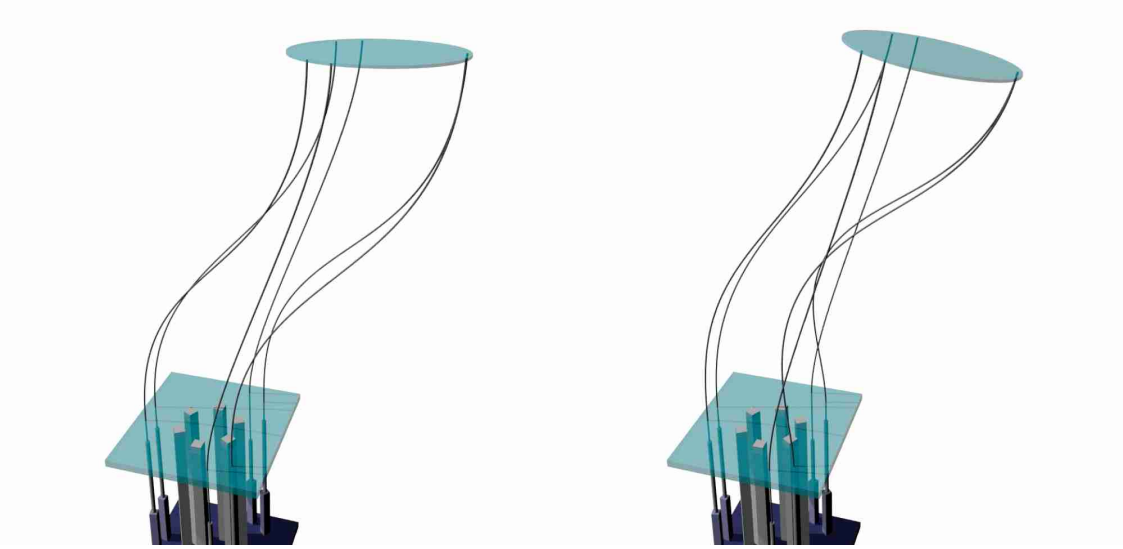


Figure 3.8: As the continuum Stewart-Gough robot is translated along a path which satisfies the equilibrium equations, the robot encounters a bifurcation. A moment prior to instability is shown on the left. On the right, the end-effector bends to an angle and begins to sway dynamically.

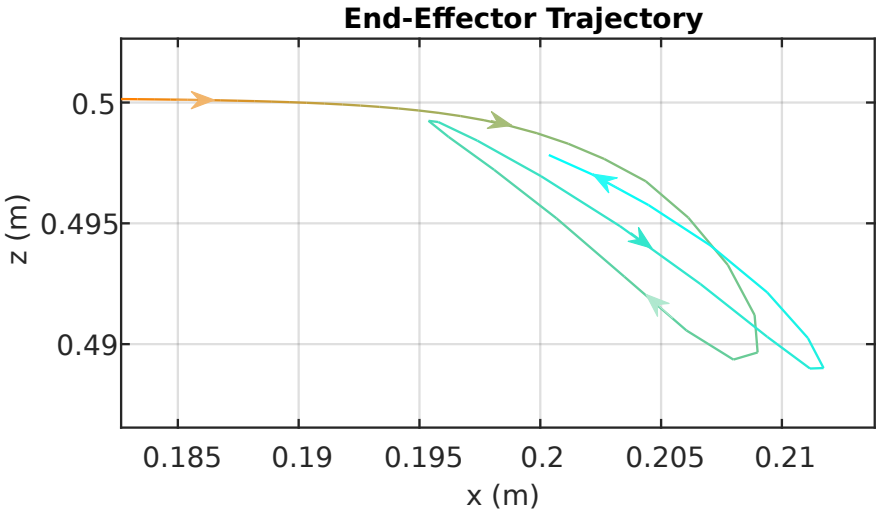


Figure 3.9: As the CSG is actuated past a bifurfaction, the end-effector sways back and forth dynamically as illustrated by the trajectory plot.

3.3 Tendon-driven Robot

A tendon-driven robot is shown in Figure 3.10. The statics and dynamics of these robots and steerable catheters were derived from the Cosserat rod framework in [97]. This prior work simulated the robot dynamics with an explicit Lax-Wendroff finite-difference scheme which was severely limited by the Courant-Friedrichs-Lewy numerical stability condition, even for a model with less than ten spatial segments. In this section, we review and extend the model of [97], adding internal damping and drag terms and demonstrating how to use our implicit approach to achieve efficient real-time simulation of the dynamics.

3.3.1 Differential Equations

We assume the robot consists of an elastic backbone member modeled as a Cosserat rod with continuous channels for actuation cables which apply shape-dependent forces and moments to the backbone when tensioned. This basic design describes many tools and is a reasonable continuous approximation in cases with discrete routing holes created by spacer disks. There are n cables, and each cable experiences a tension τ_i and is offset from the cross-section center of mass by a vector $\boldsymbol{r}_i(s)$ in the local cross-sectional plane, such that each tendon's position

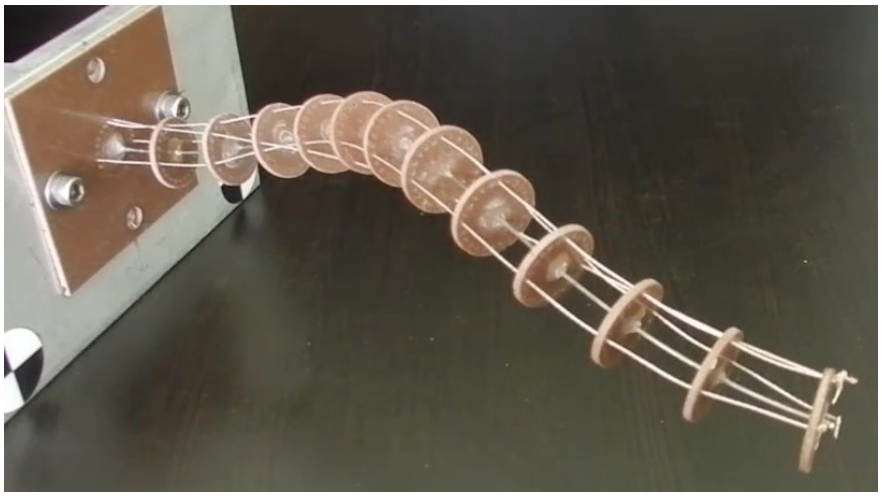


Figure 3.10: A tendon-driven continuum robot with a helical tendon is actuated to a variable-curvature shape with significant inertial dynamics.

in the global frame is

$$\boldsymbol{p}_i = \boldsymbol{p} + \boldsymbol{R}\boldsymbol{r}_i$$

The cables cause distributed forces and moments on the rod which are derived under the assumption of negligible tendon friction and inertia as

$$\begin{aligned}
\boldsymbol{f}_c &= -\sum_{i=1}^{n} \tau_i \frac{\widehat{\boldsymbol{p}}_{si}^2}{\|\boldsymbol{p}_{si}\|} \boldsymbol{p}_{ssi} \\
\boldsymbol{l}_c &= -\sum_{i=1}^{n} \left(\widehat{\boldsymbol{p}}_i - \widehat{\boldsymbol{p}}\right) \tau_i \frac{\widehat{\boldsymbol{p}}_{si}^2}{\|\boldsymbol{p}_{si}\|} \boldsymbol{p}_{ssi}
\end{aligned}$$

These can be rewritten in terms of the backbone's kinematic variables as

$$\begin{aligned}
\boldsymbol{f}_c &= \boldsymbol{R}\left(\boldsymbol{a} + \boldsymbol{A}\boldsymbol{v}_s + \boldsymbol{G}\boldsymbol{u}_s\right) \\
\boldsymbol{l}_c &= \boldsymbol{R}\left(\boldsymbol{b} + \boldsymbol{G}^\top \boldsymbol{v}_s + \boldsymbol{H}\boldsymbol{u}_s\right).
\end{aligned}$$

where expressions needed to calculate $\boldsymbol{a}$, $\boldsymbol{b}$, $\boldsymbol{A}$, $\boldsymbol{G}$, and $\boldsymbol{H}$ are defined below:

$$
\begin{aligned}
(\boldsymbol{p}_{si})^b &= \widehat{\boldsymbol{u}}\boldsymbol{r}_i + \boldsymbol{r}_{si} + \boldsymbol{v} \\
\boldsymbol{A}_i &= -\tau_i \frac{\left(((\boldsymbol{p}_{si})^b)^\wedge\right)^2}{\|(\boldsymbol{p}_{si})^b\|^3} \\
\boldsymbol{G}_i &= -\boldsymbol{A}_i \widehat{\boldsymbol{r}}_i \\
\boldsymbol{a}_i &= \boldsymbol{A}_i \left[\widehat{\boldsymbol{u}}\left((\boldsymbol{p}_{si})^b + \boldsymbol{r}_{si}\right) + \boldsymbol{r}_{ssi}\right] \\
\boldsymbol{b}_i &= \widehat{\boldsymbol{r}}_i \boldsymbol{a}_i
\end{aligned}
$$

$$
\begin{aligned}
\boldsymbol{a} &= \sum_{i=1}^{n} \boldsymbol{a}_i, \qquad \boldsymbol{b} = \sum_{i=1}^{n} \boldsymbol{b}_i, \qquad \boldsymbol{A} = \sum_{i=1}^{n} \boldsymbol{A}_i, \\
\boldsymbol{G} &= \sum_{i=1}^{n} \boldsymbol{G}_i, \qquad \boldsymbol{H} = \sum_{i=1}^{n} \widehat{\boldsymbol{r}}_i \boldsymbol{G}_i.
\end{aligned}
$$

The differential equations for internal loading are then

$$
\begin{aligned}
\boldsymbol{n}_s &= -\boldsymbol{R}\left(\boldsymbol{a} + \boldsymbol{A}\boldsymbol{v}_s + \boldsymbol{G}\boldsymbol{u}_s\right) - \bar{\boldsymbol{f}} \\
\boldsymbol{m}_s &= -\boldsymbol{R}\left(\boldsymbol{b} + \boldsymbol{G}^\top \boldsymbol{v}_s + \boldsymbol{H}\boldsymbol{u}_s\right) + \partial_t(\boldsymbol{R}\rho\boldsymbol{J}\boldsymbol{\omega}) - \widehat{\boldsymbol{p}}_s \boldsymbol{n} - \bar{\boldsymbol{l}},
\end{aligned}
$$

where $\bar{\boldsymbol{f}}$ and $\bar{\boldsymbol{l}}$ represent any distributed loading components not caused by the tendons. These equations are implicit, that is $\boldsymbol{v}_s = \boldsymbol{v}_s(\boldsymbol{n}_s)$ and $\boldsymbol{u}_s = \boldsymbol{u}_s(\boldsymbol{m}_s)$ because differentiating the constitutive equation (3.2) leads to

$$
\begin{aligned}
\boldsymbol{n}_s &= \boldsymbol{R}_s[\boldsymbol{K}_{se}(\boldsymbol{v} - \boldsymbol{v}^*) + \boldsymbol{B}_{se}\boldsymbol{v}_t] + \boldsymbol{R}[\boldsymbol{K}_{s_{se}}(\boldsymbol{v} - \boldsymbol{v}^*) + \boldsymbol{K}_{se}(\boldsymbol{v}_s - \boldsymbol{v}_s^*) + \boldsymbol{B}_{s_{se}}\boldsymbol{v}_t + \boldsymbol{B}_{se}\boldsymbol{v}_{st}] \\
\boldsymbol{m}_s &= \boldsymbol{R}_s[\boldsymbol{K}_{bt}(\boldsymbol{u} - \boldsymbol{u}^*) + \boldsymbol{B}_{bt}\boldsymbol{u}_t] + \boldsymbol{R}[\boldsymbol{K}_{s_{bt}}(\boldsymbol{u} - \boldsymbol{u}^*) + \boldsymbol{K}_{bt}(\boldsymbol{u}_s - \boldsymbol{u}_s^*) + \boldsymbol{B}_{s_{bt}}\boldsymbol{u}_t + \boldsymbol{B}_{bt}\boldsymbol{u}_{st}].
\end{aligned}
$$

We face a choice to solve for the strain equations $\boldsymbol{v}_s$ and $\boldsymbol{u}_s$ or the force equations $\boldsymbol{n}_s$ and $\boldsymbol{m}_s$. Convenience motivates us to choose $\boldsymbol{v}$ and $\boldsymbol{u}$ as state variables because the resulting equations are simpler than those for $\boldsymbol{m}$ and $\boldsymbol{n}$. We apply the time discretization, rotate the equations, and introduce intermediate variables so that the internal loading differential

equations are described by

$$\boldsymbol{R}^\top \boldsymbol{n}_s = -\boldsymbol{A}\boldsymbol{v}_s - \boldsymbol{G}\boldsymbol{u}_s + \boldsymbol{\Lambda}_{\boldsymbol{n}}$$
$$\boldsymbol{R}^\top \boldsymbol{m}_s = -\boldsymbol{G}^\top \boldsymbol{v}_s - \boldsymbol{H}\boldsymbol{u}_s + \boldsymbol{\Lambda}_{\boldsymbol{m}},$$

and the constitutive equation by

$$\boldsymbol{R}^\top \boldsymbol{n}_s = (\boldsymbol{K}_{se} + c_0 \boldsymbol{B}_{se})\boldsymbol{v}_s + \boldsymbol{\Gamma}_{\boldsymbol{v}}$$
$$\boldsymbol{R}^\top \boldsymbol{m}_s = (\boldsymbol{K}_{bt} + c_0 \boldsymbol{B}_{bt})\boldsymbol{u}_s + \boldsymbol{\Gamma}_{\boldsymbol{u}},$$

where

$$\boldsymbol{\Lambda}_{\boldsymbol{n}} = -\boldsymbol{a} + \rho A(\widehat{\boldsymbol{\omega}}\boldsymbol{q} + \boldsymbol{q}_t) + \boldsymbol{C}\boldsymbol{q} \odot |\boldsymbol{q}| - \boldsymbol{R}^\top(\rho A \boldsymbol{g} + \bar{\boldsymbol{f}})$$
$$\boldsymbol{\Lambda}_{\boldsymbol{m}} = -\boldsymbol{b} + \rho\left(\widehat{\boldsymbol{\omega}}\boldsymbol{J}\boldsymbol{\omega} + \boldsymbol{J}\boldsymbol{\omega}_t\right) - \widehat{\boldsymbol{v}}\boldsymbol{n}^b - \boldsymbol{R}^\top \bar{\boldsymbol{l}}$$
$$\boldsymbol{\Gamma}_{\boldsymbol{v}} = \widehat{\boldsymbol{u}}\boldsymbol{n}^b + \boldsymbol{K}_{s_{se}}(\boldsymbol{v} - \boldsymbol{v}^*) - \boldsymbol{K}_{se}\boldsymbol{v}_s^* + \boldsymbol{B}_{s_{se}}\boldsymbol{v}_t + \boldsymbol{B}_{se}\overset{\natural}{\boldsymbol{v}}_s$$
$$\boldsymbol{\Gamma}_{\boldsymbol{u}} = \widehat{\boldsymbol{u}}\boldsymbol{m}^b + \boldsymbol{K}_{s_{bt}}(\boldsymbol{u} - \boldsymbol{u}^*) - \boldsymbol{K}_{bt}\boldsymbol{u}_s^* + \boldsymbol{B}_{s_{bt}}\boldsymbol{u}_t + \boldsymbol{B}_{bt}\overset{\natural}{\boldsymbol{u}}_s.$$

Note that the internal loads are calculated by

$$\boldsymbol{n}^b = \boldsymbol{K}_{se}(\boldsymbol{v} - \boldsymbol{v}^*) + \boldsymbol{B}_{se}\boldsymbol{v}_t$$
$$\boldsymbol{m}^b = \boldsymbol{K}_{bt}(\boldsymbol{u} - \boldsymbol{u}^*) + \boldsymbol{B}_{bt}\boldsymbol{u}_t.$$

With the four coupled equations above, we may solve a linear system for $\boldsymbol{v}_s$ and $\boldsymbol{u}_s$. The resulting set of ODEs for the tendon robot is

$$\begin{aligned}
\boldsymbol{p}_s &= \boldsymbol{R}\boldsymbol{v} \\
\boldsymbol{R}_s &= \boldsymbol{R}\widehat{\boldsymbol{u}} \\
\begin{bmatrix} \boldsymbol{v}_s \\ \boldsymbol{u}_s \end{bmatrix} &= \boldsymbol{\Phi}^{-1} \begin{bmatrix} -\boldsymbol{\Gamma}_{\boldsymbol{v}} + \boldsymbol{\Lambda}_{\boldsymbol{n}} \\ -\boldsymbol{\Gamma}_{\boldsymbol{u}} + \boldsymbol{\Lambda}_{\boldsymbol{m}} \end{bmatrix} \\
\boldsymbol{q}_s &= \boldsymbol{v}_t - \widehat{\boldsymbol{u}}\boldsymbol{q} + \widehat{\boldsymbol{\omega}}\boldsymbol{v} \\
\boldsymbol{\omega}_s &= \boldsymbol{u}_t - \widehat{\boldsymbol{u}}\boldsymbol{\omega},
\end{aligned} \tag{3.14}$$

where

$$\boldsymbol{\Phi} = \begin{bmatrix} (\boldsymbol{K}_{se} + c_0\boldsymbol{B}_{se} + \boldsymbol{A}) & \boldsymbol{G} \\ \boldsymbol{G}^{\top} & (\boldsymbol{K}_{bt} + c_0\boldsymbol{B}_{bt} + \boldsymbol{H}) \end{bmatrix}.$$

Obtaining the steady-state ODE is straightforward. Regarding the practical implementation of this system, a matrix decomposition may take advantage of the symmetry of the 6x6 inverted matrix.

We note that depending on the scenario and the value of $\boldsymbol{B}_{se}$, stiff shear and extension dynamics can cause convergence issues for the tendon system. In many situations it is appropriate to neglect the shear and extension strains. If shear and extension are neglected, only $\boldsymbol{u}_s$ is directly dependent on solving a linear system so that

$$\begin{aligned}
\boldsymbol{u}_s &= (\boldsymbol{K}_{bt} + c_0\boldsymbol{B}_{bt} + \boldsymbol{H})^{-1} \left[-\boldsymbol{\Gamma}_{\boldsymbol{u}} - \boldsymbol{b} + \rho \left(\widehat{\boldsymbol{\omega}}\boldsymbol{J}\boldsymbol{\omega} + \boldsymbol{J}\boldsymbol{\omega}_t \right) - \widehat{\boldsymbol{e}}_3\boldsymbol{n}^b - \boldsymbol{R}^{\top}\bar{\boldsymbol{l}} \right] \\
\boldsymbol{n}_s &= \boldsymbol{R} \left[-\boldsymbol{G}\boldsymbol{u}_s - \boldsymbol{a} + \rho A \left(\widehat{\boldsymbol{\omega}}\boldsymbol{q} + \boldsymbol{q}_t \right) + \boldsymbol{C}\boldsymbol{q} \odot |\boldsymbol{q}| \right] - \rho A\boldsymbol{g} - \bar{\boldsymbol{f}}.
\end{aligned}$$

The state equation for $\boldsymbol{v}_s$ is replaced by $\boldsymbol{n}_s$. The other equations are unaffected, except of course that $\boldsymbol{v} = \boldsymbol{e}_3$ and $\boldsymbol{v}_t = \boldsymbol{0}$. Shear and extension are included in the model implementations here for the sake of generality.

3.3.2 Boundary Conditions

Aside from the ODE describing the continuous behavior of the backbone and tendons, the boundary conditions must account for the final rigid attachment of a tendon to the backbone which causes step changes in the backbone internal loading as described in [97]. The point loads are

$$\boldsymbol{F}_i^b = -\tau_i \frac{\boldsymbol{p}_{si}^b(s^-)}{\left\|\boldsymbol{p}_{si}^b(s^-)\right\|} \quad \text{and} \quad \boldsymbol{L}_i^b = \widehat{\boldsymbol{r}}_i \boldsymbol{F}_i^b$$

which causes a step change in the internal loading by

$$\boldsymbol{n}^b(s^+) = \boldsymbol{n}^b(s^-) - \boldsymbol{F}_i^b \quad \text{and} \quad \boldsymbol{m}^b(s^+) = \boldsymbol{m}^b(s^-) - \boldsymbol{L}_i^b.$$

Any rigid-body dynamics of the end effector are coupled to the robot system through the distal boundary conditions by

$$\begin{aligned} \boldsymbol{F}_e(t) + m_e \boldsymbol{g} - \boldsymbol{n}(t, L^+) &= m_e \boldsymbol{a}_e(t) \\ \boldsymbol{M}_e(t) - \boldsymbol{m}(t, L^+) &= \boldsymbol{R}_e(t) \boldsymbol{J}_e \boldsymbol{R}_e^\top(t) \boldsymbol{\omega}_{te}(t) + \widehat{\boldsymbol{\omega}}_e(t) \boldsymbol{R}_e(t) \boldsymbol{J}_e \boldsymbol{R}_e^\top(t) \boldsymbol{\omega}_e(t). \end{aligned}$$

The control input varies depending on whether one has control of the tendon tensions τ_i or the tendon displacements. If the tensions are controlled, then the shooting problem is the same as the cantilever rod, that is

$$\begin{aligned} \boldsymbol{G} &= \begin{bmatrix} \boldsymbol{n}(t,0)^\top & \boldsymbol{m}(t,0)^\top \end{bmatrix}^\top \\ \boldsymbol{E}(\boldsymbol{G}) &= \begin{bmatrix} (\boldsymbol{E}^F)^\top & (\boldsymbol{E}^M)^\top \end{bmatrix}^\top. \end{aligned}$$

The case of tendon displacement control is more complicated and is described in [73]. The tendon has an unstretched length l_i^* and compliance c_i so that the stretched length is

$$l_i = (1 + c_i \tau_i) l_i^*.$$

Let the displacement input l_i^q be the distance from the base plate to the actuator where the tendon is attached, and l_i^B be the length of tendon found by integrating the backbone. Then the total length is the sum of tendon segments along the backbone and behind the baseplate,

$$l_i = l_i^q + l_i^B.$$

Assuming the tendons do not become slack, we may guess τ_i so that

$$\boldsymbol{G} = \begin{bmatrix} \boldsymbol{n}(t,0)^\top & \boldsymbol{m}(t,0)^\top & \tau_1(t) & \dots & \tau_n(t) \end{bmatrix}^\top$$

and calculate a residual

$$\boldsymbol{E}(\boldsymbol{G}) = \begin{bmatrix} (\boldsymbol{E}^F)^\top & (\boldsymbol{E}^M)^\top & E_1^l & \dots & E_n^l \end{bmatrix}^\top,$$

where

$$E_i^l = (1 + c_i\tau_i)l_i^* - l_i^q - l_i^B.$$

However, when implementing displacement-controlled actuation, development of slack in one or more tendons is possible, and when a tendon goes slack the integrated path arc length $l_i^q + l_i^B$ will be less than the actual tendon length l_i^*. We can account for this by introducing an unknown variable β_i for the amount of slack. The length constraint is then

$$(1 + c_i\tau_i)l_i^* = l_i^q + l_i^B + \beta_i. \tag{3.15}$$

This introduces n additional unknown variables, but we can reduce the number of unknowns back to again have a square system by recognizing that tendon tension and slack are mutually exclusive and restricted to be positive semidefinite; that is, $\beta_i > 0 \implies \tau_i = 0$ and

$\tau_i > 0 \implies \beta_i = 0$. Thus, we can represent both effects with a single unknown variable γ_i:

$$\tau_i = \begin{cases} \gamma_i^2, & \gamma_i \geq 0 \\ 0, & \gamma_i < 0 \end{cases}, \quad \text{and} \quad \beta_i = \begin{cases} 0, & \gamma_i \geq 0 \\ \gamma_i^2, & \gamma_i < 0 \end{cases}. \tag{3.16}$$

Parameterizing tension and slack with a single, continuous variable eliminates the need to explicitly identify the changing slack state and the appropriate constraints during model solves. The squaring of γ_i is a choice made so that τ_i and β_i have continuous first derivatives with respect to γ_i. We want to model the slack constraint, but implementing the slack error in an optimization routine as

$$E_i^l = (1 + c_i\tau_i)l_i^* - l_i^q - l_i^B - \beta_i$$

is ineffective because the shooting method Jacobian will include a partial derivative

$$\frac{\partial E_i^l}{\partial \tau_i} = c_i l_i^* - \partial l_i^B / \partial \tau_i,$$

which is nearly zero for the common case of negligible tendon compliance. The lack of a slope leads optimization routines to stall. Instead, one can implement an error

$$E_i^{l'} = (1 + \tau_i)E_i^l.$$

This seems to result in an acceptable gradient, and we note that the $(1 + \tau_i)$ term will not nullify E_i^l because τ_i is positive semi-definite.

3.3.3 Simulation

We applied this dynamic tendon robot model to simulate a tendon robot performing an object transfer task as shown in Figure 3.11. Depending on the design and scale of a tendon robot, the inertial dynamics can give rise to significant vibrations even with slow actuator movements. After the simulated tendon robot picks up the object, its movement is highly influenced by the inertial dynamics from the added tip mass. In this simulation, the actuation

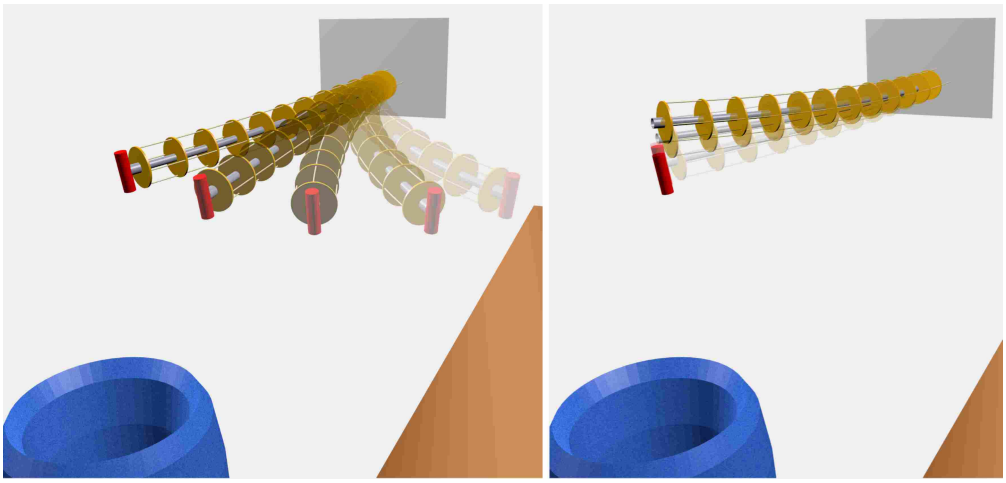


Figure 3.11: To demonstrate dynamic tendon-robot motion, a robot with four tendons and a "vacuum gripper" removes a weighted object from a table and drops it into a bin. On the left, the robot moves the object towards the bin, and on the right the arm swings upward after releasing the object. This scenario is shown in a video attachment.

to reach the object as well as the loading and unloading associated with picking it up and dropping it all involve underactuated robot dynamics captured by our model in real time.

The simulated robot contains four tendons offset from the backbone by 9.5mm with an angular separation of 90° about the backbone. The backbone length was 0.24m. The backbone had Young's modulus $E = 207\text{GPa}$, $r = 0.4\text{mm}$, and $\rho = 1.6 \times 10^4$ kg/m^3, which is about twice as heavy as steel to account for both the backbone and the support disks. There is some damping with $\boldsymbol{C} = \boldsymbol{I} * 0.03$ kg/m^2 and $\boldsymbol{B}_{bt} = \boldsymbol{I} \times 10^{-6}$ Nm^{2}s. The object had a mass of 1g. The ODE integration was performed with Euler's method using 200 points. The time discretization used the BDF-α method with $\delta t = 1/60\text{s}$ and $\alpha = -0.03$. The simulation achieved a real-time speed ratio greater than 3.5.

3.3.4 Experimental Comparison

To evaluate the tendon robot model, a robot was constructed from a spring steel backbone with acrylic spacer disks and a single Kevlar tendon. The tendon displacement is controlled by a geared servo motor (Dynamixel MX-28-AT), which is a prescribed-displacement input as studied in [73]. A step input is applied to pull the robot upward, then after steady state is reached, another step input returns to the original displacement as shown in Figure 3.12.

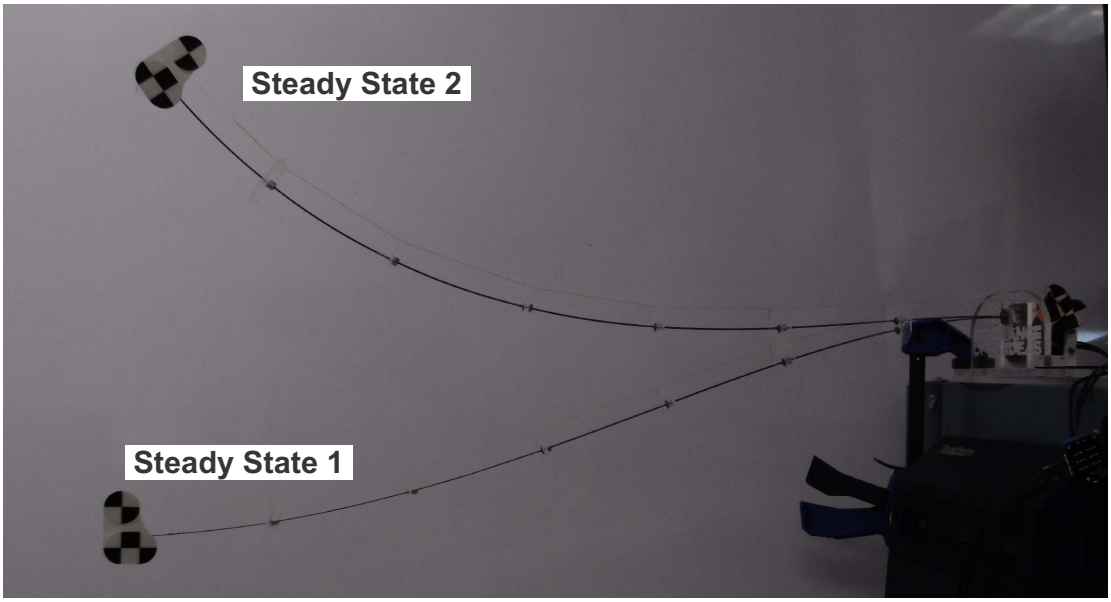
Friction between the robot and the tendons is more significant in bent configurations due to higher normal forces on the tendons. To approximate the effect of tendon friction, we define a simple distributed damping force and moment applied to the backbone. If the tendon is routed in a straight path parallel to the backbone, the damping force is approximately proportional to the tension τ (still assumed constant), magnitude of the backbone curvature $||\boldsymbol{u}||$, and relative tangential velocity ν_i between the tendon and its channel as follows:

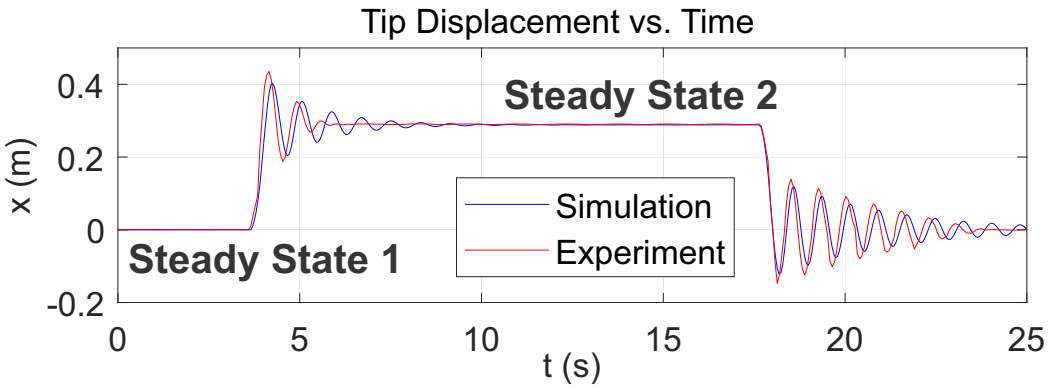
$$\boldsymbol{f}_{f,i} = -\beta\tau \left\|\boldsymbol{u}\right\| \nu_i \boldsymbol{R}\boldsymbol{e}_3 \quad \text{and} \quad \boldsymbol{l}_{f,i} = \boldsymbol{r}_i \times \boldsymbol{f}_{f,i},$$

where β is a damping coefficient and ν_i is computed from past time steps.

The backbone has a length of 0.7144m and diameter of 0.00135m. The tendon is at a constant offset of 0.0151m from the backbone and there is a distance of 0.0518m from the baseplate to the motor. The frame convention defines $\boldsymbol{r} = 0.0151\boldsymbol{e}_1$m and $\boldsymbol{g} = -9.81\boldsymbol{e}_1$ m/s^2. The whole arm was weighed to be 0.034kg. The initial displacement holds the tip tangent to the z-axis. The tip position and motor response were measured by a stereoscopic camera system (MicronTracker H3-60, Claron Technology Inc.). The tendon is retracted 0.01619m, and the motor response to the commanded step was nearly a linear ramp occurring over 0.31s for both upward and downward motions. The tendon compliance was calibrated to a value of 1.6×10^{-3} m/m. The simulation used BDF2 implicit time discretization with $\delta t = 0.05$s and Euler's method spatial integration with $N = 200$ points. The friction coefficients are $\boldsymbol{C} = \boldsymbol{I} \times 10^{-4}$ kg/m^2, $\boldsymbol{B}_{bt} = \boldsymbol{I} \times 5 \times 10^{-4}$ Nm^{2}s, and $\beta = 5$ s/m.

The model response roughly lines up with observed behavior. The friction model is simplified since it does not include static friction effects, but does result in greater damping for the retracted motor input. The magnitudes and steady state behavior are reasonable.





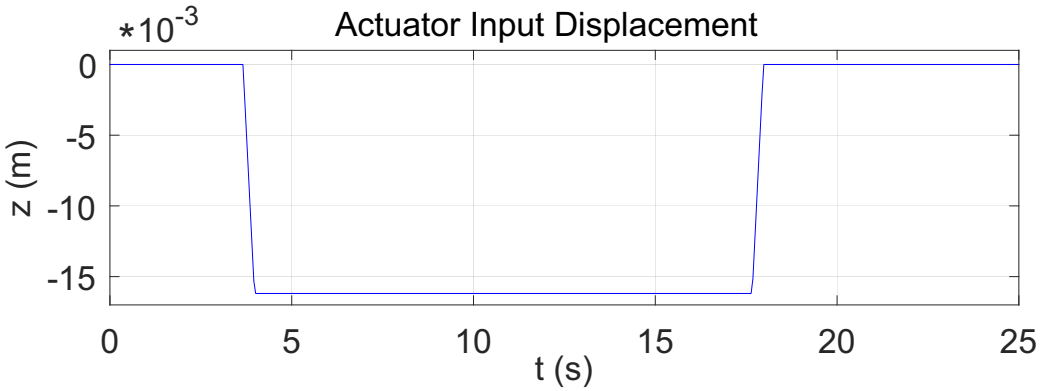


Figure 3.12: The dynamic model is validated against this tendon robot. The attached markers allow the tip position to be measured by a stereoscopic camera system. The composite image shows the steady state configurations before and after the step input. A pair of step inputs is applied to the tendon displacement, resulting in dynamic tip motion.

3.4 Fluidic Soft Robot

3.4.1 Differential Equations

We consider a soft robot with one or more hollow actuation chambers offset from the neutral axis. There is a vector $\boldsymbol{r}_i$ from the cross-section center of mass to the center of the ith chamber, which is similar to the tendon robot variable $\boldsymbol{r}_i$. We restrict our attention to cases with $\boldsymbol{r}_{si} = \mathbf{0}$ so that the channel has a constant offset from the cross-section centroid. Fluid pressure is applied in the chamber, which results in a bending motion of the robot. We assume quasi-static fluid dynamics in the chamber so that there is a single uniform pressure $P_i(t)$. The situation is illustrated in Figure 3.13.

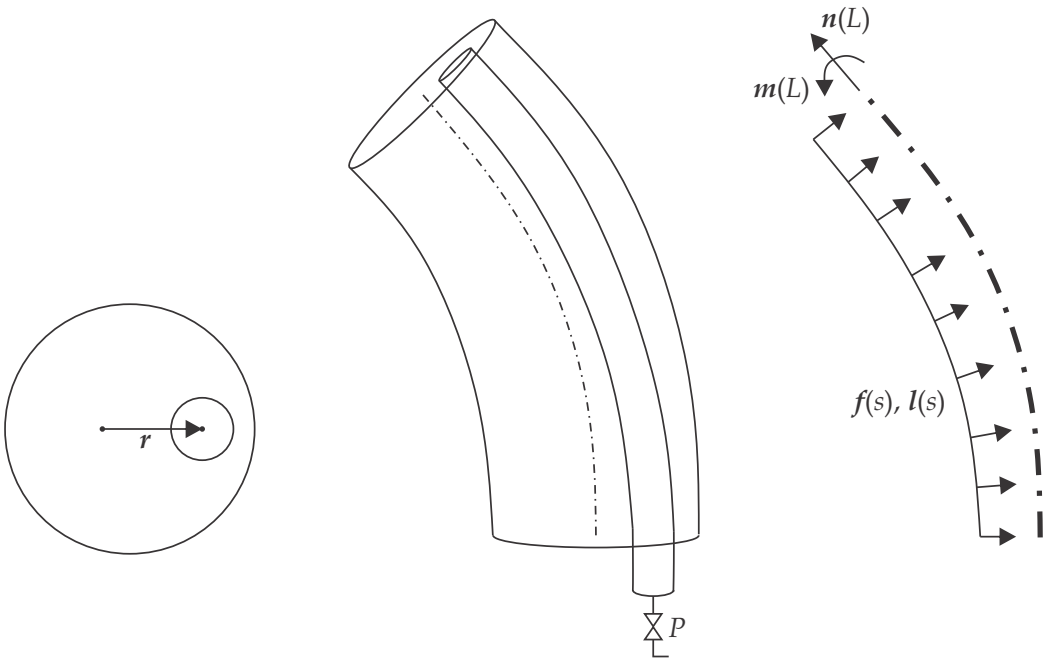


Figure 3.13: A soft elastic rod has a hollow chamber offset from the central axis which is subject to some quasi-static and uniform pressure $P(t)$. The chamber is offset from the central axis by a vector $\boldsymbol{r}(s)$ in the local frame. The pressure results in a force at the cap of the chamber, which also generates a moment due to the offset. The outer curve of the chamber has more surface area than the inner curve, resulting in a net distributed force and moment.

The distributed loading and point wrenches caused by pressurizing the chamber are similar to the effects of a tendon robot, but the two are not quite equivalent because tendon forces are transmitted along the tendon tangent line, while pressure forces act normal to a cross-sectional plane. We assume all chambers extend to the distal end of the robot, with flat chamber caps of area A_i so that the magnitude of the force on the cap is $P_i A_i$. The force and moment vectors applied to the end of the elastic member are then

$$\boldsymbol{F}_i^b = P_i A_i \boldsymbol{e}_3$$
$$\boldsymbol{L}_i^b = \widehat{\boldsymbol{r}}_i \boldsymbol{F}_i^b.$$

The wrenches at the chamber ends cause a point change in the internal loading by

$$\boldsymbol{n}^b(L) = \boldsymbol{n}^b(L^-) - \sum_{i=1}^{n} \boldsymbol{F}_i^b$$
$$\boldsymbol{m}^b(L) = \boldsymbol{m}^b(L^-) - \sum_{i=1}^{n} \boldsymbol{L}_i^b.$$

To obtain an expression for the distributed loading, we will consider the force and moment balance for a cut section of the soft robot as shown in Figure 3.14.

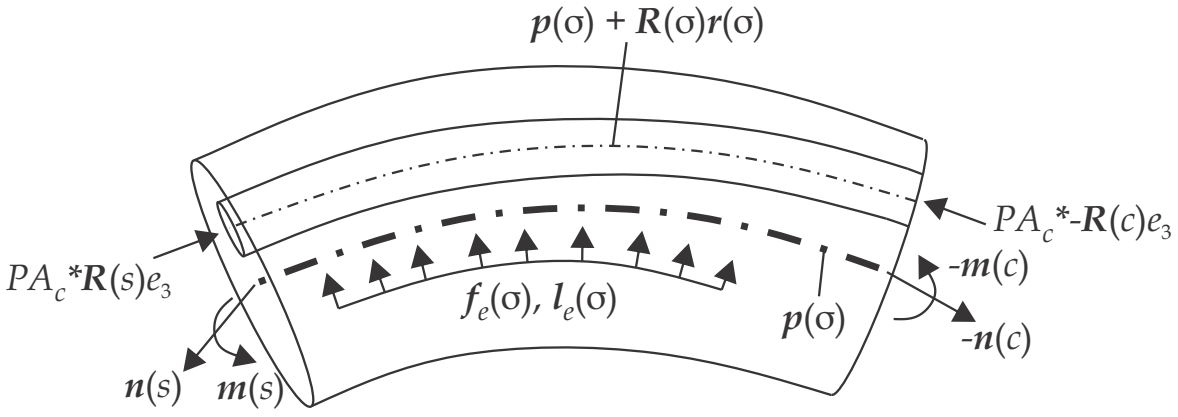


Figure 3.14: The forces and moments acting on a segment of the robot are shown for cuts at s and c with $s > c$. The segment is subject to internal forces and moments at the centerline of the cuts, forces maintaining the pressure where the chamber is cut, and external distributed loading.

We write the acceleration $\boldsymbol{p}_{tt}$ in terms of the local frame velocity $\boldsymbol{q}$ so that

$$\boldsymbol{p}_{tt} \equiv \boldsymbol{R}\left(\widehat{\boldsymbol{\omega}}\boldsymbol{q} + \boldsymbol{q}_t\right).$$

The dynamic force balance on the section is given by

$$\int_c^s \boldsymbol{f}_e(\sigma)d\sigma + \boldsymbol{n}(s) - \boldsymbol{n}(c) - \sum_{i=1}^{n} P_i A_i \left[\boldsymbol{R}(s) - \boldsymbol{R}(c)\right]\boldsymbol{e}_3 = \int_c^s \rho A \boldsymbol{R}\left(\widehat{\boldsymbol{\omega}}\boldsymbol{q} + \boldsymbol{q}_t\right)d\sigma,$$

This can be differentiated and simplified to obtain

$$\boldsymbol{n}_s = -\boldsymbol{f}_e + \rho A \boldsymbol{R}\left(\widehat{\boldsymbol{\omega}}\boldsymbol{q} + \boldsymbol{q}_t\right) + \sum_{i=1}^{n} P_i A_i \boldsymbol{R}_s \boldsymbol{e}_3.$$

Similarly, we consider the moment balance

$$\begin{aligned}\boldsymbol{m}(s) - \boldsymbol{m}(c) + \boldsymbol{p}(s)\times\boldsymbol{n}(s) - \boldsymbol{p}(c)\times\boldsymbol{n}(c) + \int_c^s \left[\boldsymbol{l}_e(\sigma) + \boldsymbol{p}(\sigma)\times\boldsymbol{f}_e(\sigma)\right]d\sigma \\ - \sum_{i=1}^{n}\left[\boldsymbol{p}(s) + \boldsymbol{R}(s)\boldsymbol{r}_i\right]\times P_i A_i \boldsymbol{R}(s)\boldsymbol{e}_3 - \left[\boldsymbol{p}(c) + \boldsymbol{R}(c)\boldsymbol{r}_i\right]\times P_i A_i \boldsymbol{R}(c)\boldsymbol{e}_3 \\ = \int_c^s \partial_t\left(\boldsymbol{R}\rho\boldsymbol{J}\boldsymbol{\omega}\right)d\sigma,\end{aligned}$$

which is differentiated to find

$$\boldsymbol{m}_s = -\boldsymbol{l}_e - (\boldsymbol{p}\times\boldsymbol{n})_s - \boldsymbol{p}\times\boldsymbol{f}_e + \partial_t\left(\boldsymbol{R}\rho\boldsymbol{J}\boldsymbol{\omega}\right) + \sum_{i=1}^{n} P_i A_i \frac{\partial}{\partial s}\left[\left(\boldsymbol{p} + \boldsymbol{R}\boldsymbol{r}_i\right)\times\boldsymbol{R}\boldsymbol{e}_3\right].$$

We note that additional terms are present if A_i varies as a function of arc length. Expanding and canceling terms leads to

$$\boldsymbol{m}_s = -\boldsymbol{l}_e - \boldsymbol{p}_s\times\boldsymbol{n} + \partial_t\left(\boldsymbol{R}\rho\boldsymbol{J}\boldsymbol{\omega}\right) + \sum_{i=1}^{n} P_i A_i \boldsymbol{R}\left[\left(\boldsymbol{v} + \widehat{\boldsymbol{u}}\boldsymbol{r}_i\right)\times\boldsymbol{e}_3 + \boldsymbol{r}_i\times\widehat{\boldsymbol{u}}\boldsymbol{e}_3\right].$$

To more clearly indicate the distributed loads caused by the pressure, we define intermediate variables

$$\begin{aligned} \boldsymbol{f}_P &:= -\sum_{i=1}^{n} P_i A_i \boldsymbol{R}_s \boldsymbol{e}_3 \\ \boldsymbol{l}_P &:= -\sum_{i=1}^{n} P_i A_i \boldsymbol{R} \left[(\boldsymbol{v} + \widehat{\boldsymbol{u}} \boldsymbol{r}_i) \times \boldsymbol{e}_3 + \boldsymbol{r}_i \times \widehat{\boldsymbol{u}} \boldsymbol{e}_3 \right] \end{aligned} \tag{3.17}$$

so that the differential equations are

$$\begin{aligned} \boldsymbol{n}_s &= \rho A \boldsymbol{R} \left(\widehat{\boldsymbol{\omega}} \boldsymbol{q} + \boldsymbol{q}_t \right) - \boldsymbol{f}_e - \boldsymbol{f}_P \\ \boldsymbol{m}_s &= \partial_t \left(\boldsymbol{R} \rho \boldsymbol{J} \boldsymbol{\omega} \right) - \boldsymbol{p}_s \times \boldsymbol{n} - \boldsymbol{l}_e - \boldsymbol{l}_P. \end{aligned}$$

With the differential equations formulated, we may consider how the fluid properties influence the boundary conditions. The robot configuration defines the fluid chamber volume by integrating

$$V_i^{\mathrm{I}} = \int_0^L A_i \left\| \frac{\partial}{\partial s} \left(\boldsymbol{p} + \boldsymbol{r}_i \right) \right\| ds,$$

where

$$\left\| \frac{\partial}{\partial s} \left(\boldsymbol{p} + \boldsymbol{r}_i \right) \right\| = \left\| \widehat{\boldsymbol{u}} \boldsymbol{r}_i + \boldsymbol{v} \right\|.$$

Note in this preliminary derivation we consider the cross-sectional geometry constant in time, but future work could consider the dependence of A_i on robot shape and chamber pressure.

The fluid volume is coupled to pressure, temperature, and molar quantity. Modeling approaches vary; it can be adequate to assume direct input control of fluid pressure, but in some cases the fluid response is sufficiently slow that it is more appropriate to use sophisticated fluid dynamics models [79]. We assume uniform fluid pressure subject to the ideal gas law, which has precedent [48, 74]. Although pump dynamics are potentially significant [51], actuation problems are often idealized as independent from the robot system, and we neglect pump dynamics here so that the system input is the mass of fluid in the chamber. We consider two cases: 1) incompressible fluid as in a hydraulic robot, so that

fluid volume is effectively controlled, and 2) compressible fluid as in a pneumatic robot, so that molar quantity of fluid is the input to the system.

3.4.2 Incompressible Working Fluid

In the case of an incompressible working fluid, the volume of fluid inside a chamber could be directly controlled by a positive-displacement pump. We denote the controlled volume by V_i^{C}. For the shooting problem, we may guess the chamber pressures P_i and obtain a residual comparing the controlled volume to the integrated chamber volume V_i^{I}. The error in volume may be poorly scaled since it has units of distance cubed. Our simulation code addresses this by normalizing the error by the volume in the straight configuration so that the error is $E_i^V = (V_i^{\mathrm{I}} - V_i^{\mathrm{C}})/(A_i L)$. The full shooting method residual function has guessed values

$$\boldsymbol{G} = \begin{Bmatrix} \boldsymbol{n}(t,0) & \boldsymbol{m}(t,0) & P_1(t) & \ldots & P_n(t) \end{Bmatrix},$$

and the residual terms include the force and moment balance at the tip

$$\boldsymbol{E}(\boldsymbol{G}) = \begin{Bmatrix} \boldsymbol{E}^F & \boldsymbol{E}^M & E_1^V & \ldots & E_n^V \end{Bmatrix}.$$

3.4.3 Compressible Working Fluid

For compressible working fluids, the boundary conditions must account for a gas law relating the mass of fluid to the pressure. We consider the ideal gas law

$$(P_i + P_{\mathrm{atm}})V_i^{\mathrm{G}} = n_i R_i T_i,$$

where n_i is the moles of gas in the chamber, R_i is the gas constant, and T_i the temperature of the fluid. One may form a shooting method by guessing the pressures P_i and comparing the resulting gas law volume $V_i^{\mathrm{G}} = n_i R_i T_i/(P_i + P_{\mathrm{atm}})$ to the integrated volume V_i^{I}. The compressible and incompressible working fluid models differ in that the compressible model compares V_i^{I} to the volume according to the gas law $V_i^{\mathrm{G}}(n_i)$ involving the controlled mass of fluid, whereas the incompressible model simply compares V_i^{I} to the controlled volume V_i^{C}.

3.4.4 Simulated Comparison

We simulate the change in natural frequency associated with compressibility of the working fluid. We compare two sets of fluid parameters, the first has a controlled amount of air governed by the ideal gas law, and the second has an incompressible fluid of controlled volume. The air has $R = 8.314$ J/(mol K), $P_{\text{atm}} = 101325$Pa, and $T = 20°\text{C} = 293.15$K. There are no parameters for the incompressible fluid. Besides the fluids, the robots have identical parameters $L = 0.1$m, $E = 20$MPa, radius $= 0.015$m, and $\rho = 300\text{kg/m}^3$. There is a single fluid chamber with a radius of 2.5mm and a constant offset from the center of 0.01m in the $-x$ direction, which leads to an offset $\boldsymbol{r}_1 \approx -0.0103\boldsymbol{e}_1$m from the neutral axis. Numerical parameters were $\delta t = 0.01$s, $\alpha = -0.3$, and $N = 50$ for the spatial resolution.

The robots began in the straight configuration. The first simulation linearly increases n to 2×10^{-4}mol, and the second simulation increases the volume of the chamber linearly to $2.65 \times 10^{-6}\text{m}^3$. These two final configurations are equivalent at steady state. For both simulations the changes occur over a period of two seconds, and the dynamic response without any additional actuation is simulated for another two and half seconds. Simulation renders are shown in Figure 3.15. The real-time ratio was in the range of 26-32 for the incompressible fluid and about 28 with air as the fluid. The transverse tip response is shown in Figure 3.16. The limit cycle amplitude is roughly twice as large for the compressible air compared to an incompressible fluid. This indicates that choosing a fluid with greater compressibility results in larger vibrations. The shear and extension strains are significant for this robot; at the final steady-state solution the elongation strain average over arclength is 0.12.

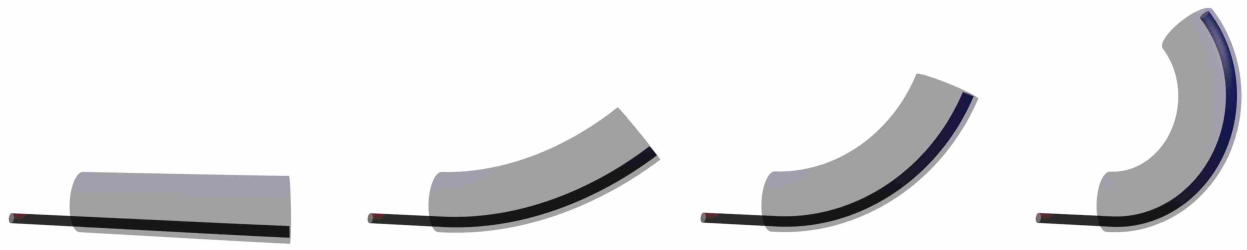


Figure 3.15: Still frames convey the simulated soft robot motion. Starting from zero gauge pressure, additional fluid is added to the chamber, resulting in a bending motion.

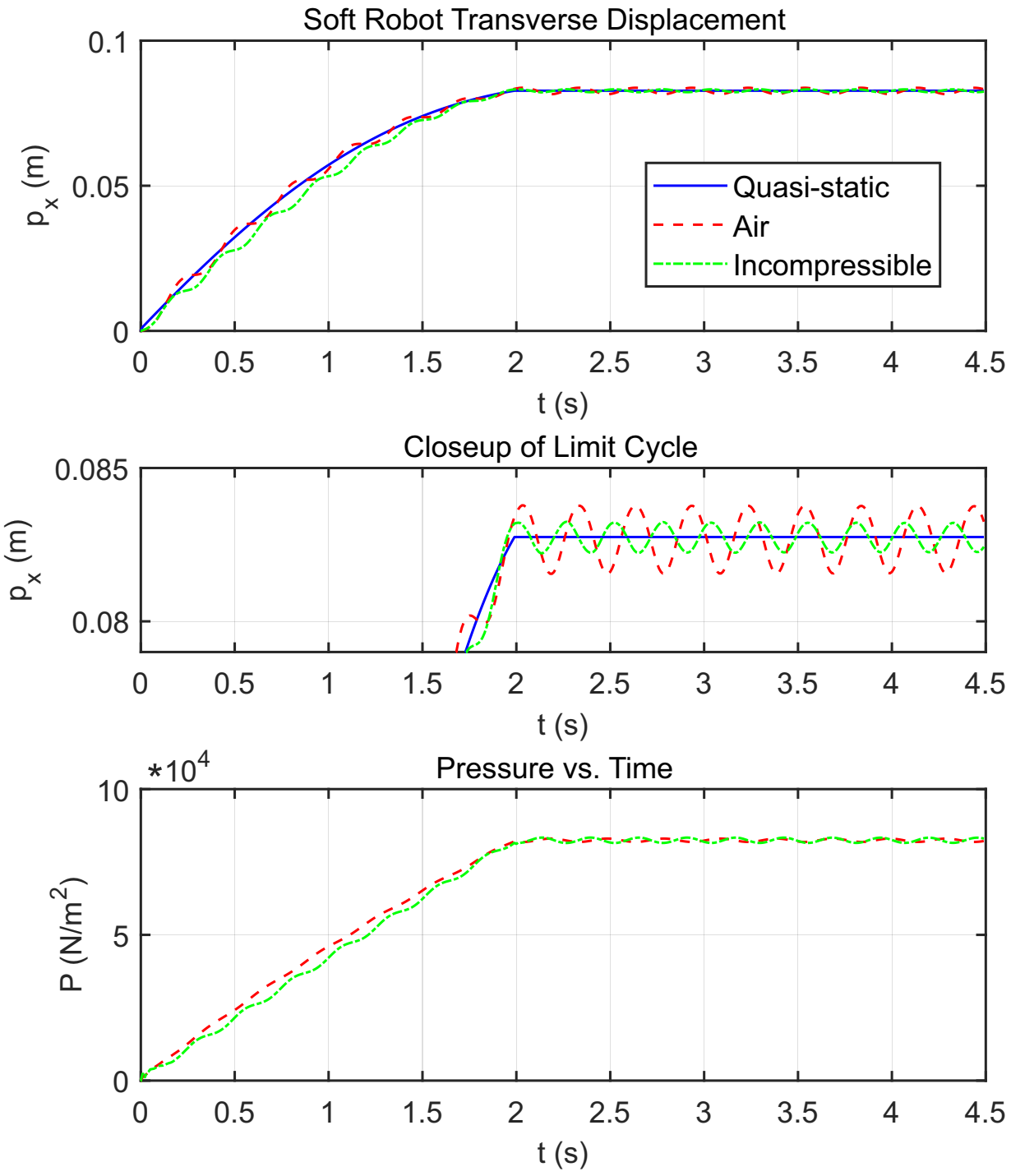


Figure 3.16: The dynamic responses for soft robots with compressible and incompressible fluids are compared. The robot with incompressible fluid experiences less vibration when the fluid quantity is held constant at $t = 2$s. The two fluids have similar fluctuations in pressure.

3.5 Concentric Tube Robot

This section paraphrases the dynamic concentric tube robot model of [111] and offers some additional details. The model is capable of resolving the high-frequency torsional dynamics that occur during unstable "snapping" motions [87] and provides a simulation tool that can track the true robot configuration through such transitions. Experimental verification of the model shows that the torsional oscillations during snapping are captured accurately.

3.5.1 Derivation of Concentric Tube PDEs

We set about to derive the dynamic equations of motion for concentric tubes by starting from the dynamics of a single rod (3.1) with the Kirchhoff inextensibility constraint that $\boldsymbol{v} = \boldsymbol{e}_3$ and applying the kinematic constraints that enforce multiple tubes to be concentric.

Arc-Length Kinematics

Let there be N inextensible tubes. As shown in Figure 3.17, the arc length parameter s is defined so that $\boldsymbol{p}_i(t, 0) = \mathbf{0}$ is the fixed location of a constraining baseplate hole through which all tubes pass. An actuator translation β_i is defined so that the global position of the i^{th} tube base is $[0 \;\; 0 \;\; \beta_i(t)]^\top$. Note that β_i will be a negative number since the actuators are behind the baseplate. Each tube has a total length of l_i.

Note that our convention of prescribing $s = 0$ at the baseplate means that a particular value of the parameter s will describe different material tube points over time since the tubes can slide in and out of the base plate as they are actuated. This choice departs slightly from a conventional Cosserat rod framework where s would correspond to a material point, but it is consistent with prior concentric-tube robot models and is more convenient for formulating the kinematics. To reduce the complexity of the derivation, we assume the insertion speed of the motors is slow so that $\beta_{i,t} \approx 0$. This quasi-static insertion speed allows us to ignore some terms arising from the chain rule and is reasonable given that typical actuator insertion motions are many orders of magnitude slower than the inertial dynamics our model describes.

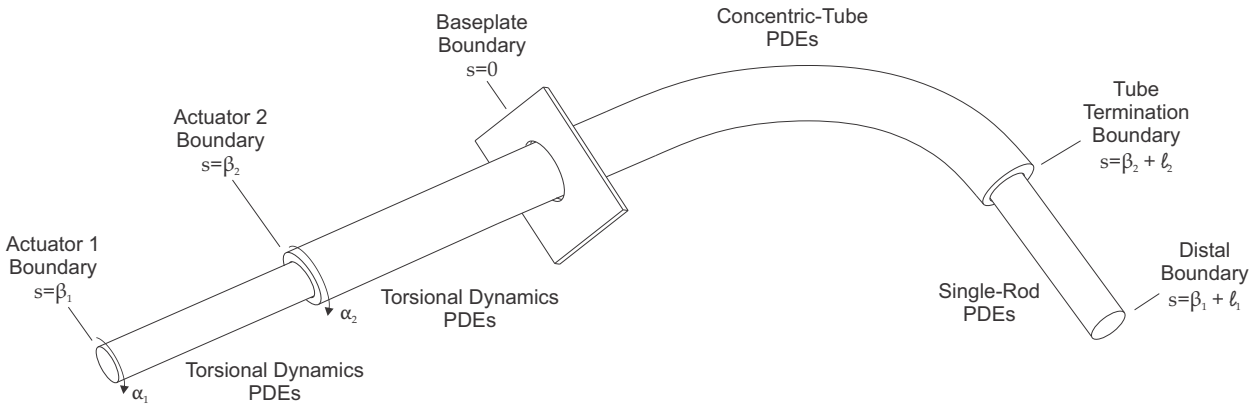


Figure 3.17: A concentric tube robot sketch is annotated to describe the PDE boundary value problem. The equations of motion for a concentric tube collection are given by (3.28), and a simplified set of PDEs (3.31) is obtained under the assumption that tubes are held straight below the base.

The tube indices are ordered so that a larger index corresponds to a larger cross section, i.e. tube 1 is the innermost tube and tube 2 is the second innermost tube. We restrict our attention to configurations where $\beta_i < \beta_j$ and $\beta_i + l_i > \beta_j + l_j$ for $i < j$ so that transition points are always caused by the termination of the outermost tube. The concentric constraint is that all tubes have the same centerline, which is expressed by the equation

$$\boldsymbol{p}_i(t,s) = \boldsymbol{p}_1(t,s) \qquad \forall s \in [\beta_i \quad \beta_i + l_i]. \tag{3.18}$$

This equation may be differentiated with respect to arc length (s) to obtain the constraint that the tube tangents must be aligned

$$\boldsymbol{R}_i \boldsymbol{e}_3 = \boldsymbol{R}_1 \boldsymbol{e}_3. \tag{3.19}$$

This implies that the tube rotation matrices only differ by a rotation about their common z-axes. Thus we define an angle θ_i such that

$$\boldsymbol{R}_i = \boldsymbol{R}_1 \boldsymbol{R}_z(\theta_i)$$

where

$$\boldsymbol{R}_z(\theta_i) = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 \\ \sin\theta_i & \cos\theta_i & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

and $\theta_1 = 0$ by definition. Substituting this relationship into the definition of the curvature of the i^{th} tube results in

$$\boldsymbol{u}_i = \left(\boldsymbol{R}_i^\top \boldsymbol{R}_{i,s}\right)^\vee = \boldsymbol{R}_z^\top(\theta_i)\boldsymbol{u}_1 + \theta_{i,s}\boldsymbol{e}_3 \tag{3.20}$$

The third component of the above equation defines the arc length derivative of θ_i as the difference between the tube torsional strains:

$$\theta_{i,s} = u_{i,z} - u_{1,z}. \tag{3.21}$$

where the subscript z denotes the third (z-axis) component of a vector expressed in the body frame throughout the paper. The above description of the arc-length kinematics is common to the prior static models of concentric tube robots, and more detail can be found in [94].

Relative Angular Velocities

The concentric constraint (3.18) implies that all tubes have the same global linear velocity $\boldsymbol{p}_{i,t}$ and acceleration $\boldsymbol{p}_{i,tt}$. Analogous to (3.20), since the tube rotation matrices share the same z-axis, the body-frame angular velocities are related by

$$\boldsymbol{\omega}_i = \left(\boldsymbol{R}_i^\top \boldsymbol{R}_{i,t}\right)^\vee = \boldsymbol{R}_z^\top(\theta_i)\boldsymbol{\omega}_1 + \theta_{i,t}\boldsymbol{e}_3 \tag{3.22}$$

the third component of which is

$$\theta_{i,t} = \omega_{i,z} - \omega_{1,z}.$$

For convenience, and to eventually arrive at a first-order system of PDEs, we define a new state variable $\gamma_i := \theta_{i,t}$ representing the difference between the z-axis angular velocities of

tube i and tube 1 such that

$$\boldsymbol{\omega}_i = \boldsymbol{R}_z^\top(\theta_i)\boldsymbol{\omega}_1 + \gamma_i \boldsymbol{e}_3 \tag{3.23}$$

To get the arc-length derivative of γ_i, we differentiate (3.21) with respect to time:

$$\gamma_{i,s} = u_{i,z,t} - u_{1,z,t}. \tag{3.24}$$

Forces and Inertial Dynamics

Next, we consider the dynamic equilibrium of internal forces and moments carried by the tubes. We introduce a variable for the concentric tube robot's total internal force, which is the sum of the global-frame internal force vectors $\boldsymbol{n}_i$ carried by each tube:

$$\boldsymbol{n} := \sum_{i=1}^{N} \boldsymbol{n}_i.$$

This is differentiated and the value of $\boldsymbol{n}_{i,s}$ from (3.1) is substituted to obtain

$$\boldsymbol{n}_s = \sum_{i=1}^{N} \rho_i A_i \boldsymbol{p}_{i,tt} - \boldsymbol{f}_i.$$

The concentric constraint with the assumption of quasi-static actuator motion implies that $\boldsymbol{p}_{i,tt} = \boldsymbol{p}_{1,tt}$ (all tubes share the same linear acceleration in the global frame), and we can differentiate the kinematics to find $\boldsymbol{p}_{1,tt} = \boldsymbol{R}_1\left(\widehat{\omega}_1\boldsymbol{q}_1 + \boldsymbol{q}_{1,t}\right)$ (which is also stated in (3.1). Thus we can write

$$\boldsymbol{n}_s = -\boldsymbol{f} + \boldsymbol{R}_1\left(\widehat{\omega}_1\boldsymbol{q}_1 + \boldsymbol{q}_{1,t}\right)(\rho A), \tag{3.25}$$

where $(\rho A) := \sum_{i=1}^{N} \rho_i A_i$ and $\boldsymbol{f} := \sum_{i=1}^{N} \boldsymbol{f}_i$ is the total external distributed load applied to the robot.

Turning now to moments, we seek a differential equation governing the axial (body-frame z) component of each tube's moment vector (the torsional moment) and an additional equation governing the transverse (body-frame xy) component of the total moment carried

by the robot. Defining $\boldsymbol{m}_i^b$ as the internal moment of tube i expressed in the in the body-frame of tube i we have

$$\boldsymbol{m}_i^b = \boldsymbol{R}_i^\top \boldsymbol{m}_i \quad \text{and} \quad \boldsymbol{m}_{i,s}^b = -\widehat{\boldsymbol{u}}_i \boldsymbol{m}_i^b + \boldsymbol{R}_i^\top \boldsymbol{m}_{i,s}$$

since $\boldsymbol{R}_{i,s}^\top = -\widehat{\boldsymbol{u}}_i \boldsymbol{R}_i^\top$. Now substituting in $\boldsymbol{m}_{i,s}$ from (3.1), selecting only the third (z-axis) component of $\boldsymbol{m}_{i,s}^b$, and neglecting any external distributed moments $\boldsymbol{l}_i$, we write

$$\frac{\partial m_{i,z}}{\partial s} = -\boldsymbol{e}_3^\top \widehat{\boldsymbol{u}}_i \boldsymbol{m}_i^b + \rho_i \boldsymbol{e}_3^\top \left(\widehat{\boldsymbol{\omega}}_i \boldsymbol{J}_i \boldsymbol{\omega}_i + \boldsymbol{J}_i \boldsymbol{\omega}_{i,t} \right)$$

where we have used the properties $\boldsymbol{a}^\top \widehat{\boldsymbol{a}} = \boldsymbol{0}$ and $(\boldsymbol{R}\boldsymbol{a})^\wedge = \boldsymbol{R}\widehat{\boldsymbol{a}}\boldsymbol{R}^\top$ for $\boldsymbol{a} \in \mathbb{R}^3$ and $\boldsymbol{R} \in \mathrm{SO}(3)$ from [69] to reveal that $\boldsymbol{e}_3^\top \boldsymbol{R}_i^\top \widehat{\boldsymbol{p}}_{i,s} \boldsymbol{n}_i = \boldsymbol{0}$. Additional simplifications are gained by recognizing that $\boldsymbol{J}_i$ is the second moment of area tensor of the i^{th} tube cross section expressed in the body-frame:

$$\boldsymbol{J}_i = \begin{bmatrix} I_{xx,i} & 0 & 0 \\ 0 & I_{yy,i} & 0 \\ 0 & 0 & I_{zz,i} \end{bmatrix}$$

The terms in $\boldsymbol{J}_i$ can be calculated for a circular tube with inner diameter ID_i and outer diameter OD_i as

$$I_{xx,i} = I_{yy,i} = \frac{1}{2} I_{zz,i} = I_i = \pi (OD_i^4 - ID_i^4)/64.$$

This then implies $\rho_i \boldsymbol{e}_3^\top \widehat{\boldsymbol{\omega}}_i \boldsymbol{J}_i \boldsymbol{\omega}_i = \boldsymbol{0}$ so that using (3.23) we can write

$$\begin{aligned} \frac{\partial m_{i,z}}{\partial s} &= -\boldsymbol{e}_3^\top \widehat{\boldsymbol{u}}_i \boldsymbol{m}_i^b + \rho_i I_{zz,i} \boldsymbol{\omega}_{i,z,t} \\ &= -\boldsymbol{e}_3^\top \widehat{\boldsymbol{u}}_i \boldsymbol{m}_i^b + 2\rho_i I_i \left(\boldsymbol{\omega}_{1,z,t} + \gamma_{i,t} \right) \end{aligned} \tag{3.26}$$

Finally, to derive equations for the transverse components of the total moment, we first define the concentic tube robot's total internal moment, which is the sum of the global frame internal force vectors $\boldsymbol{m}_i$ carried by each tube:

$$\boldsymbol{m} := \sum_{i=1}^{N} \boldsymbol{m}_i.$$

Substituting the arc length derivative of a dynamic moment balance on each tube, and again neglecting $\boldsymbol{l}_i$ we have

$$\boldsymbol{m}_s = \sum_{i=1}^{N} \rho_i \boldsymbol{R}_i \left(\widehat{\boldsymbol{\omega}}_i \boldsymbol{J}_i \boldsymbol{\omega}_i + \boldsymbol{J}_i \boldsymbol{\omega}_{i,t} \right) - \widehat{\boldsymbol{p}}_{i,s} \boldsymbol{n}_i$$

Now define $\boldsymbol{m}^b$ as the total internal moment written in the body frame of tube 1 (the innermost tube) so that

$$\boldsymbol{m}^b = \boldsymbol{R}_1^\top \boldsymbol{m} \quad \text{and} \quad \boldsymbol{m}_s^b = -\widehat{\boldsymbol{u}}_1 \boldsymbol{m}^b + \boldsymbol{R}_1^\top \boldsymbol{m}_s.$$

Now rewriting $\widehat{\boldsymbol{p}}_{i,s}$ using the concentric constraint (3.19) we have

$$\widehat{\boldsymbol{p}}_{i,s} \boldsymbol{n}_i = \left(\boldsymbol{R}_i \boldsymbol{e}_3 \right)^\wedge \boldsymbol{n}_i = \left(\boldsymbol{R}_1 \boldsymbol{e}_3 \right)^\wedge \boldsymbol{n}_i = \boldsymbol{R}_1 \widehat{\boldsymbol{e}}_3 \boldsymbol{R}_1^\top \boldsymbol{n}_i.$$

Using this we rewrite $\boldsymbol{m}_s^b$ as

$$\boldsymbol{m}_s^b = -\widehat{\boldsymbol{u}}_1 \boldsymbol{m}^b - \widehat{\boldsymbol{e}}_3 \boldsymbol{R}_1^\top \boldsymbol{n} + \sum_{i=1}^{N} \rho_i \boldsymbol{R}_z(\theta_i) \left[\widehat{\boldsymbol{\omega}}_i \boldsymbol{J}_i \boldsymbol{\omega}_i + \boldsymbol{J}_i \boldsymbol{\omega}_{i,t} \right].$$

The terms in the summation are simplified again because the structure of $\boldsymbol{J}_i$ for circular tubes implies $\boldsymbol{R}_z(\theta_i) \boldsymbol{J}_i = \boldsymbol{J}_i \boldsymbol{R}_z(\theta_i)$, and the product product $\widehat{\boldsymbol{\omega}}_i \boldsymbol{J}_i \boldsymbol{\omega}_i$ simplifies to

$$\widehat{\boldsymbol{\omega}}_i \rho_i \begin{bmatrix} I_i & 0 & 0 \\ 0 & I_i & 0 \\ 0 & 0 & 2I_i \end{bmatrix} \boldsymbol{\omega}_i = \rho_i I_i \omega_{i,z} \begin{bmatrix} \omega_{i,y} \\ -\omega_{i,x} \\ 0 \end{bmatrix}$$

After a few more algebraic steps, we can finally extract the x and y components of $\boldsymbol{m}_s^b$ as

$$\frac{\partial \boldsymbol{m}_{xy}^b}{\partial s} = \left\{ -\widehat{\boldsymbol{u}}_1 \boldsymbol{m}^b - \widehat{\boldsymbol{e}}_3 \boldsymbol{R}_1^\top \boldsymbol{n} + (\rho I) \boldsymbol{\omega}_{1,t} \right\}_{xy} + \begin{bmatrix} \omega_{1,y} \\ -\omega_{1,x} \end{bmatrix} \sum_{i=1}^{N} \rho_i I_i (\omega_{1,z} + \gamma_i) \tag{3.27}$$

where $(\rho I) = \sum_{i=1}^{N} \rho_i I_i$.

Summary of Concentric-Tube PDEs

Pulling together all the results in this section, we can succinctly state the set of PDEs for a concentric-tube system in the form of a first-order vector system

$$\boldsymbol{y}_s = \boldsymbol{f}\left(\boldsymbol{y}, \boldsymbol{y}_t\right)$$

where the state vector $\boldsymbol{y}$ contains state variables $\boldsymbol{p}$, $\boldsymbol{R}$, $\boldsymbol{q}$, $\boldsymbol{\omega}$, $\boldsymbol{n}$, $\boldsymbol{m}_{xy}^b$, $m_{1,z}^b$, and $m_{i,z}^b$, θ_i, and γ_i for $i \in [2 \;\; N]$. The full system can be summarized:

$$\begin{aligned}
\boldsymbol{p}_s &= \boldsymbol{R}_1 \boldsymbol{e}_3 \\
\boldsymbol{R}_{1,s} &= \boldsymbol{R}_1 \widehat{\boldsymbol{u}}_1, \\
\boldsymbol{q}_{1,s} &= -\widehat{\boldsymbol{u}}_1 \boldsymbol{q}_1 + \widehat{\boldsymbol{\omega}}_1 \boldsymbol{e}_3 \\
\boldsymbol{\omega}_{1,s} &= \boldsymbol{u}_{1,t} - \widehat{\boldsymbol{u}}_1 \boldsymbol{\omega}_1 \\
\boldsymbol{n}_s &= -\boldsymbol{f} + \boldsymbol{R}_1 \left(\widehat{\boldsymbol{\omega}}_1 \boldsymbol{q}_1 + \boldsymbol{q}_{1,t}\right)(\rho A) \\
\frac{\partial \boldsymbol{m}_{xy}^b}{\partial s} &= \left\{-\widehat{\boldsymbol{u}}_1 \boldsymbol{m}^b - \widehat{\boldsymbol{e}}_3 \boldsymbol{R}_1^\top \boldsymbol{n} + (\rho I)\boldsymbol{\omega}_{1,t}\right\}_{xy} + \begin{bmatrix} \omega_{1,y} \\ -\omega_{1,x} \end{bmatrix} \left((\rho I)\omega_{1,z} + \sum_{i=2}^{N} \rho_i I_i \gamma_i\right) \\
\frac{\partial m_{i,z}^b}{\partial s} &= -\boldsymbol{e}_3^\top \widehat{\boldsymbol{u}}_i \boldsymbol{m}_i^b + 2\rho_i I_i \left(\omega_{1,z,t} + \gamma_{i,t}\right) \\
\theta_{i,s} &= u_{i,z} - u_{1,z} \\
\gamma_{i,s} &= u_{i,z,t} - u_{1,z,t}
\end{aligned} \tag{3.28}$$

This system is analogous to the classical PDEs for a single-rod in (3.1), but it accounts for multiple concentric tubes. As in (3.1), in order to solve the system, we will need to implement a specific constitutive stress-strain relation, as well as a strategy for numerical discretization and solution of the resulting discretized equations. These two additions are developed together in the next section.

3.5.2 Numerical Solution of Concentric-Tube PDEs

In this section we discuss the details of numerical solution of the concentric-tube PDEs stated in (3.28) subject to a specific constitutive equation.

Constitutive Equation

To obtain a complete set of equations for the dynamics of a concentric-tube system, we must postulate a material constitutive equation that relates the kinematic variables $\boldsymbol{u}_i$ to the body-frame internal moments $\boldsymbol{m}_i$. We adopt a linear visco-elastic equation with material damping [59] given earlier by (3.2). The total internal moment in the tube 1 body frame is

$$\boldsymbol{m}^b = \sum_{i=1}^{N} \boldsymbol{R}_z(\theta_i) \left[\boldsymbol{K}_i(\boldsymbol{u}_i - \boldsymbol{u}_i^*) + \boldsymbol{B}_i \boldsymbol{u}_{i,t} \right].$$

Note that this is a differential equation in $\boldsymbol{u}_i$, which was not originally included as a state variable. However, the time discretization strategy in the next section converts this visco-elastic constitutive equation into an algebraic equation which allows us to compute $\boldsymbol{u}_i$ from existing state variables.

Implicit Time Discretization

Applying the implicit time discretization (3.4) to the differential equation defined by the constitutive equation allows one to solve for each tube's independent $u_{i,z}$ in terms of its torsional moment $m_{i,z}$ so that

$$u_{i,z} = \frac{m_{i,z}^b + G_i J_i u_{i,z}^* - B_{i,z} \hat{\mathfrak{h}} u_{i,z}}{G_i J_i + c_0 B_{i,z}}. \tag{3.29}$$

Applying the discretization to the total internal moment yields

$$\boldsymbol{m}^b = \sum_{i=1}^{N} \boldsymbol{R}_z(\theta_i) \left[\boldsymbol{K}_i(\boldsymbol{u}_i - \boldsymbol{u}_i^*) + \boldsymbol{B}_i(c_0 \boldsymbol{u}_i + \hat{\mathfrak{h}} \boldsymbol{u}_i) \right].$$

We can then apply (3.20) and solve for $\boldsymbol{u}_{1,xy}$ as

$$\boldsymbol{u}_{1,xy} = \frac{\boldsymbol{m}_{xy}^b - B_{xy} \hat{\mathfrak{h}} \boldsymbol{u}_{1,xy} + \sum_{i=1}^{N} \boldsymbol{R}_z(\theta_i) E_i I_i \boldsymbol{u}_{i,xy}^*}{\sum_{i=1}^{N} E_i I_i + c_0 B_{i,xy}} \tag{3.30}$$

where $B_{xy} := \sum_{i=1}^{N} B_{i,xy}$ and we have overloaded the symbol $\boldsymbol{R}_z(\theta_i)$ to include its 2×2 version, understood by context.

All terms on the right hand side of (3.28) can now be computed from existing state variables through the algebraic equations (3.29), (3.30), (3.20), and (3.2).

Boundary Conditions

The inputs to the robot are the actuator positions $\beta_i(t)$ and angles α_i. Below the base we assume the tubes are held straight, which results in simplifications to the equations of motion. We use an absolute angle ψ_i to describe the rotation of each tube below the base. The PDE system describing angular rotation and torque is

$$\begin{aligned} \psi_{i,s} &= u_{i,z} \\ \psi_{i,ts} &= u_{i,z,t} \\ m^b_{i,z,s} &= (\rho I_{zz})_i \psi_{i,tt}, \end{aligned} \tag{3.31}$$

where $u_{i,z}$ is calculated as previously described. There is an unknown reaction torque on each actuator $m_{i,z}(t, \beta_i)$. At the base, there is an unknown reaction force $\boldsymbol{n}(t, 0)$ and transverse moment $\boldsymbol{m}_{xy}(t, 0)$. The baseplate is stationary and arbitrarily defined at the origin so that $\boldsymbol{p}(t, 0) = \boldsymbol{q}(t, 0) = \boldsymbol{0}$. While we have neglected insertion speed, we still account for the axial angular velocity of the tube bases so that only the transverse angular velocities are zero at the base, $\boldsymbol{\omega}_{1,xy}(t, 0) = \boldsymbol{0}$. The main system is coupled to the system below the base so that

$$\begin{aligned} \boldsymbol{R}_1(t, 0) &= \boldsymbol{R}_z[\psi_1(t, 0)] \\ \omega_{1,z}(t, 0) &= \psi_{1,t}(t, 0) \\ \theta_i(t, 0) &= \psi_i(t, 0) - \psi_1(t, 0) \\ \gamma_i(t, 0) &= \psi_{i,t}(t, 0) - \psi_{1,t}(t, 0). \end{aligned} \tag{3.32}$$

At distal tube ends, we can use boundary conditions to prescribe external point forces and moments as well as coupling to the rigid body dynamics of external objects the robot is manipulating. For example, suppose that at the end of tube i there are an external force $\boldsymbol{F}$ and moment $\boldsymbol{M}$ and a coupled rigid body with mass m_i and mass moment of inertia $\boldsymbol{H}$

symmetric about the tube axis:

$$\boldsymbol{H} = \begin{bmatrix} H_T & 0 & 0 \\ 0 & H_T & 0 \\ 0 & 0 & H_A \end{bmatrix}.$$

A force and moment balance on the attached rigid body in the body frame of tube 1 yields

$$\begin{aligned} \boldsymbol{F}(t,s) + \boldsymbol{n}(t,s^+) - \boldsymbol{n}(t,s^-) &= m_i \boldsymbol{p}_{tt} \\ \boldsymbol{M}^b_{xy}(s) + \boldsymbol{m}^b_{xy}(s^+) - \boldsymbol{m}^b_{xy}(s^-) &= \left(\boldsymbol{H}_i \bar{\boldsymbol{\omega}}_{i,t} + \widehat{\bar{\boldsymbol{\omega}}}_i \boldsymbol{H}_i \bar{\boldsymbol{\omega}}_i, \right)_{xy} \\ M_z(t) - m^b_{i,z}(t, \beta_i + l_i) &= H_{i,A} \omega_{i,z,t}, \end{aligned} \tag{3.33}$$

where for convenience we have defined $\bar{\boldsymbol{\omega}}_i := \boldsymbol{R}_z^\top(\theta_i)\boldsymbol{\omega}_i$. The termination of the final tube requires the entire distal wrench be balanced, that is

$$\begin{aligned} \boldsymbol{F}(t) - \boldsymbol{n}(t, \beta_1 + l_1) &= m_1 \boldsymbol{p}_{tt} \\ \boldsymbol{M}^b(t) - \boldsymbol{m}^b(t, \beta_1 + l_1) &= \boldsymbol{H}_1 \boldsymbol{\omega}_t + \widehat{\boldsymbol{\omega}} \boldsymbol{H}_1 \boldsymbol{\omega}. \end{aligned} \tag{3.34}$$

Orientation as Quaternions

We have used rotation matrices in our model development, but in our numerical implementation, we represent orientation using non-unit quaternions as presented in [92] and paraphrased in (2.5) to avoid degradation of orthogonality in the rotation matrices due to numerical approximation. The orientation at the base is then given by

$$\boldsymbol{h}(t,0) = \begin{bmatrix} \cos\left(\frac{1}{2}\psi_1(t,0)\right) & 0 & 0 & \sin\left(\frac{1}{2}\psi_1(t,0)\right) \end{bmatrix}^\top.$$

Spatial BVP Solution

The fast torsional dynamics of the snap-through motion occur over an interval of about 1ms [87], making it necessary to simulate using time steps on the order of microseconds. Although past work found success using shooting methods to solve concentric tube kinematics

efficiently [15], for our implicit time discretization scheme shooting methods can become ill-conditioned at extremely small time-steps [112]. Instead we use an iterative solution of a finite difference system as described in Section 3.1.3.

Note that the the concentric tube problem involves the coupling of systems (3.1), (3.28), and (3.31) as shown in Figure 3.17. We use a Levenberg-Marquardt algorithm to solve the coupled nonlinear system so that $\|\boldsymbol{E}\|^2 < 10^{-9}$ using standard SI units for all variables. The time difference is implemented using BDF2 [24], which in the context of (3.4) has non-zero coefficients $c_0 = 3/(2\Delta t)$, $c_1 = -2/\Delta t$, and $c_2 = 1/(2\Delta t)$ The Jacobian of the above system is sparse, so we use sparse matrix data structures and sparse linear solving routines implemented in C++ using the matrix library Eigen [37]. The Jacobian is calculated by first order finite differences with appropriately chosen increments for the magnitudes the variables. Note that the accuracy of this Jacobian approximation does not affect the accuracy of the model, only the convergence of the iterative solution.

3.5.3 Simulation and Experimental Verification

Our experimental setup (Figure 3.18) consists of a two-tube robot with the outer tube rigidly attached to the baseplate as shown in Figure 3.19. A Vision Research Phantom® v310 high-speed camera was used to study the robot as it was actuated through an elastic instability, followed by oscillations. The high-speed camera collected data at 50,000FPS ($\Delta t = 20\,\mu s$) with a resolution of 256x128 pixels. Disk-shaped markers were affixed to each tube at its tip, so that the relative angle θ_f between the tubes can be easily reconstructed from video data. The marker features are colored lighter than the other objects in the video. We use MATLAB's "regionprops" function to find the marker centroids, and subsequently calculate the relative tip angle θ_f based on the marker pixel locations. We assume roll and pitch of the tip are negligible so that θ_f is calculated assuming the markers are parallel to the camera plane, which is a reasonable assumption based on the model solution for tip orientation.

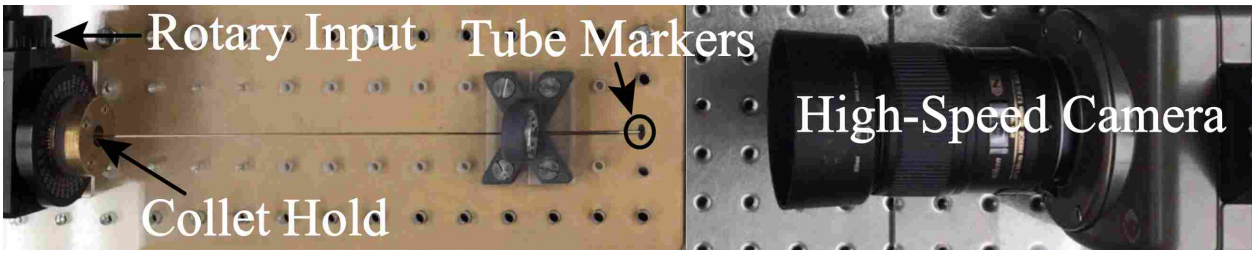


Figure 3.18: The experimental setup is shown. The snapping bifurcation was captured on a high-speed camera. Tube markers were affixed to the tip of each tube allowing the relative angle θ to be visually reconstructed.

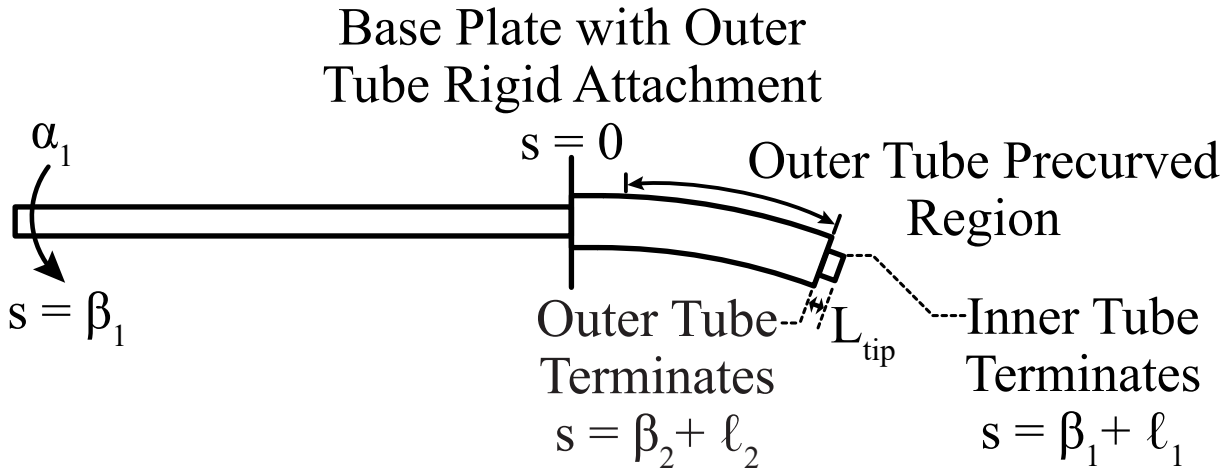


Figure 3.19: The validation was performed using a two-tube robot as in [87]. The outer tube was rigidly attached to the baseplate, and the inner tube was rotated at a distance from the baseplate β_1 which is constant for any given trial, and varied between trials.

Measured and Approximate Parameters

The CTR tubes used in these experiments are shown in Figure 3.20. The inner tube was made from Nitinol and the outer from stainless steel. The tubes have precurved sections of constant curvature. The precurved section of the outer tube extends all the way to its tip, while the inner tube has a short 15.0 mm straight segment at the tip after the precurved section. The precurvature functions were fit from images of the tubes with the results

$$u^*_{1,x}(s) = \begin{cases} 0\text{m}^{-1}, & s < L_1 - 0.044\text{m} \\ 46.9\text{m}^{-1}, & L_1 - 0.044\text{m} \leq s \leq L_1 - 0.015\text{m} \\ 0\text{m}^{-1}, & s > L_1 - 0.015\text{m} \end{cases} \tag{3.35}$$

$$u^*_{2,x}(s) = \begin{cases} 0\text{m}^{-1}, & s < L_2 - 0.0399\text{m} \\ 8.72\text{m}^{-1}, & s \geq L_2 - 0.0399\text{m} \end{cases}$$

where the tube tip locations $L_i = l_i + \beta_i$ have been defined for convenience.

A description of parameters and their measured, calibrated, or known values is given in Table 3.2. Because the inner tube properties can have a significant effect on the dynamic response, the Young's modulus was experimentally calibrated in a separate static cantilever

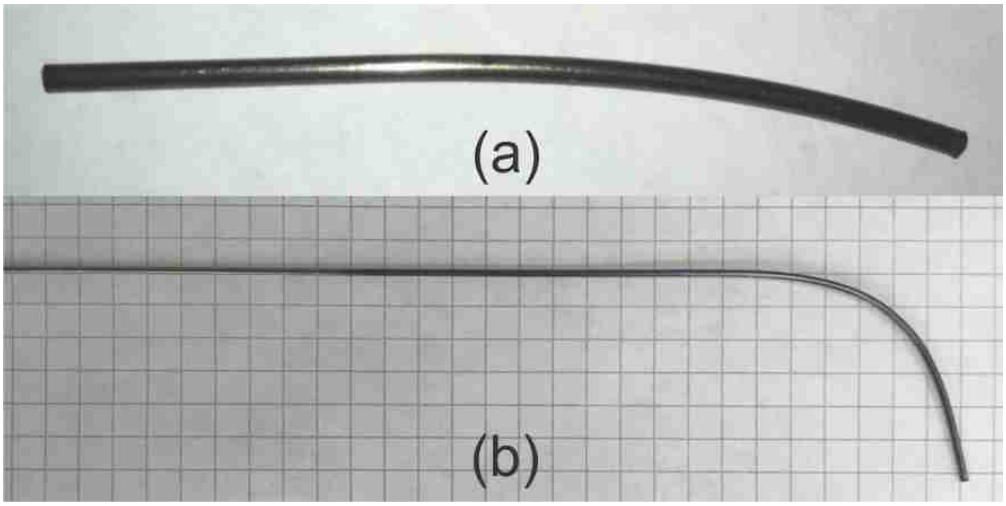


Figure 3.20: Component tubes used for two tube robot in validation study before assembly. (a) outer stainless steel tube. (b) inner Nitinol tube (which is a solid circular rod).

Table 3.2: Concentric Tube Model Parameters

NAME	DESCRIPTION	METHOD	VALUE
$r_{i,1}$	Inner radius	Data Sheet	0m
$r_{o,1}$	Outer radius	Data Sheet	0.000508m
E_1	Young's modulus	Deflection test	81.97GPa
$u^*_{1,x}(s)$	Precurvature	Fit from image	Given in text
ρ_1	Density	Mass/Volume	6493 kg/m^3
L_{tip}	Tip extension	Measured	0.002m
$r_{i,2}$	Inner radius	Data Sheet	0.00062m
$r_{o,2}$	Outer radius	Data Sheet	0.001055m
E_2	Young's modulus	Data Sheet	210GPa
$u^*_{2,x}(s)$	Precurvature	Fit from image	Given in text
ρ_2	Density	Data Sheet	8000 kg/m^3
l_2	Length of outer tube	Measured	0.0531m
m_1	Marker 1 Mass	Measured	0.0278g
$r_{M,1}$	Marker 1 Radius	Measured	0.00273m
$t_{M,1}$	Marker 1 Thickness	Measured	0.00113m
m_2	Marker 2 Mass	Measured	0.0714g
$r_{M,2}$	Marker 2 Radius	Measured	0.00446m
$t_{M,2}$	Marker 2 Thickness	Measured	0.00109m
m_G	Glue mass	Measured	0.02 g
l_1	Exp. 1 inner tube length	Measured	0.1671m
l_2	Exp. 2 inner tube length	Measured	0.2179m
l_3	Exp. 3 inner tube length	Measured	0.2687m
B_*	All mat. damping coeffs	Calibrated	5.91×10^{-8} Nm^{2}s

deflection test. The density was determined by dividing the measured tube mass by the volume calculated from the tube dimensions.

Preliminary simulation results showed that the inertia of the tracking markers attached to the tube tips was not negligible, so we modeled the markers by implementing the rigid body coupling boundary conditions described in our model equations above. The markers were circular disks with holes to fit around the tubes, so that the inertia components are given by

$$H_{i,xy} = m_i(3(r_{M,i}^2 + r_{o,i}^2) + t_{M,i})/12$$
$$H_{i,z} = m_i(r_{M,i}^2 + r_{o,i}^2)/2,$$

where $r_{M,i}$ is the marker outer radius, $r_{o,i}$ is the tube outer radius which is also the marker inner radius, and $t_{M,i}$ is the marker thickness. Super glue was used to attach the markers to the tubes. Weighing a single drop to be 0.02g, we added this amount to each marker mass.

Simulation

The simulation uses $N_1 = 15$ grid points for the dynamics of the inner tube below the baseplate, modeled by (3.31), $N_2 = 60$ grid points for the concentric tube PDEs, modeled by (3.28), and $N_3 = 4$ grid points for the slight extension of the inner tube beyond the tip of the outer tube, modeled by (3.1). The time step is Δt =10 µs. While our overall approach is capable of running stably at large time steps and capturing slower bending dynamics at real-time rates [112], the small time steps required to resolve the detailed torsional elastic instability dynamics.

Building a grid of states for each section based on the solver guess and the boundary conditions is a fairly complicated process. The mapping of elements from the guess vector to the grid point values is shown in Table 3.3, as well as the grid point values which are found by strongly enforcing the boundary conditions. Gray cells represent strongly enforced values from boundary conditions, and white cells indicate the corresponding indices of the guess vector using MATLAB notation. A minus superscript indicates that a value comes from the previous section. The first grid is the section below the base described by (3.31),

Table 3.3: Grid Layout for Three Section Concentric Tube Robot

	Col 1 (Actuator)	Col 2 : Col $N_1 - 1$	Col N_1 (Baseplate)
α_1	Input	$2 : 3N_1 - 5$	$3N_1 - 4$
$\alpha_{1,t}$	Input derivative		$3N_1 - 3$
$m^b_{z,1}$	1		$3N_1 - 2$

	Col 1 (Baseplate)	Col 2 : Col $N_2 - 1$	Col N_2 (Outer tube ends)
$\boldsymbol{p}$	$\mathbf{0}$	$3N_1 + 5$: $3N_1 + 22N_2 - 40$	$3N_1 + 22N_2 - 39 : 3N_1 + 22N_2 - 37$
$\boldsymbol{h}$	$[\cos(\alpha_1/2) \quad 0 \quad 0 \quad \sin(\alpha_1/2)]^T$		$3N_1 + 22N_2 - 36 : 3N_1 + 22N_2 - 33$
$\boldsymbol{q}$	$\mathbf{0}$		$3N_1 + 22N_2 - 32 : 3N_1 + 22N_2 - 30$
$\boldsymbol{\omega}$	$[0 \quad 0 \quad \alpha_{1,t}]^T$		$3N_1 + 22N_2 - 29 : 3N_1 + 22N_2 - 27$
$\boldsymbol{n}$	$3N_1 - 1 : 3N_1 + 1$		$3N_1 + 22N_2 - 26 : 3N_1 + 22N_2 - 24$
$\boldsymbol{m}^b_{xy}$	$3N_1 + 2 : 3N_1 + 3$		$3N_1 + 22N_2 - 23 : 3N_1 + 22N_2 - 22$
$m^b_{z,1}$	$m^{b-}_{z,1}$		$3N_1 + 22N_2 - 21$
$m^b_{z,2}$	$3N_1 + 4$		$-H_{2,A}\omega_{2,z,t}$
θ	$-\alpha_1$		$3N_1 + 22N_2 - 20$
γ	$-\alpha_{1,t}$		$3N_1 + 22N_2 - 19$

	Col 1 (Outer tube ends)	Col 2 : Col $N_3 - 1$	Col N_3 (Inner tube ends)
$\boldsymbol{p}$	$\boldsymbol{p}^-$	$3N_1 + 22N_2 - 18$: $3N_1 + 22N_2 + 19N_3 - 57$	$3N_1 + 22N_2 + 19N_3 - 56 : 3N_1 + 22N_2 + 19N_3 - 54$
$\boldsymbol{h}$	$\boldsymbol{h}^-$		$3N_1 + 22N_2 + 19N_3 - 53 : 3N_1 + 22N_2 + 19N_3 - 50$
$\boldsymbol{q}$	$\boldsymbol{q}^-$		$3N_1 + 22N_2 + 19N_3 - 49 : 3N_1 + 22N_2 + 19N_3 - 47$
$\boldsymbol{\omega}$	$\boldsymbol{\omega}^-$		$3N_1 + 22N_2 + 19N_3 - 46 : 3N_1 + 22N_2 + 19N_3 - 44$
$\boldsymbol{n}$	$\boldsymbol{n}^- + m_2(\boldsymbol{R}(\boldsymbol{q}_t + \hat{\boldsymbol{\omega}}\boldsymbol{q}) - \boldsymbol{g})$		$m_1(\boldsymbol{g} - \boldsymbol{R}(\boldsymbol{q}_t + \hat{\boldsymbol{\omega}}\boldsymbol{q}))$
$\boldsymbol{m}^b$	$\begin{bmatrix} \boldsymbol{m}^{b-}_{xy} + \left(\boldsymbol{H}_2\bar{\boldsymbol{\omega}}_{2,t} + \hat{\bar{\boldsymbol{\omega}}}_2\boldsymbol{H}_2\bar{\boldsymbol{\omega}}_2\right)_{xy} \\ m^{b-}_{z,1} \end{bmatrix}$		$-\boldsymbol{H}_1\boldsymbol{\omega}_t - \hat{\boldsymbol{\omega}}\boldsymbol{H}_1\boldsymbol{\omega}$

the second grid is the concentric tube section described by (3.28), and the third grid is the extension of the inner tube beyond the outer tube as described by (3.1). The total number of guessed variables is $3(N_1 - 1) + 22(N_2 - 1) + 19(N_3 - 1)$, which matches the number of internal residual terms from the midpoint method.

Experimental Protocol

Experimental validation of this simulation consisted of actuating a concentric-tube robot through an elastic instability transition for three different inner tube transmission lengths β_1 from the base plate to the inner tube's actuator. Increasing this transmission length also increases the potential of the robot to store elastic energy, resulting in a more forceful bifurcation [87]. The total inner tube lengths for the three experimental configurations can be viewed in Table 3.2. In each configuration, the inner tube tip was extended 2mm out of the outer tube. The rotary dial was slowly rotated by hand to induce an elastic instability and video was collected at 50,000 frames per second. To verify repeatability of the data with this

procedure, we conducted 4 trials for each transmission length configuration. A comparison of every outcome is overlaid in Figure 3.21, illustrating that the dynamics introduced by this procedure are highly repeatable.

Calibration and Results

We first verify the static robot parameters by comparing the experimental snap angles to the well-known CTR "S-curve" which uses the static model solution to express the relationship between the relative base angle θ_b and the tip angle θ_f between the tubes [27]. The S-curves for the three experimental cases are plotted and compared with the experimental steady-state experimental behavior as shown in Figure 3.22.

While the majority of the model parameters were either read from datasheets, measured directly, or determined from a separate experiment, the material damping parameters (especially those associated with torsion) are more difficult to directly measure, and we wish to investigate whether frictional energy dissipation could potentially be accounted for by

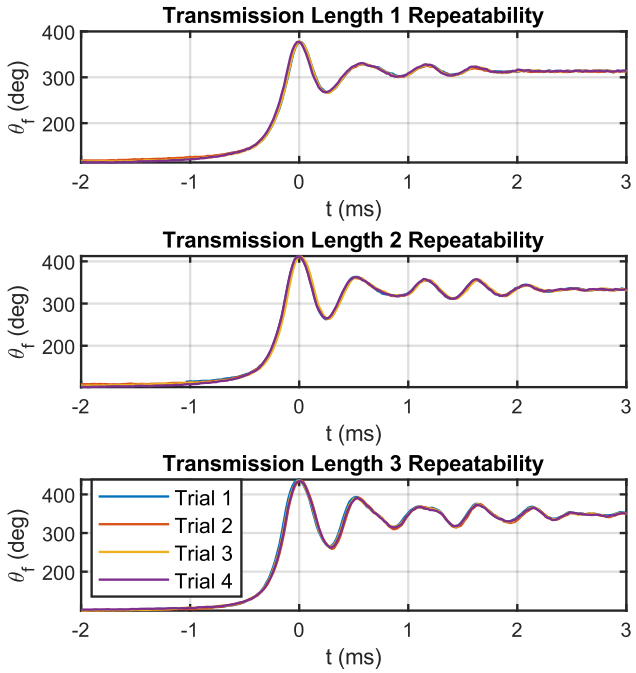


Figure 3.21: Each transmission length was tested in four trials to ensure that the robot motion was repeatable. Any differences between trials are minor and consistent behavior is observed.

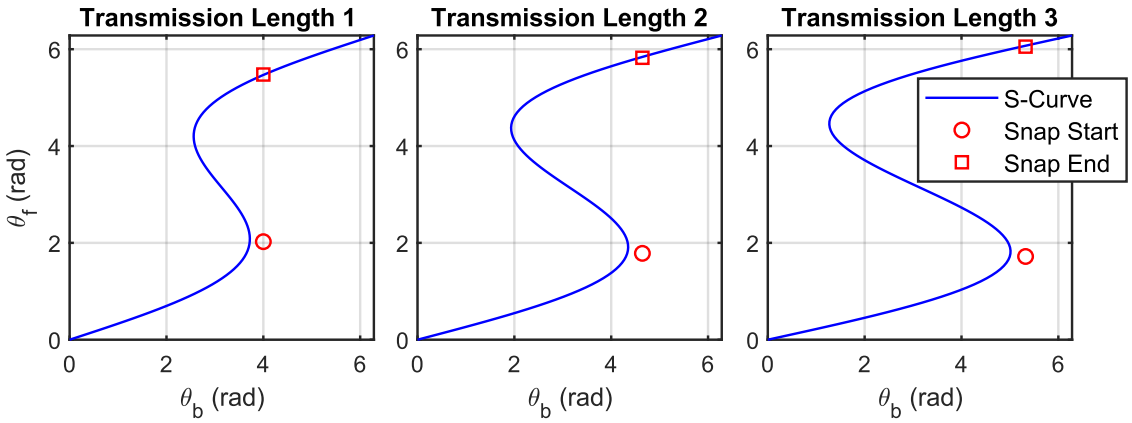


Figure 3.22: The base actuation angle was recorded for each trial, and the steady-state tip angles before and after snapping are extracted from the data. Simulated "S-curves" are compared with the experimental behavior. The S-curves pass to the left of the observed initial snapping angles, which is likely due to friction.

lumping its effects into the material damping term. Therefore, using the three experimental datasets, we calibrated a single material damping parameter used for $B_{B,1}$, $B_{T,1}$, $B_{B,2}$, and $B_{T,2}$ by a least squares fit of peak heights. The calibrated damping parameter is listed in Table 3.2, and the resulting model solution for θ_f is compared to the experimental datasets in Figure 3.23. Note that we synchronized camera time with simulation time by setting the time datum $t = 0$ at the location of the first peak. The results show that the dynamic model predicts the main features of the experimental dataset reasonably well, especially the rise curve when the tube is transitioning through the snap. The overshoot behavior, period of vibration, and subsequent peak heights are also captured well, although an unknown effect around 1 ms creates a phase shift in the experimental data that persists as the oscillations decay. A render of the simulation is shown in Figure 3.24.

The model is able to capture dynamic transition behavior during an elastic instability and the associated release of stored elastic energy. The fading oscillations after the instability are not perfectly described by the model, and future work could attempt to improve model accuracy by incorporating additional terms such as clearance between tubes and dynamic or static friction terms. Still, the current numerical model solution provides an excellent prediction of the rapid tip angle transition at elastic instability, the overshoot, and the initial oscillation decay.

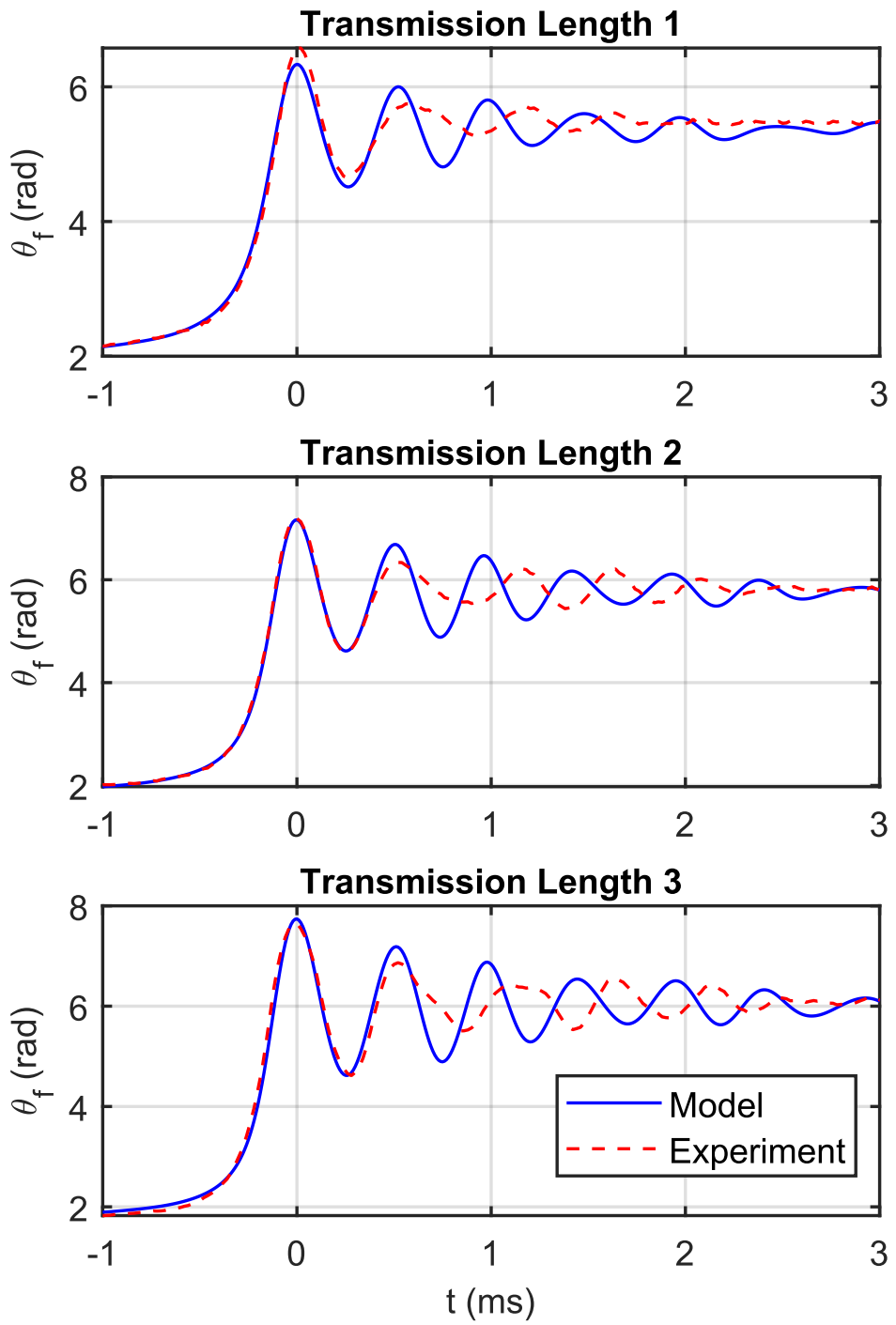


Figure 3.23: The model-predicted tip angle is compared with the experimentally observed tip angle using a calibrated damping constant of $5.91 \times 10^{-8}\text{Nm}^2/\text{s}$.

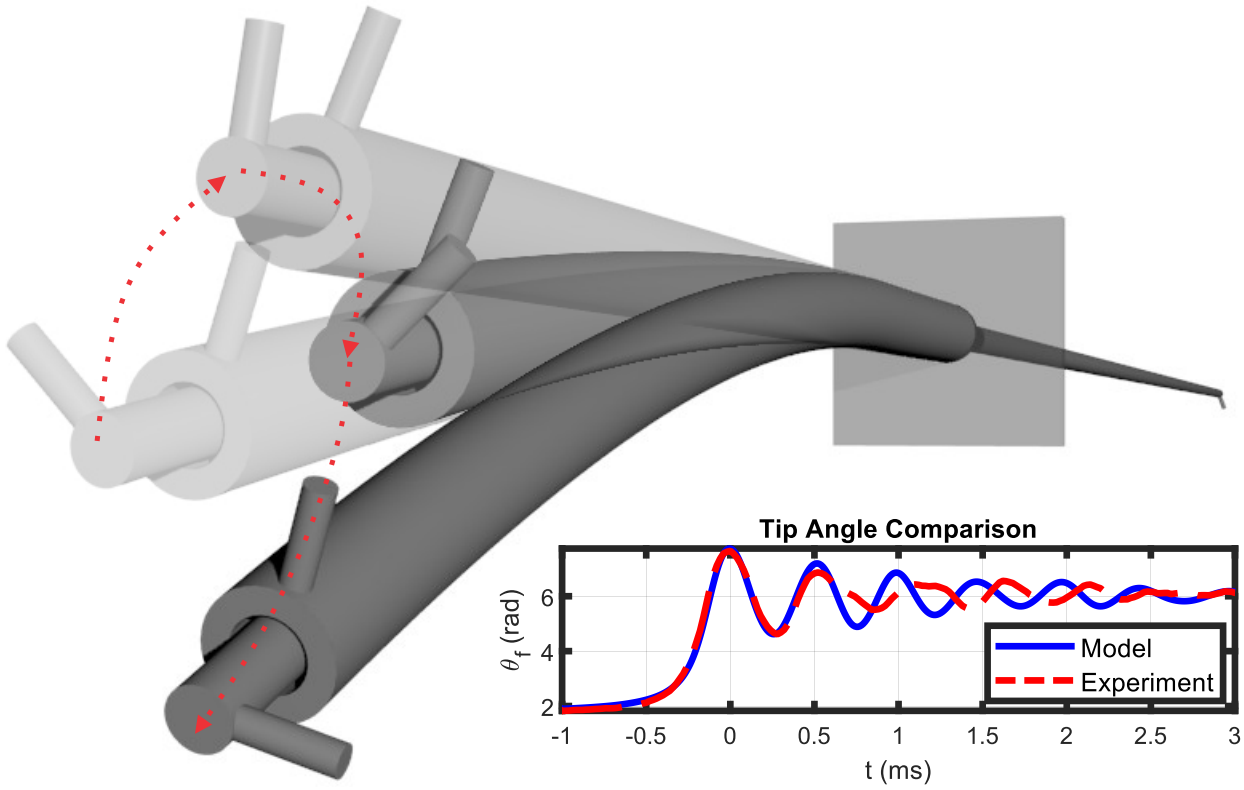


Figure 3.24: A rendering of our model solution showing the concentric tube robot undergoing an elastic instability.

3.6 Discussion

We have presented a numerical framework for solving Cosserat-rod based dynamic models of soft and continuum robots. The stability provided by the implicit time discretization often enables one to solve robot dynamics problems at real-time rates. Experimental trials demonstrated the accuracy of the proposed model. The framework is adaptable to various designs of continuum robots with different operating principles as shown by the examples considered here. We anticipate that our approach can be widely applied across the spectrum of continuum robot designs.

Chapter 4

Stability

The ODE systems derived in Chapter 2 satisfy the first-order conditions necessary for static equilibrium but do not provide any information about the elastic stability of the solution. This chapter includes results from the author's journal paper [113] to adapt results from optimal control to determine the stability of Kirchhoff rods and Cosserat rods subject to general terminal constraints, including the multi-rod coupled models that describe parallel continuum robots. We formulate a sufficient condition for the stability of a solution, a numerical test for evaluating this condition, and a heuristic stability metric. It is verified that the numerical stability test agrees with the classical results for the buckling of single columns with various terminal constraints and for multi-column frames. We then validate our approach experimentally on a six degree-of-freedom parallel continuum robot.

We begin by considering simple problems and gradually increase in complexity. In Section 4.1, we describe our approach in the context of a classical optimal control problem for a control $\boldsymbol{u}(s) \in \mathbb{R}^m$ and a state $\boldsymbol{x}(s) \in \mathbb{R}^n$. Section 4.2 applies this framework to derive the first- and second-order conditions for a stable planar rod subject to various end-point constraints. Section 4.3 considers the joining of multiple planar rods to create a planar parallel continuum robot, which is validated in several special cases by comparison to the classical Euler buckling formulas for columns and multi-column sway frames. In Section 4.4, we derive the stability conditions for a single spatial rod by considering the optimal control framework for a problem with a control $\boldsymbol{u}(s) \in \mathbb{R}^m$ and a state $\boldsymbol{x}(s) \in \mathrm{SE}(3)$. Section 4.5 extends this result to coupled spatial rods (i.e. parallel continuum robots), and Section 4.6

validates the resulting stability test experimentally using a prototype parallel continuum robot, while providing insight into issues involved in the application to stability detection and avoidance.

4.1 Bolza Problem

4.1.1 General Problem Statement

Planar rod problems fit the form of a fixed-time, Bolza problem in optimal control. The Bolza problem statement is

$$\begin{aligned} \underset{\boldsymbol{u}}{\text{minimize}} \quad & J(\boldsymbol{u}) = \phi(\boldsymbol{x}_L) + \int_0^L \mathcal{L}(s, \boldsymbol{x}(s), \boldsymbol{u}(s)) ds \\ \text{subject to} \quad & \boldsymbol{x}_s = \boldsymbol{f}(s, \boldsymbol{x}(s), \boldsymbol{u}(s)) \\ & \boldsymbol{x}(0) = \boldsymbol{x}_0, \quad \boldsymbol{\beta}(\boldsymbol{x}_L) = \boldsymbol{0}, \end{aligned} \tag{4.1}$$

where $\boldsymbol{x}(s) \in \mathbb{R}^n$ is the state vector, $\boldsymbol{u}(s) \in \mathbb{R}^m$ is the "control", $\boldsymbol{\beta} : \mathbb{R}^n \to \mathbb{R}^p$ are general terminal constraints, $\phi : \mathbb{R}^n \to \mathbb{R}$ is the terminal cost, $\mathcal{L} : \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ is the Lagrangian (the cost-density), and $\boldsymbol{f} : \mathbb{R} \times \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ is the state derivative.

4.1.2 First-Order Necessary Conditions

The derivation of minimal conditions is kept brief as it follows that of Hull [49], but we take the additional step of eliminating the terminal Lagrange multipliers from the problem and formulating a reduced set of terminal boundary conditions, and we also use the opposite sign for Lagrange multipliers in the Hamiltonian as this eventually yields a cleaner result for the rod problems. We form an augmented cost function J' with Lagrange multiplier vectors $\boldsymbol{\lambda}(s) \in \mathbb{R}^n$ and $\boldsymbol{\nu} \in \mathbb{R}^p$ to enforce the differential and terminal constraints:

$$J'(\boldsymbol{u}, \boldsymbol{\lambda}, \boldsymbol{\nu}) = G + \int_0^L \left(H + \boldsymbol{\lambda}^\top \boldsymbol{x}_s \right) ds,$$

where H is the Hamiltonian

$$H(s, \boldsymbol{x}, \boldsymbol{u}, \boldsymbol{\lambda}) := \mathcal{L}(s, \boldsymbol{x}, \boldsymbol{u}) - \boldsymbol{\lambda}^\top \boldsymbol{f}(s, \boldsymbol{x}, \boldsymbol{u})$$

and G is the augmented terminal cost function

$$G(\boldsymbol{x}_L, \boldsymbol{\nu}) := \phi(\boldsymbol{x}_L) + \boldsymbol{\nu}^\top \boldsymbol{\beta}(\boldsymbol{x}_L).$$

Taking the first variation of J', canceling terms where appropriate, and integrating by parts so that $\int_0^L \boldsymbol{\lambda}^\top \delta \boldsymbol{x}_s ds = \boldsymbol{\lambda}_L^\top \delta \boldsymbol{x}_L - \int_0^L \boldsymbol{\lambda}_s^\top \delta \boldsymbol{x} ds$ yields

$$dJ' = G_{\boldsymbol{x}_L} \delta \boldsymbol{x}_L + \boldsymbol{\lambda}_L^\top \delta \boldsymbol{x}_L + \int_0^L \left[\left(H_{\boldsymbol{x}} - \boldsymbol{\lambda}_s^\top \right) \delta \boldsymbol{x} + H_{\boldsymbol{u}} \delta \boldsymbol{u} \right] ds.$$

Choosing Lagrange multipliers $\boldsymbol{\lambda}_s = H_{\boldsymbol{x}}^\top$ and $\boldsymbol{\lambda}_L = -G_{\boldsymbol{x}_L}^\top$ is convenient because the first differential reduces to $dJ' = \int_0^L H_{\boldsymbol{u}} \delta \boldsymbol{u} ds$. As demonstrated in Chapter 9.3.2 in Hull [49], it can be shown that $H_{\boldsymbol{u}} = \boldsymbol{0}$ is a necessary condition for the first differential to vanish.

We may avoid solving for $\boldsymbol{\nu}$ by premultiplying both sides of $\boldsymbol{\lambda}_L = -G_{\boldsymbol{x}_L}^\top$ by a matrix $\boldsymbol{P}^\top$, where the columns of $\boldsymbol{P}$ form an orthonormal basis for the nullspace of $\boldsymbol{\beta}_{\boldsymbol{x}_L} \in \mathbb{R}^{p \times n}$. $\boldsymbol{P}$ can be calculated from a singular value decomposition $\boldsymbol{\beta}_{\boldsymbol{x}_L} = \boldsymbol{U} \boldsymbol{\Sigma} \boldsymbol{V}^\top$ by selecting the $n-p$ columns of $\boldsymbol{V}$ that correspond to the $n-p$ singular values that equal zero. MATLAB's `null()` function conveniently obtains $\boldsymbol{P}$ as $\boldsymbol{P} = \texttt{null}(\boldsymbol{\beta}_{\boldsymbol{x}_L})$. In the problems here $\boldsymbol{\beta}(\boldsymbol{x}_L)$ is linear in $\boldsymbol{x}_L$ so that $\boldsymbol{P}$ is constant. Thus we obtain a reduced set of $n-p$ co-state boundary conditions

$$\boldsymbol{P}^\top \boldsymbol{\lambda}_L = -\boldsymbol{P}^\top \left(\phi_{\boldsymbol{x}_L} + \boldsymbol{\nu}^\top \boldsymbol{\beta}_{\boldsymbol{x}_L} \right)^\top = -\boldsymbol{P}^\top \phi_{\boldsymbol{x}_L}^\top.$$

These are the so-called "natural" boundary conditions, which are often equivalent to equations that could be obtained from a Newtonian approach. We combine the state and

co-state constraints into a single terminal constraint function

$$\boldsymbol{E}(\boldsymbol{x}_L, \boldsymbol{\lambda}_L) = \begin{bmatrix} \boldsymbol{\beta}(\boldsymbol{x}_L) \\ \left(\boldsymbol{P}^\top \boldsymbol{\lambda}_L + \boldsymbol{P}^\top \phi_{\boldsymbol{x}_L}^\top\right) \end{bmatrix} = \boldsymbol{0}, \tag{4.2}$$

where $\boldsymbol{E} : \mathbb{R}^n \times \mathbb{R}^n \to \mathbb{R}^n$.

Summarizing the development, our use of Lagrange multipliers has led us to the first-order necessary conditions:

$$\begin{aligned} \boldsymbol{x}_s &= \boldsymbol{f} \\ \boldsymbol{\lambda}_s &= \boldsymbol{H}_{\boldsymbol{x}}^\top \\ H_{\boldsymbol{u}} &= \boldsymbol{0} \\ \boldsymbol{E} &= \boldsymbol{0}. \end{aligned} \tag{4.3}$$

Thus there is a BVP which may be solved to find a candidate control trajectory. However, further analysis is necessary to determine if a control satisfying these conditions minimizes the cost function.

4.1.3 Second-Order Sufficient Conditions

As detailed in [50, 49, 82], whether an optimal solution is a local minimum is partially determined by the presence of admissible comparison paths. The existence of another optimal path which satisfies the constraints is equivalent to the existence of a so-called conjugate point $s_{cp} \in [0\ L)$. The nonexistence of conjugate points is the classical Jacobi condition, which is a known sufficient condition for weak local optimality if the first-order necessary conditions and the strong Legendre-Clebsch condition ($H_{\boldsymbol{uu}}$ positive definite) are already satisfied. The existence of a conjugate point in $s_{cp} \in (0\ L)$ is sufficient to conclude that the optimal path is not a minimum, and $s_{cp} \notin [0\ \ L)$ is sufficient to conclude that the optimal path is a minimum [49] (given the first-order and strong Legendre-Clebsch conditions are met). In the case of $s_{cp} = 0$, there exists an admissible comparison path for which the second differential vanishes, so the third and fourth differentials would need to be considered to investigate stability. For our application, we make the assumption that a conjugate point $s_{cp} = 0$ is unsafe.

The existence of conjugate points can be determined by examining the "sweep method" matrix $\bar{\boldsymbol{S}}$, which is defined by

$$\delta\boldsymbol{\lambda}(s) = \bar{\boldsymbol{S}}(s)\delta\boldsymbol{x}(s),$$

as given in [49], where $\delta\boldsymbol{\lambda}(s)$ is the associated small change in $\boldsymbol{\lambda}$ at s that would be required in order to continue satisfying the terminal boundary conditions. s_{cp} is a conjugate point if and only if the matrix $\bar{\boldsymbol{S}}(s)$ becomes infinite at $s = s_{cp}$.

We can formulate $\bar{\boldsymbol{S}}$ by recognizing that $\boldsymbol{E}(\boldsymbol{x}_L, \boldsymbol{\lambda}_L)$ is implicitly a function of $\boldsymbol{x}(s)$ and $\boldsymbol{\lambda}(s)$ through its arguments, that is

$$\boldsymbol{E}(\boldsymbol{x}_L, \boldsymbol{\lambda}_L) = \boldsymbol{E}\left(\boldsymbol{x}_0 + \int_0^L \boldsymbol{x}_s ds, \boldsymbol{\lambda}_0 + \int_0^L \boldsymbol{\lambda}_s ds\right).$$

Thus we may take the variation of $\boldsymbol{E}$:

$$\delta\boldsymbol{E} = \boldsymbol{E}_{\boldsymbol{x}(s)}\delta\boldsymbol{x}(s) + \boldsymbol{E}_{\boldsymbol{\lambda}(s)}\delta\boldsymbol{\lambda}(s) = \boldsymbol{0}.$$

Recalling that $\boldsymbol{E} \in \mathbb{R}^n$ and $\boldsymbol{\lambda} \in \mathbb{R}^n$, the above can be solved for

$$\delta\boldsymbol{\lambda} = -\left[\boldsymbol{E}_{\boldsymbol{\lambda}(s)}\right]^{-1}\boldsymbol{E}_{\boldsymbol{x}(s)}\delta\boldsymbol{x}(s).$$

This shows that

$$\bar{\boldsymbol{S}}(s) = -\left[\boldsymbol{E}_{\boldsymbol{\lambda}(s)}\right]^{-1}\boldsymbol{E}_{\boldsymbol{x}(s)}.$$

Assuming that $\boldsymbol{E}_{\boldsymbol{x}(s)}$ is finite (which is true under the mild and verifiable assumption that $\boldsymbol{E}_{\boldsymbol{x}_L}$ and $\boldsymbol{E}_{\boldsymbol{\lambda}_L}$ are finite), then s_{cp} is a conjugate point only if the matrix $\boldsymbol{E}_{\boldsymbol{\lambda}(s_{cp})}$ is singular.

4.1.4 Numerical Test

To test for conjugate points on $[0\ L)$, $\boldsymbol{E}_{\boldsymbol{\lambda}(s)}$ can be calculated by first obtaining the transition matrix $\boldsymbol{\Phi}(s)$ which maps small changes in $\boldsymbol{x}_L$ and $\boldsymbol{\lambda}_L$ to small changes in $\boldsymbol{x}(s)$ and $\boldsymbol{\lambda}(s)$:

$$\boldsymbol{\Phi}(s) := \begin{bmatrix} \partial \boldsymbol{x}(s)/\partial \boldsymbol{x}_L & \partial \boldsymbol{x}(s)/\partial \boldsymbol{\lambda}_L \\ \partial \boldsymbol{\lambda}(s)/\partial \boldsymbol{x}_L & \partial \boldsymbol{\lambda}(s)/\partial \boldsymbol{\lambda}_L \end{bmatrix}.$$

Then, $\boldsymbol{E}_{\boldsymbol{\lambda}(s)}$ can be expressed as

$$\boldsymbol{E}_{\boldsymbol{\lambda}(s)} = \frac{\partial \boldsymbol{E}}{\partial \boldsymbol{x}_L}\frac{\partial \boldsymbol{x}_L}{\partial \boldsymbol{\lambda}(s)} + \frac{\partial \boldsymbol{E}}{\partial \boldsymbol{\lambda}_L}\frac{\partial \boldsymbol{\lambda}_L}{\partial \boldsymbol{\lambda}(s)} = \frac{\partial \boldsymbol{E}}{\partial \boldsymbol{x}_L}\left(\boldsymbol{\Phi}^{-1}\right)_{12} + \frac{\partial \boldsymbol{E}}{\partial \boldsymbol{\lambda}_L}\left(\boldsymbol{\Phi}^{-1}\right)_{22}, \tag{4.4}$$

where $\left(\boldsymbol{\Phi}^{-1}\right)_{12}$ and $\left(\boldsymbol{\Phi}^{-1}\right)_{22}$ denote the upper right and lower right $n \times n$ blocks of $\boldsymbol{\Phi}^{-1}(s)$, respectively. Note that $\boldsymbol{\Phi}(s) \in \mathbb{R}^{2n\times 2n}$ is always invertible because it is the transition matrix for a linearized system of differential equations. We note that $\boldsymbol{\Phi}(s)$ can be obtained by differentiating the model equations to obtain the following differential equations for $\boldsymbol{\Phi}$.

$$\boldsymbol{\Phi}_s = \begin{bmatrix} \dfrac{\partial^2 \boldsymbol{x}}{\partial s \partial \boldsymbol{x}_L} & \dfrac{\partial^2 \boldsymbol{x}}{\partial s \partial \boldsymbol{\lambda}_L} \\ \dfrac{\partial^2 \boldsymbol{\lambda}}{\partial s \partial \boldsymbol{x}_L} & \dfrac{\partial^2 \boldsymbol{\lambda}}{\partial s \partial \boldsymbol{\lambda}_L} \end{bmatrix} \boldsymbol{\Phi}.$$

These can be integrated backward from $\boldsymbol{\Phi}(L) = \boldsymbol{I}$. The use of the transition matrix in continuum robotics is explored in more detail in [95].

4.1.5 Heuristic Metric

Note that $\boldsymbol{E}_{\boldsymbol{\lambda}(L)}$ is singular if any terminal state constraints $\boldsymbol{\beta}(\boldsymbol{x}_L)$ exist, but the sufficient conditions for optimality only require nonsingularity of $\boldsymbol{E}_{\boldsymbol{\lambda}(s)}$ for all $s \in [0\ L)$. Also, the values of various elements in $\boldsymbol{E}_{\boldsymbol{\lambda}(s)}$ may depend on choices of problem units. For these reasons, metrics for the closeness of $\boldsymbol{b}_{\boldsymbol{\lambda}(s)}$ to singularity over $[0\ L)$ (such as minimum determinant or condition number) cannot meaningfully indicate closeness to instability in general. Instead, we suggest a potentially useful heuristic based on integration length. If the path is determined to be optimal (no conjugate points on $[0\ L)$), but there exists a conjugate point $s_{cp} < 0$

(determined by checking $\det\left(\boldsymbol{E}_{\boldsymbol{\lambda}(s)}\right)$ for $s < 0$, which is accomplished by continuing to integrate the solution backwards past 0), the distance $d = -t_{cp}$ can be regarded as a heuristic metric for the relative distance to non-optimality: d is the amount that the integration length would need to be increased in order for a conjugate point to appear on the interval assuming all other conditions in the problem remain constant. However, this heuristic should still be used with caution as the sensitivity of conjugate point location to small changes in other problem parameters (other than arc length) could be high, and conjugate point locations may not be continuous in all problem parameters. The sensitivity issue is explored further in simulation in Section 4.6.

4.2 Single Planar Rod

Kirchoff Rod Conditions

Many elastostatic mechanics problems can be naturally cast as optimal control problems via the principle of minimal total potential energy. The main restriction to doing so is that the external loading mechanism must be conservative (i.e. path independent, able to be written as the gradient of some global potential function). In this context, a configuration of an elastostatic system is considered stable if and only if it is a local minimizer of total potential energy. Consider a single planar rod subject to various possible boundary conditions and loadings as shown in Figure 4.1. The rod state vector can be defined as

$$\boldsymbol{x}(s) = \begin{bmatrix} p_x(s) & p_y(s) & \theta(s) \end{bmatrix}^\top,$$

where $p_x(s)$, $p_y(s)$, and $\theta(s)$ are scalar functions that describe the planar position and tangent angle of the rod along the length as depicted in Figure 4.1. The state vector derivative for a Kirchhoff rod (no shear or extension effects) is

$$\boldsymbol{x}_s = \boldsymbol{f}(s, \boldsymbol{x}, u) = \begin{bmatrix} \cos\theta & \sin\theta & u \end{bmatrix}^\top,$$

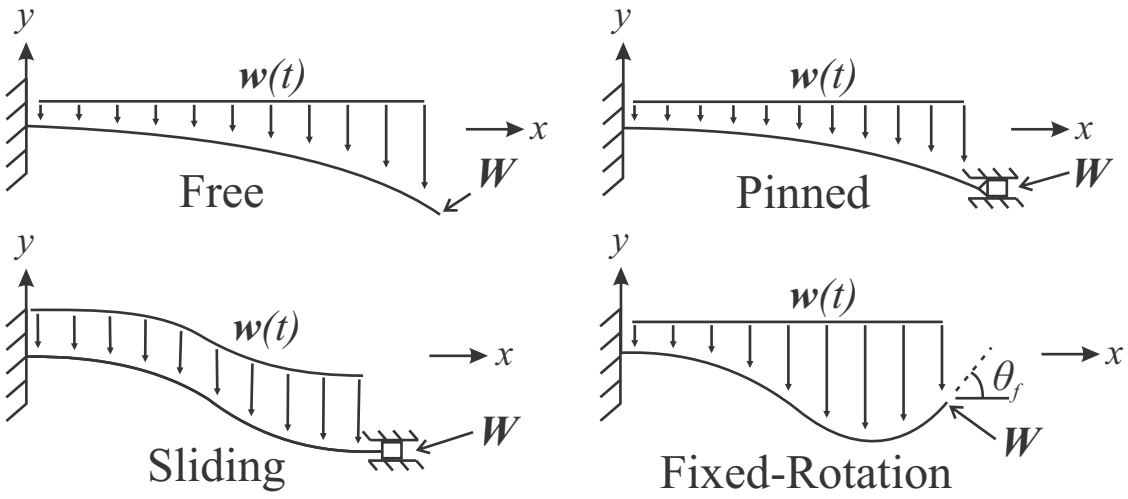


Figure 4.1: Various end constraints are illustrated. In the special case where the rod is initially straight and there is only a force in the x-direction, the buckling loads found with optimal control can be compared to the Euler critical buckling loads.

where $u(s)$ is the rod curvature and corresponds to the "control" in the optimal control framework. We assume there is a general applied wrench, $\boldsymbol{W} = \begin{bmatrix} F_x & F_y & M_z \end{bmatrix}^\top$, acting at $s = L$ and a distributed wrench $\boldsymbol{w}(s) = \begin{bmatrix} f_x(s) & f_y(s) & l_z(s) \end{bmatrix}^\top$ along the length such that the potential energy of the applied loads is given by

$$V = -\boldsymbol{W}^\top \boldsymbol{x}_L - \int_0^L \boldsymbol{w}(s)^\top \boldsymbol{x}(s) ds,$$

where $\boldsymbol{w}(s)$ has here been defined as a known function of s.

As shown in Figure 4.1, we also consider terminal constraints on the rod state of the general form $\boldsymbol{\beta}(\boldsymbol{x}_L) = \boldsymbol{0}$. Assuming a linear constitutive material equation leads to a strain energy density of the form $U(s) = \frac{1}{2} EIu^2$. This means that the total potential energy is given by

$$J = \phi(\boldsymbol{x}_L) + \int_0^L \left(\frac{1}{2} EIu^2 - \boldsymbol{w}^\top \boldsymbol{x} \right) ds,$$

where $\phi(\boldsymbol{x}_L) = -\boldsymbol{W}^\top \boldsymbol{x}_L$ is the terminal cost (energy). Thus, our mechanics problem takes the form of a Bolza problem with a Hamiltonian

$$H = \frac{1}{2} EIu^2 - \boldsymbol{w}^\top \boldsymbol{x} - \boldsymbol{\lambda}^\top \begin{bmatrix} \cos\theta & \sin\theta & u \end{bmatrix}.$$

Applying the constraint $H_u = EIu - \lambda_3 = 0$ leads to $u = \lambda_3/EI$. The evolution of the co-state $\boldsymbol{\lambda}_s = H_{\boldsymbol{x}}^\top$ is given by

$$\boldsymbol{\lambda}_s = -\boldsymbol{w} + (\lambda_1 \sin\theta - \lambda_2 \cos\theta)\, \boldsymbol{e}_3.$$

The reduced set of terminal boundary conditions $\boldsymbol{E}(\boldsymbol{x}_L, \boldsymbol{\lambda}_L)$ are obtained as described by (4.2) and are given for the various cases in Table 4.1. The physical meaning of $\boldsymbol{\lambda}$ can be seen by examining the standard governing equations of Kirchhoff rod theory [2]:

$$\begin{aligned} (\boldsymbol{n}^b)_s &= \begin{bmatrix} -f_x^b & -f_y^b \end{bmatrix}^\top \\ (m_z^b)_s &= -l_z^b + n_x^b \sin\theta - n_y^b \cos\theta, \end{aligned}$$

which reveals that $\lambda_{1-2} \equiv \boldsymbol{n}_{xy}^b$ and $\lambda_3 \equiv m_z^b$. Note that $H_{uu} = EI$, so the strong Legendre-Clebsch condition is satisfied over the whole interval. This is also the case for the more complex problems we consider in the following sections.

In the specific case of an initially straight Kirchhoff rod with a uniform cross-section and homogenous material properties, the stability when subjected to an axially applied force can

Table 4.1: Terminal Boundary Conditions $\boldsymbol{E}(\boldsymbol{x}_L, \boldsymbol{\lambda}_L) = \boldsymbol{0}$

Free	Pinned	Sliding	Fixed-Rotation
$\boldsymbol{\lambda}_L + \boldsymbol{W}$	$\begin{bmatrix} p_y(L) - p_{y_L} \\ \lambda_{1_L} + F_x \\ \lambda_{3_L} + M_z \end{bmatrix}$	$\begin{bmatrix} p_y(L) - p_{y_L} \\ \theta(L) - \theta_L \\ \lambda_{1_L} + F_x \end{bmatrix}$	$\begin{bmatrix} \theta(L) - \theta_L \\ \lambda_{1_L} + F_x \\ \lambda_{2_L} + F_y \end{bmatrix}$

be predicted using the classical Euler buckling formula

$$F_{crit} = \frac{EI\pi^2}{(KL)^2}, \tag{4.5}$$

where K is a length factor corresponding to certain types of end constraints.

We considered the four specific cases listed in Table 4.1 and shown in Figure 4.1, with $p_{y_L} = 0$ in the pinned case, $p_{y_L} = \theta_L = 0$ in the fixed case, and $\theta_L = 0$ in the fixed-rotation case. E, I, and L were arbitrarily set to unity. $\mathbf{\Phi}$ was numerically integrated and $\det(\boldsymbol{E}_{\boldsymbol{\lambda}(s)})$ was checked for a change in signs indicating singularity over n points on the interval $[0\ L)$. We used a bisection method to iteratively converge on the minimum load for which our sufficient stability test fails. We then verified that these numerically predicted loads agree with the known Euler critical loads for straight columns.

We also illustrate this agreement in Figure 4.2, where we plot $\det(\boldsymbol{E}_{\boldsymbol{\lambda}(s)})$ versus arc length s for the same four cases with $L = 1$ and $F_x = -\pi^2$. This is the critical load given by (4.5) for the fixed-rotation case, which has a known length factor of $K = 1$. The plot illustrates that our approach agrees with the Euler theory as evidenced by the purple line intercepting (0,0). In the free-end, pinned, and fixed cases, the Euler length factors are $K = 2$, $K \approx .699$, and $K = 0.5$, respectively, which all agree with the locations of the zero crossings for $\det(\boldsymbol{E}_{\boldsymbol{\lambda}(s)})$ in those cases, i.e. $K = \frac{1}{1-s_{cp}}$ in each case. In this special circumstance, the stability heuristic exactly corresponds to the length of a column for which the applied force is the critical buckling load.

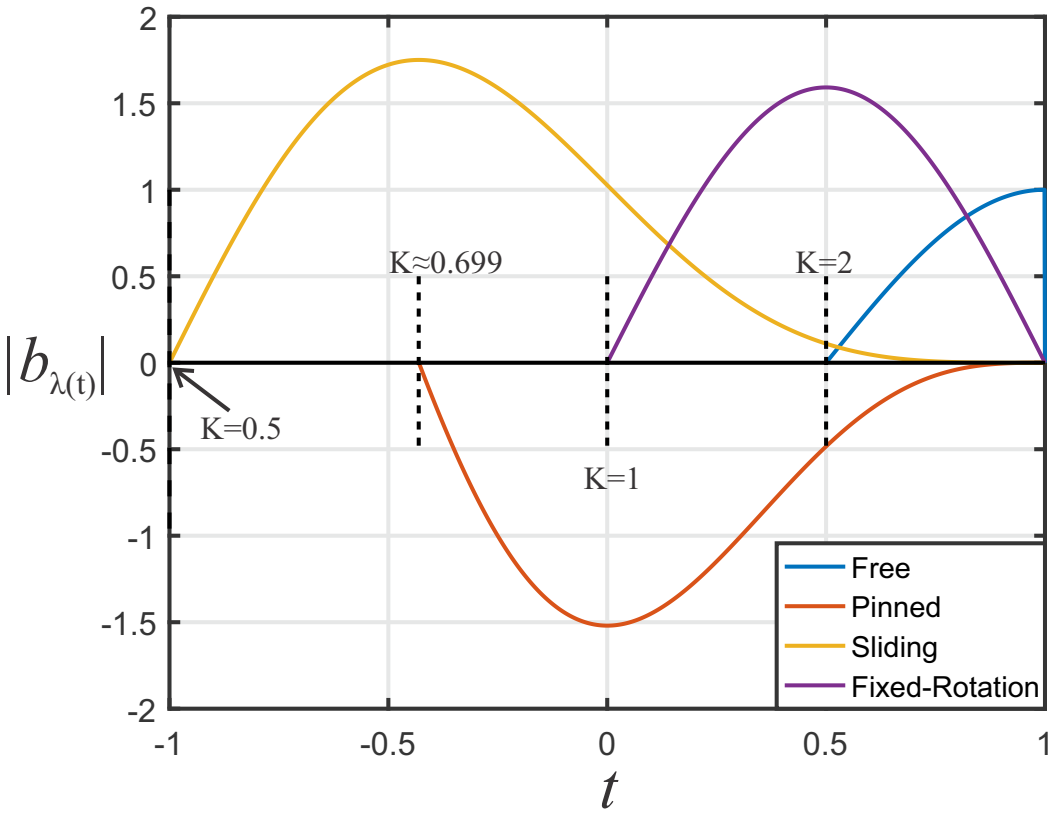


Figure 4.2: The plot shows $\det(\boldsymbol{E}_{\boldsymbol{\lambda}(s)})$ vs. arc length s for a straight rod under the four boundary condition sets with $F_x = -\pi^2$. The conjugate points are shown as the zero crossings of $\det(\boldsymbol{E}_{\boldsymbol{\lambda}(s)})$ in each case. The corresponding equivalent Euler length factor K is shown to be in agreement with the location of each conjugate point. Note that $\det(\boldsymbol{E}_{\boldsymbol{\lambda}(s)})$ contains various combinations of mixed units across the different cases, so its value is not physically meaningful. It has also been scaled for better visibility of the plots, but this does not affect the location of conjugate points.

Cosserat Rod Conditions

The planar Kirchhoff rod model presented above (also known as the planar "elastica") is a special case of the more general Cosserat rod model [2]. The Cosserat model includes the effect of transverse shear strain and axial strain (elongation/compression), which are assumed to be zero in Kirchhoff models. Using the full Cosserat framework, we have the same rod state vector

$$\boldsymbol{x}(s) = \begin{bmatrix} p_x(s) & p_y(s) & \theta(s) \end{bmatrix}^\top,$$

but the state derivatives are functions of the control vector $\boldsymbol{u} = \begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix}^\top$ as follows:

$$\boldsymbol{x}_s = \boldsymbol{f}(s, \boldsymbol{x}, \boldsymbol{u}) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 + u_1 \\ u_2 \\ u_3 \end{bmatrix}, \tag{4.6}$$

where u_1 is the axial extension strain, u_2 is the transverse shear strain, and u_3 is the rod curvature (corresponding to u in the Kirchhoff model).

Assuming a linear constitutive equation then gives a strain energy density of the quadratic form $U(s) = \frac{1}{2}\boldsymbol{u}^\top(s)\boldsymbol{K}(s)\boldsymbol{u}(s)$. $\boldsymbol{K}(s) = \text{diag}\left\{E(s)A(s), G(s)A(s), E(s)I(s)\right\}$ where $E(s)$ is Young's modulus, $G(s)$ is the shear modulus, $A(s)$ is the rod's cross-sectional area, and $I(s)$ is the second area moment of inertia of the rod cross section about its centroidal axis. The potential energy of the loading is the same as in the Kirchhoff case above, which leads to the following energy functional:

$$J = \phi(\boldsymbol{x}_L) + \int_0^L \left(\frac{1}{2}\boldsymbol{u}^\top \boldsymbol{K}\boldsymbol{u} - \boldsymbol{w}^\top \boldsymbol{x}\right) ds.$$

The boundary-value problem resulting from the first-order necessary conditions is then exactly the same as the Kirchhoff case (4.2), except for the state derivatives and the

calculation of $\boldsymbol{u}$, which are given by (4.6) and

$$\boldsymbol{u} = \boldsymbol{K}^{-1} \begin{bmatrix} \cos\theta & \sin\theta & 0 \\ -\sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \boldsymbol{\lambda}.$$

In predicting deformation, it is important to consider the full Cosserat rod model instead of the Kirchhoff approximation in cases where the elastic member has a low slenderness ratio *or* a low axial or shear stiffness relative to the bending stiffness (e.g. a compression spring). In some cases, this difference can also affect stability behavior and the results of our stability assessment. Figure 4.3 illustrates the ratio of the critical buckling loads obtained with each model as a function of the slenderness ratio for a pinned steel tube. As the slenderness ratio decreases below 10, the results diverge, indicating that shear and axial stiffness can significantly affect the critical buckling load. While the rods in our experimental prototype have high axial stiffness and $L/r \approx 400$, soft elastomer parallel robots such as those studied in [88, 47] often have $L/r < 15$, and spring-backbone robots such as [124] have lower axial stiffness relative to bending stiffness. Additionally, when

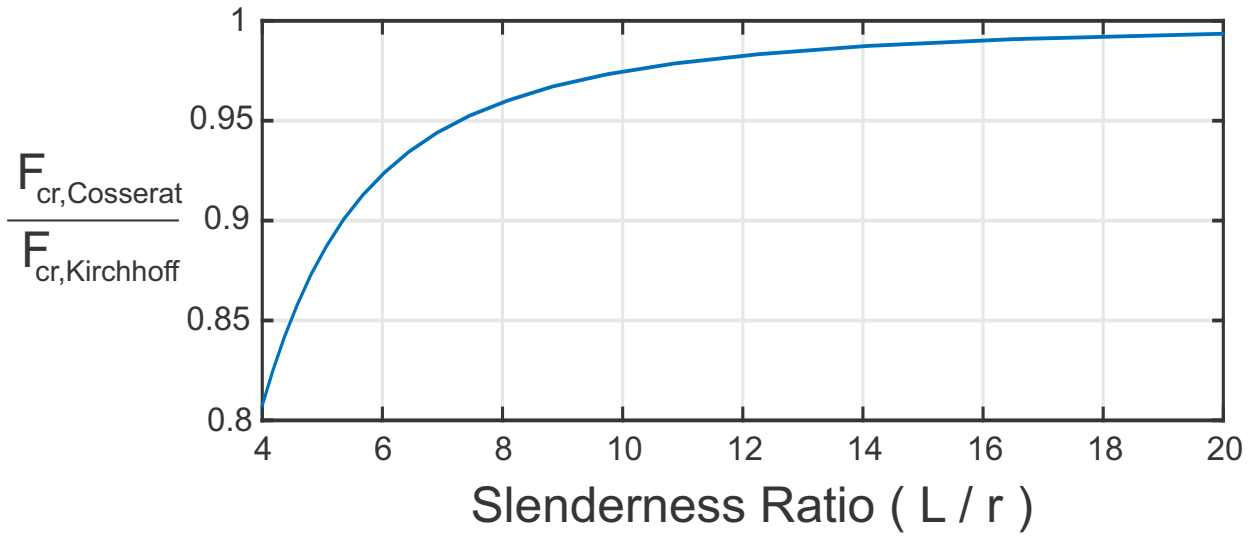


Figure 4.3: This plot illustrates the ratio of the critical buckling loads obtained using the Cosserat and Kirchhoff models as a function of slenderness ratio (total length over outer radius) for a steel tube with a wall thickness equal to 10% of the outer radius in the pinned case.

assessing stability, the Kirchhoff incompressibility constraint gives rise to an indeterminate special case (an "abnormal extremal") in the case of a straight rod with two completely fixed ends, as discussed in [12], while the Cosserat model avoids this complication by allowing axial compression. All of these factors motivate our study of the full Cosserat model in order for the approach and results to apply as generally as possible.

4.3 Coupled Planar Rods

Consider multiple planar Cosserat rods with their ends fixed to a rigid body as shown in Figure 4.4. The i^{th} rod has length L_i and is described by a state vector $\boldsymbol{x}_i = [\boldsymbol{p}_i \ \theta_i]^\top = [p_{x_i} \ p_{y_i} \ \theta_i]^\top$ with state derivatives (with respect to arc length s_i) given by

$$\frac{\partial \boldsymbol{x}_i}{\partial s_i} = \boldsymbol{f}(s_i, \boldsymbol{x}_i, \boldsymbol{u}_i) \tag{4.7}$$

and known initial state $\boldsymbol{x}_i(0)$, where $\boldsymbol{f}(s_i, \boldsymbol{x}_i, \boldsymbol{u}_i)$ has the form given in (4.6). We assume there is a globally defined wrench $\boldsymbol{W}$ applied at some reference point on the end-effector, the location of which is constant in each of the reference frames defined by the terminal rod poses, given by a known vector $\boldsymbol{r}_i = \begin{bmatrix} r_{ix} & r_{iy} \end{bmatrix}^\top$, so that the global location of the reference

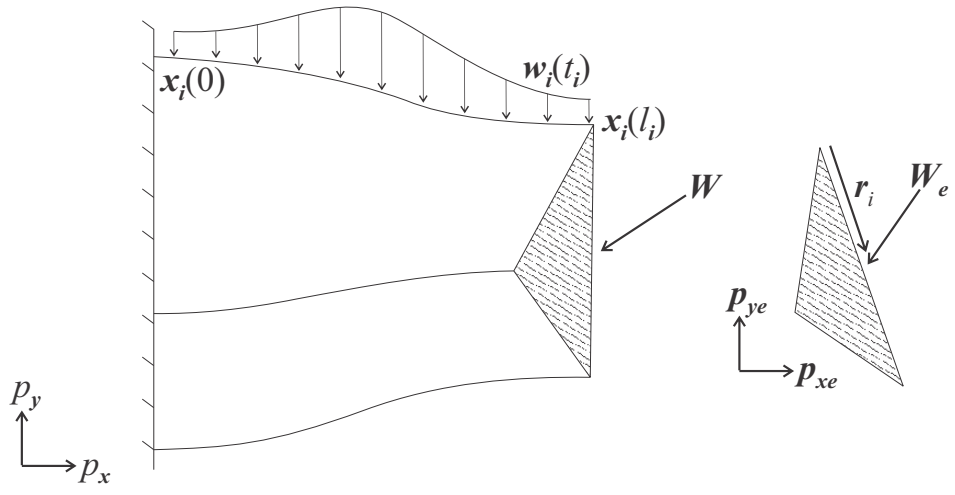


Figure 4.4: Multiple planar rods are constrained at their tips by a rigid body. The stability of the structure depends on the coupled behavior of the rods.

point $\boldsymbol{p}_r$ is

$$\boldsymbol{p}_r = \boldsymbol{p}_{iL} + \boldsymbol{R}_z(\theta_{iL})\boldsymbol{r}_i,$$

where $\boldsymbol{R}_z(\theta_i)$ denotes the standard two-dimensional rotation matrix associated with θ_i. For simplicity, we restrict the rods to attach to the rigid body with the same angle, so that $\theta_{iL} = \theta_{jL}$. The state is defined as $\boldsymbol{x} := \begin{bmatrix} \boldsymbol{x}_1^\top & ... & \boldsymbol{x}_n^\top \end{bmatrix}^\top$, and the terminal constraints are written as a single function so that

$$\boldsymbol{\beta}(\boldsymbol{x}_L) = \begin{bmatrix} \boldsymbol{\beta}_1^\top(\boldsymbol{x}_{1_L}, \boldsymbol{x}_{2_L}) & ... & \boldsymbol{\beta}_{n-1}^\top(\boldsymbol{x}_{n-1_L}, \boldsymbol{x}_{n_L}) \end{bmatrix}^\top = \boldsymbol{0},$$

where

$$\boldsymbol{\beta}_i(\boldsymbol{x}_{i_L}, \boldsymbol{x}_{(i+1)_L}) = \boldsymbol{x}_{i_L} + \begin{bmatrix} \cos\theta_{iL} & -\sin\theta_{iL} \\ \sin\theta_{iL} & \cos\theta_{iL} \\ 0 & 0 \end{bmatrix} (\boldsymbol{r}_i - \boldsymbol{r}_{i+1}) - \boldsymbol{x}_{(i+1)_L}.$$

The potential energy of the point-wrench $\boldsymbol{W}$ applied to the end effector is given by

$$\phi(\boldsymbol{x}_{1_L}) = -\boldsymbol{W}^\top \left(\boldsymbol{x}_{1_L} + \begin{bmatrix} \cos\theta_{1L} & -\sin\theta_{1L} \\ \sin\theta_{1L} & \cos\theta_{1L} \\ 0 & 0 \end{bmatrix} \boldsymbol{r}_1 \right).$$

Note that the choice to use the first rod to locate the end-effector reference point was arbitrary. The total potential energy of the system is then given by

$$J = \phi(\boldsymbol{x}_{1_L}) + \sum_{i=1}^{n} \int_0^{L_i} \frac{1}{2} \left(\boldsymbol{u}_i^\top \boldsymbol{K}_i \boldsymbol{u}_i - \boldsymbol{w}_i^\top \boldsymbol{x}_i \right) ds_i.$$

We can rewrite this expression by expressing each integral in terms of a common integration variable σ, where $s_i = \sigma L_i$ and $ds_i = L_i d\sigma$:

$$J = \phi(\boldsymbol{x}_L) + \int_0^1 \sum_{i=1}^{n} \frac{1}{2} \left(\boldsymbol{u}_i^\top \boldsymbol{K}_i \boldsymbol{u}_i - \boldsymbol{w}_i^\top \boldsymbol{x}_i \right) L_i d\sigma.$$

The final step in reformulating this problem to fit the optimal control framework is to rewrite the state derivatives with respect to σ as

$$\boldsymbol{x}_\sigma = \begin{bmatrix} L_1 \boldsymbol{f}_1 \left(s_1(\sigma), \boldsymbol{x}_1, \boldsymbol{u}_1\right) \\ \vdots \\ L_n \boldsymbol{f}_n \left(s_n(\sigma), \boldsymbol{x}_n, \boldsymbol{u}_n\right) \end{bmatrix}$$

The minimization problem now fits the Bolza form.

4.3.1 First-Order Necessary Conditions

Applying the first-order conditions of a Bolza problem (4.3) then results in the boundary value problem:

$$\begin{aligned} \boldsymbol{x}_{i_\sigma} &= L_i \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 + u_{i1} \\ u_{i2} \\ u_{i3} \end{bmatrix} \\ \boldsymbol{\lambda}_{i_\sigma} &= L_i \begin{bmatrix} -f_{ix} \\ -f_{iy} \\ \lambda_{i_1} \sin\theta_i - \lambda_{i2} \cos\theta_i - l_{iz} \end{bmatrix}, \end{aligned}$$

where

$$\boldsymbol{u}_i = \boldsymbol{K}_i^{-1} \begin{bmatrix} \cos\theta_i & \sin\theta_i & 0 \\ -\sin\theta_i & \cos\theta_i & 0 \\ 0 & 0 & 1 \end{bmatrix} \boldsymbol{\lambda}_i$$

subject to

$$\begin{aligned}
\boldsymbol{x}(0) &= \boldsymbol{x}_0 \\
\boldsymbol{\beta}_i(\boldsymbol{x}_L) &= \mathbf{0} \\
-F_x + \sum_{i=1}^{n} \boldsymbol{\lambda}_{i1L} &= 0 \\
-F_y + \sum_{i=1}^{n} \boldsymbol{\lambda}_{i2L} &= 0 \\
-M_z + \sum_{i=1}^{n} \boldsymbol{\lambda}_{i3L} + \begin{bmatrix} -\boldsymbol{\lambda}_{i2L} & \boldsymbol{\lambda}_{i1L} \end{bmatrix} \boldsymbol{R}_z(\theta_{iL}) \boldsymbol{r}_i &= 0.
\end{aligned}$$

We note that an equivalent formulation could be obtained in terms of derivatives with respect to s_i as $\partial \boldsymbol{x}_i / \partial s_i = (\partial \boldsymbol{x}_i / \partial \sigma)/L_i$ and $\partial \boldsymbol{\lambda}_i / \partial s_i = (\partial \boldsymbol{\lambda}_i / \partial \sigma)/L_i$, subject to the same boundary conditions above.

4.3.2 Second-Order Sufficient Conditions and Validation

To apply our test for stability, we need to calculate the $3n \times 3n$ matrix $\boldsymbol{E}_{\boldsymbol{\lambda}(\sigma)}$ at all points $\sigma \in [0\ 1)$. By definition, $\boldsymbol{E}_{\boldsymbol{\lambda}(\sigma)}$ can be written in block form as

$$\boldsymbol{E}_{\boldsymbol{\lambda}(\sigma)} = \begin{bmatrix} \boldsymbol{E}_{\boldsymbol{\lambda}_1(\sigma)} & \dots & \boldsymbol{E}_{\boldsymbol{\lambda}_n(\sigma)} \end{bmatrix},$$

where $\boldsymbol{E}_{\boldsymbol{\lambda}_i(\sigma)}$ is a $3n \times 3$ matrix that can be calculated using the 6×6 transition matrix of the i^{th} rod $\boldsymbol{\Phi}_i$:

$$\boldsymbol{E}_{\boldsymbol{\lambda}_i(\sigma)} = \frac{\partial \boldsymbol{E}}{\partial \boldsymbol{x}_{iL}} (\boldsymbol{\Phi}_i^{-1})_{12} + \frac{\partial \boldsymbol{E}}{\partial \boldsymbol{\lambda}_{iL}} (\boldsymbol{\Phi}_i^{-1})_{22}.$$

This formulation is efficient and modular; $\partial \boldsymbol{E}/\partial \boldsymbol{x}_{iL}$ and $\partial \boldsymbol{E}/\partial \boldsymbol{\lambda}_{iL}$ are computed once for all σ. Then, following (4.4), only the individual rod transition matrices are needed to compute the full $\boldsymbol{E}_{\boldsymbol{\lambda}(\sigma)}$ since each set of rod differential equations is decoupled from the others (the coupling only happens through the boundary conditions).

We can also compare the predictions of our approach to known results in the stability of frame and truss structures. We consider the special case of straight, parallel rods of equal length coupled by a rigid member at their distal ends, which is illustrated in Figure 4.5. This coupled structure is known as a sway frame. For a sway frame with n parallel columns of length L and bending stiffness EI, where the column pattern is symmetrical about the center and the frame is subjected to a downward vertical load applied at the center of the top, the in-plane critical buckling load is given by the analytical formula [21]

$$P_{cr} = n\frac{EI\pi^2}{L^2}.$$

For 3 cylindrical columns with $L = 1$, $E = 207$ GPa, and a radius of 0.001 m, the critical load is $P_{cr} = 4.814$N.

After applying our numerical approach to this same stability problem, we used a bisection method to iteratively find the smallest load for which our stability test fails, and obtained 4.814N, agreeing with the known analytical result to four digits. Note that we do not expect exact agreement in this case because we used axially compressible, shearable Cosserat rod models in this section, and the analytical formula assumes an Euler elastica with no axial compression or shear strain. However, due to the long, slender rod geometry chosen

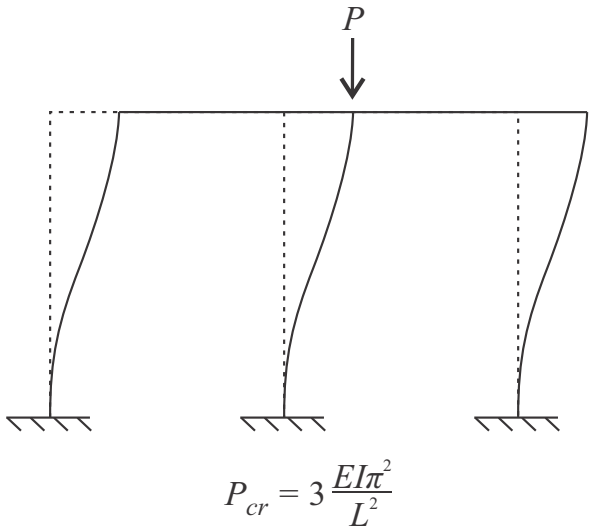


Figure 4.5: In the case of initially straight rods, the optimal control approach can be compared with analytical formulations for sway frames. The two methods are in agreement.

for this particular example, bending dominates the behavior, and the effects of shear and extension/compression are minimized, resulting in close agreement between our approach and the analytical formula.

4.4 Extension to Spatial Rods

4.4.1 First-Order Necessary Conditions

We here provide a short derivation of the first-order necessary conditions for a spatial Cosserat rod with general end constraints and conservative applied loads. The derivation follows the same pattern as Section 4.1, with the addition of Euler-Poincaré reduction following [45] because the rod state variable is not an element of a vector space as in the planar case, but is rather a member of the Lie group SE(3). The Cosserat rod state variable is a homogeneous transformation matrix $\boldsymbol{T}(s) \in \mathrm{SE}(3)$, which has the form

$$\boldsymbol{T}(s) = \begin{bmatrix} \boldsymbol{R}(s) & \boldsymbol{p}(s) \\ \boldsymbol{0} & 1 \end{bmatrix}.$$

The state differential equation is

$$\boldsymbol{T}_s = \boldsymbol{T}\widehat{\boldsymbol{\xi}},$$

where $\boldsymbol{\xi} = \begin{bmatrix} \boldsymbol{v}^\top & \boldsymbol{\omega}^\top \end{bmatrix}^\top$ is the body-frame twist associated with the arc length derivative of $\boldsymbol{T}$, composed of linear and angular components $\boldsymbol{v}$ and $\boldsymbol{\omega}$ [69]. These are analogous to linear and angular velocity, with derivatives with respect to arc-length instead of time. The $\widehat{\ }$ symbol is overloaded to denote the standard isomorphic mapping from $\mathbb{R}^6$ to $\mathfrak{se}(3)$ and $\vee$ is

similarly overloaded as defined in [69], that is

$$\widehat{\boldsymbol{y}} = \begin{bmatrix} 0 & -y_6 & y_5 & y_1 \\ y_6 & 0 & -y_4 & y_2 \\ -y_5 & y_4 & 0 & y_3 \\ 0 & 0 & 0 & 0 \end{bmatrix} \text{ for } \boldsymbol{y} \in \mathbb{R}^6.$$

Assuming a linear constitutive equation, the energy density per unit length stored in the rod's deformation then has the form

$$U = \frac{1}{2}\boldsymbol{u}^\top \boldsymbol{K}\boldsymbol{u},$$

where $\boldsymbol{u}(s) \in \mathbb{R}^m$ is any set of kinematic strain variables that we wish to consider and $\boldsymbol{K}(s)$ is the stiffness matrix associated with those strains. The twist $\boldsymbol{\xi}$ is a function of $\boldsymbol{u}$, and this framework naturally accounts for both the full Cosserat model (in which $\boldsymbol{u} = \boldsymbol{\xi} - [0\ 0\ 1\ 0\ 0\ 0]^\top$ for an initially straight rod with the z-axis of $\boldsymbol{R}$ pointing along the rod axis and $\boldsymbol{K} = \text{diag}\{GA, GA, EA, EI_{xx}, EI_{yy}, GI_{zz}\}$) and the shearless inextensible Kirchhoff model (in which $\boldsymbol{v} = [0\ 0\ 1]^\top$, $\boldsymbol{\omega} = \boldsymbol{u}$ and $\boldsymbol{K} = \text{diag}\{EI_{xx}, EI_{yy}, GI_{zz}\}$).

The potential energy of a distributed force $\boldsymbol{f}(t)$ and point force $\boldsymbol{F}$ applied at $t = t_L$ is

$$V = -\boldsymbol{F}^\top \boldsymbol{p}_L - \int_0^L \boldsymbol{f}^\top \boldsymbol{p}\ ds.$$

In contrast to our planar formulation, we here neglect any applied moments because constant applied moments (defined in either body-frame or global frame) are known to be non-conservative in the spatial case [126]. Stability analysis with non-conservative loadings requires a different framework and definition of stability in terms of dynamics. There are certain types of "rotation-dependent" moments that are conservative, but we will not consider them here, and we note that many moment loadings can be closely approximated by a suitably chosen distributed force. We can now state the optimal control problem for a

spatial rod as

$$\begin{aligned} \text{minimize} \quad & J = \phi(\boldsymbol{T}_L) + \int_0^L \mathcal{L}(s, \boldsymbol{T}, \boldsymbol{u}) ds \\ \text{subject to} \quad & \left(\boldsymbol{T}^{-1}\boldsymbol{T}_s\right)^\vee = \boldsymbol{\xi}(\boldsymbol{u}) \\ & \boldsymbol{T}(0) = \boldsymbol{T}_0, \quad \boldsymbol{\beta}(\boldsymbol{T}_L) = \boldsymbol{0} \end{aligned}$$

where

$$\begin{aligned} \phi(\boldsymbol{T}_L) &= -\boldsymbol{F}^\top \boldsymbol{p}_L \\ \mathcal{L}(s, \boldsymbol{T}, \boldsymbol{u}) &= \frac{1}{2}\boldsymbol{u}^\top \boldsymbol{K}\boldsymbol{u} - \boldsymbol{f}^\top \boldsymbol{p}, \end{aligned}$$

and we assume that $\boldsymbol{\beta}(\boldsymbol{T}_L)$ contains $p \leq 6$ independent constraints that can be satisfied by some $\boldsymbol{T}_L \in \mathrm{SE}(3)$. For example, a full rotation constraint $\boldsymbol{R}_L = \boldsymbol{I}$ can be expressed minimally by $\boldsymbol{\beta}(\boldsymbol{T}_L) = \left(\log\left(\boldsymbol{R}_L\right)\right)^\vee = \boldsymbol{0}$, for which $p = 3$.

One main difference between the problem statement above and that in (4.1) is that the state differential equations are written in their reduced form on the vector space $\mathbb{R}^6$ rather than on $\mathrm{SE}(3)$. As discussed by Holm in [45], this allows us to employ a reduced Lagrange multiplier vector $\boldsymbol{\lambda} \in \mathbb{R}^6$ to enforce the differential constraints and leads to the correct first-order conditions on the manifold. In [22], Chirkjian arrives at the same final equations (The Euler-Poincaré equations) by equivalently using the unreduced equations and the appropriate group law to formulate the variations. The augmented cost function is then

$$J' = G + \int_0^L \left[H + \boldsymbol{\lambda}^\top \left(\boldsymbol{T}^{-1}\boldsymbol{T}_s\right)^\vee\right] ds,$$

where the Hamiltonian is

$$H(s, \boldsymbol{T}, \boldsymbol{u}, \boldsymbol{\lambda}) = L(s, \boldsymbol{T}, \boldsymbol{u}) - \boldsymbol{\lambda}^\top \boldsymbol{\xi}(\boldsymbol{u})$$

and the augmented terminal cost function is

$$G(\boldsymbol{T}_L, \boldsymbol{\nu}) = \phi(\boldsymbol{T}_L) + \boldsymbol{\nu}^\top \boldsymbol{\beta}(\boldsymbol{T}_L).$$

Now the first variation of J' can be written as

$$\begin{aligned}\delta J' =& \operatorname{tr}\left(G_{\boldsymbol{T}_L}\delta\boldsymbol{T}_L\right) + \beta^\top(\boldsymbol{T}_L)\delta\boldsymbol{\nu} \\ & \int_0^L \left(H_{\boldsymbol{u}}\delta\boldsymbol{u} + \left(H_{\boldsymbol{\lambda}} - \left(\boldsymbol{T}^{-1}\dot{\boldsymbol{T}}\right)^{\vee T}\right)\delta\boldsymbol{\lambda}\right. \\ & \qquad \left. + \operatorname{tr}\left(H_{\boldsymbol{T}}\delta\boldsymbol{T}\right) + \boldsymbol{\lambda}^\top\delta\left(\boldsymbol{T}^{-1}\dot{\boldsymbol{T}}\right)^\vee\right)ds,\end{aligned}$$

where tr denotes the matrix trace operator, and the partial derivative of a scalar with respect to a matrix is defined as

$$H_{\boldsymbol{T}} = \frac{\partial H}{\partial \boldsymbol{T}} = \left[\frac{\partial H}{\partial \boldsymbol{T}_{i,j}}\right]^\top,$$

using consistent numerator layout. As discussed in [45] Chapter 7, this trace pairing between a matrix partial derivative and the matrix variation provides the correct expression for the resulting scalar variation. As also detailed in [45] (Equation 7.34), $\delta\left(\boldsymbol{T}^{-1}\dot{\boldsymbol{T}}\right)^\vee$ can be expressed as

$$\delta\left(\boldsymbol{T}^{-1}\dot{\boldsymbol{T}}\right)^\vee = \delta\dot{\boldsymbol{\Sigma}} + \left(\widehat{\boldsymbol{\xi}}\widehat{\delta\boldsymbol{\Sigma}} - \widehat{\delta\boldsymbol{\Sigma}}\widehat{\boldsymbol{\xi}}\right)^\vee = \delta\dot{\boldsymbol{\Sigma}} + \operatorname{ad}_{\boldsymbol{\xi}}\delta\boldsymbol{\Sigma},$$

where $\delta\boldsymbol{\Sigma} = \left(\boldsymbol{T}^{-1}\delta\boldsymbol{T}\right)^\vee$ and

$$\operatorname{ad}_{\boldsymbol{\xi}} = \begin{bmatrix}\widehat{\boldsymbol{\omega}} & \widehat{\boldsymbol{v}} \\ 0 & \widehat{\boldsymbol{\omega}}\end{bmatrix}.$$

Thus, $\delta\boldsymbol{T}$ is completely captured by the reduced variation $\delta\boldsymbol{\Sigma} \in \mathbb{R}^6$ as $\delta\boldsymbol{T} = \boldsymbol{T}\widehat{\delta\boldsymbol{\Sigma}}$, so we can write

$$\begin{aligned}\operatorname{tr}\left(H_{\boldsymbol{T}}\delta\boldsymbol{T}\right) =& \operatorname{tr}\left(H_{\boldsymbol{T}}\boldsymbol{T}\widehat{\delta\boldsymbol{\Sigma}}\right) = -\begin{bmatrix}(\boldsymbol{f}^\top\boldsymbol{R}) & \boldsymbol{0}\end{bmatrix}\delta\boldsymbol{\Sigma} \\ \operatorname{tr}\left(G_{\boldsymbol{T}_L}\delta\boldsymbol{T}_L\right) =& \operatorname{tr}\left(G_{\boldsymbol{T}_L}\boldsymbol{T}_L\widehat{\delta\boldsymbol{\Sigma}}_L\right) = C(\boldsymbol{T}_L,\boldsymbol{\nu})\delta\boldsymbol{\Sigma}_L,\end{aligned}$$

where

$$C(\boldsymbol{T}_L,\boldsymbol{\nu}) = \begin{bmatrix}G_{\boldsymbol{p}_L}\boldsymbol{R}_L & \left(\boldsymbol{R}^\top G_{\boldsymbol{R}_L}^\top - G_{\boldsymbol{R}_L}\boldsymbol{R}\right)^{\vee T}\end{bmatrix}.$$

The above expression for $C(\boldsymbol{T}_L, \boldsymbol{\nu})$ is consistent with the same calculation performed in [45] section 7.2, though notational choices make this unapparent at first ([45] uses denominator layout for matrix partial derivatives and defines the "breve" map as $\frac{1}{2}$ of the hat map). Subbing these results in and integrating by parts, we get

$$\begin{aligned}\delta J' = &\left(C(\boldsymbol{T}_L, \boldsymbol{\nu}) + \boldsymbol{\lambda}_f^\top\right) \delta\boldsymbol{\Sigma}_f + \boldsymbol{\beta}(\boldsymbol{T}_L)^\top \delta\boldsymbol{\nu} \\ &\int_0^L \Big(H_{\boldsymbol{u}} \delta\boldsymbol{u} + \left(H_{\boldsymbol{\lambda}} - \left(\boldsymbol{T}^{-1}\dot{\boldsymbol{T}}\right)^{\vee T} \right) \delta\boldsymbol{\lambda} \\ &\quad \Big(- \left[\boldsymbol{f}^\top \boldsymbol{R}\ 0\right] - \dot{\boldsymbol{\lambda}}^\top + \boldsymbol{\lambda}^\top \mathrm{ad}_{\boldsymbol{\xi}} \Big) \delta\boldsymbol{\Sigma} \Big) ds,\end{aligned}$$

where we have used $\boldsymbol{\lambda}_0^\top \delta\boldsymbol{\Sigma}_0 = 0$. A necessary condition for local optimality is that $\delta J' = 0$ for any $\delta\boldsymbol{u}$, $\delta\boldsymbol{\lambda}$, $\delta\boldsymbol{T}$ and $d\boldsymbol{\nu}$. Thus, we choose Lagrange multipliers so that

$$\begin{aligned}\dot{\boldsymbol{T}} &= \boldsymbol{T}\widehat{\boldsymbol{\xi}}(\boldsymbol{u}) \\ \dot{\boldsymbol{\lambda}} &= \mathrm{ad}^\top_{\boldsymbol{\xi}(\boldsymbol{u})}\boldsymbol{\lambda} - \begin{bmatrix} \boldsymbol{R}^\top \boldsymbol{f} \\ \boldsymbol{0} \end{bmatrix} \\ \boldsymbol{u} &= \boldsymbol{K}^{-1}\boldsymbol{\xi}_{\boldsymbol{u}}^\top \boldsymbol{\lambda} \\ \boldsymbol{T}(0) &= \boldsymbol{T}_0 \\ \boldsymbol{\beta}(\boldsymbol{T}_L) &= \boldsymbol{0} \\ \boldsymbol{\lambda}_L &= -\boldsymbol{C}^\top(\boldsymbol{T}_L, \boldsymbol{\nu}).\end{aligned}$$

The second differential equation above is equivalent to the classical equilibrium differential equations describing internal force and moment (in body-frame coordinates) in Cosserat rod theory, as follows:

$$\begin{aligned}(\boldsymbol{n}^b)_s &= -\widehat{\boldsymbol{\omega}}\boldsymbol{n}^b - \boldsymbol{R}^\top \boldsymbol{f} \\ (\boldsymbol{m}^b)_s &= -\widehat{\boldsymbol{\omega}}\boldsymbol{m}^b - \widehat{\boldsymbol{v}}\boldsymbol{n}^b.\end{aligned}$$

Thus, the internal force vector $\boldsymbol{n}^b$ is equivalent to $\boldsymbol{\lambda}_{1-3}$ and the internal moment vector $\boldsymbol{m}^b$ is equivalent to $\boldsymbol{\lambda}_{4-6}$.

As described in Section 4.1, we can pre-multiply both sides of $\boldsymbol{\lambda}_L = -\boldsymbol{C}^\top$ by $\boldsymbol{P}^\top$ where $\boldsymbol{P}$ is a matrix whose columns form an orthonormal basis for the nullspace of $\boldsymbol{C}_{\boldsymbol{\nu}} = \frac{\partial \boldsymbol{C}}{\partial \boldsymbol{\nu}}$. This eliminates $\boldsymbol{\nu}$ from the equations and provides a reduced set of $6-p$ boundary conditions for $\boldsymbol{\lambda}_L$. This results in a general set of terminal boundary conditions of the form

$$\boldsymbol{E}(\boldsymbol{T}_L, \boldsymbol{\lambda}_L) = \begin{bmatrix} \boldsymbol{\beta}(\boldsymbol{T}_L) \\ \left(\boldsymbol{P}^\top \boldsymbol{\lambda}_L + \boldsymbol{P}^\top \boldsymbol{C}^\top\right) \end{bmatrix} = \boldsymbol{0},$$

where $\boldsymbol{E} : \mathrm{SE}(3) \times \mathbb{R}^6 \to \mathbb{R}^6$.

4.4.2 Second-Order Sufficient Conditions

The second-order conditions can be obtained by restricting all admissible comparison paths to SE(3) and determining whether any are neighboring optimal paths. This requires only a slight modification to the calculation of $\det\left(\boldsymbol{E}_{\boldsymbol{\lambda}(s)}\right)$, which we will subsequently check for zeros on the interval $[0\ L)$. Recognizing that $\delta \boldsymbol{T} = \boldsymbol{T}\widehat{\delta \boldsymbol{\Sigma}}$, we define a reduced transition matrix $\boldsymbol{\Phi}(s, L)$ as

$$\begin{bmatrix} \delta \boldsymbol{\Sigma}(s) \\ \delta \boldsymbol{\lambda}(s) \end{bmatrix} = \boldsymbol{\Phi}(s, L) \begin{bmatrix} \delta \boldsymbol{\Sigma}_L \\ \delta \boldsymbol{\lambda}_L \end{bmatrix}. \tag{4.8}$$

$\boldsymbol{\Phi}(s, L)$ can be obtained by integrating the following differential equation from L to s, starting at $\boldsymbol{\Phi}(L, L) = \mathbf{I}$:

$$\dot{\boldsymbol{\Phi}} = \begin{bmatrix} -\mathrm{ad}_{\boldsymbol{\xi}} & \boldsymbol{\xi}_u \boldsymbol{K}^{-1} \boldsymbol{\xi}_u^\top \\ \boldsymbol{0} & \mathrm{ad}_{\boldsymbol{\xi}}^\top + \mathrm{cd}_{\boldsymbol{\lambda}} \boldsymbol{\xi}_u \boldsymbol{K}^{-1} \boldsymbol{\xi}_u^\top \end{bmatrix} \boldsymbol{\Phi} + \boldsymbol{M},$$

where $\mathrm{cd}_{\boldsymbol{\lambda}}$ is defined as

$$\mathrm{cd}_{\boldsymbol{\lambda}} = \begin{bmatrix} \boldsymbol{0} & \widehat{\boldsymbol{\lambda}}_{1-3} \\ \widehat{\boldsymbol{\lambda}}_{1-3} & \widehat{\boldsymbol{\lambda}}_{4-6} \end{bmatrix}$$

such that $\mathrm{cd}_{\boldsymbol{x}}\boldsymbol{y} = \mathrm{ad}_{\boldsymbol{y}}^{\top}\boldsymbol{x}$ for all $\boldsymbol{x}, \boldsymbol{y} \in \mathfrak{se}(3)$, and

$$\boldsymbol{M} = \begin{bmatrix} & \mathbf{0}_{6\times 12} & \\ (\boldsymbol{R}\boldsymbol{\Phi}_{4-6,1})^{\wedge}\,\mathbf{f} & \dots & (\boldsymbol{R}\boldsymbol{\Phi}_{4-6,12})^{\wedge}\,\mathbf{f} \\ & \mathbf{0}_{3\times 12} & \end{bmatrix}.$$

The above formulation is valid for both the Cosserat and Kirchhoff models.

δE_i can be expressed as

$$\begin{aligned} \delta E_i =& \mathrm{tr}\left(\frac{\partial E_i}{\partial \boldsymbol{T}_L}\delta \boldsymbol{T}_L\right) + \frac{\partial E_i}{\partial \boldsymbol{\lambda}_L}\delta\boldsymbol{\lambda}_L \\ =& \mathrm{tr}\left(\frac{\partial E_i}{\partial \boldsymbol{T}_L}\boldsymbol{T}_L\widehat{\delta\boldsymbol{\Sigma}}_L\right) + \frac{\partial E_i}{\partial \boldsymbol{\lambda}_L}\delta\boldsymbol{\lambda}_L \\ =& \boldsymbol{d}_i\delta\boldsymbol{\Sigma}_L + \frac{\partial E_i}{\partial \boldsymbol{\lambda}_L}\delta\boldsymbol{\lambda}_L, \end{aligned}$$

where

$$\boldsymbol{d}_i = \begin{bmatrix} \frac{\partial E_i}{\partial \boldsymbol{p}_L}\boldsymbol{R}_L & \left(\boldsymbol{R}^{\top}\left(\frac{\partial E_i}{\partial \boldsymbol{R}_L}\right)^{\top} - \frac{\partial E_i}{\partial \boldsymbol{R}_L}\boldsymbol{R}_L\right)^{\vee T} \end{bmatrix}$$

is a row vector. Thus, $\boldsymbol{E}_{\boldsymbol{\lambda}(s)}$ is given by

$$\begin{aligned} \boldsymbol{E}_{\boldsymbol{\lambda}(s)} =& \boldsymbol{D}\frac{\partial \boldsymbol{\Sigma}_L}{\partial \boldsymbol{\lambda}(s)} + \frac{\partial \boldsymbol{b}}{\partial \boldsymbol{\lambda}_L}\frac{\partial \boldsymbol{\lambda}_L}{\partial \boldsymbol{\lambda}(s)} \\ =& \boldsymbol{D}\left(\boldsymbol{\Phi}^{-1}\right)_{12} + \frac{\partial \boldsymbol{b}}{\partial \boldsymbol{\lambda}_L}\left(\boldsymbol{\Phi}^{-1}\right)_{22}, \end{aligned} \tag{4.9}$$

where the i^{th} row of matrix $\boldsymbol{D}$ is $\boldsymbol{d}_i$, and $\left(\boldsymbol{\Phi}^{-1}\right)_{12}$ and $\left(\boldsymbol{\Phi}^{-1}\right)_{22}$ denote the upper right and lower right 6×6 blocks of $\boldsymbol{\Phi}^{-1}(s, L)$, respectively, with $\boldsymbol{\Phi}(s, L)$ as defined in (4.8). $\frac{\partial \boldsymbol{E}}{\partial \boldsymbol{\lambda}_L}$ and each row of $\boldsymbol{D}$ can be easily obtained analytically by direct differentiation of $\boldsymbol{E}(\boldsymbol{T}_L, \boldsymbol{\lambda}_L)$ or approximated by finite differences.

An alternative way to compute $\boldsymbol{\Phi}(s, L)$ is to approximate it numerically by a finite difference procedure. We can increment a final variable by a numerically small amount Δ, integrate the model equations backwards to get the associated changes at t, and divide by the increment to obtain a column of $\boldsymbol{\Phi}(s, L)$, with the following two modifications:

- For the columns of $\boldsymbol{\Phi}(s, L)$ corresponding to $\delta\boldsymbol{\Sigma}_L$, the increment in $\boldsymbol{T}_L$ is generated by choosing a $\Delta\boldsymbol{\Sigma}_L$ and calculating $\Delta\boldsymbol{T}_L = \boldsymbol{T}_L\widehat{\Delta\boldsymbol{\Sigma}}_L$.

- For the rows of $\boldsymbol{\Phi}(s, L)$ associated with $\delta\boldsymbol{\Sigma}(s)$, the change in $\boldsymbol{\Sigma}(s)$ is calculated as $\Delta\boldsymbol{\Sigma}(s) = \left(\boldsymbol{T}^{-1}(s)\Delta\boldsymbol{T}(s)\right)^{\vee}$.

4.5 Parallel Continuum Robots

Following our development of the coupled planar rods case in Section 4.3, we now extend that approach to the spatial case to formulate the first-order necessary conditions for spatial parallel continuum robots and apply our stability test. We consider multiple Cosserat rods with their terminal ends attached to a rigid body, again representing a spatial parallel continuum robot with an end-effector as shown in Figure 4.6.

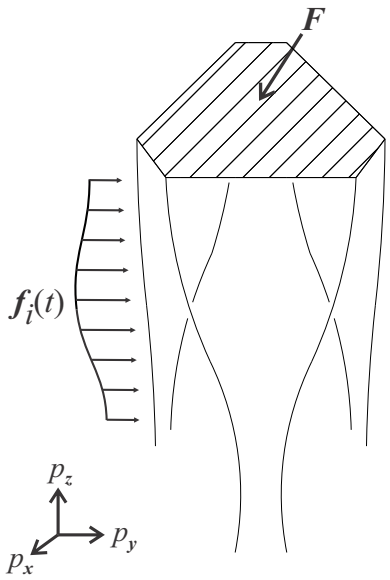


Figure 4.6: Spatial rods are constrained by a rigid body to form a single structure. The rods have rigid attachments to the ground and the body. There is some point force on the rigid body, and the rods are subject to distributed forces.

The i^{th} rod has length l_i and is described by a state variable $\boldsymbol{T}_i = \begin{bmatrix} \boldsymbol{R}_i & \boldsymbol{p}_i \\ \boldsymbol{0} & 1 \end{bmatrix} \in \mathrm{SE}(3)$ with state derivatives (with respect to arc length s_i) given by

$$\frac{d\boldsymbol{T}_i}{ds_i} = \boldsymbol{T}_i \widehat{\boldsymbol{\xi}}_i(\boldsymbol{u}_i) \tag{4.10}$$

and known initial state $\boldsymbol{T}_i(0) = \boldsymbol{T}_{i0}$. Our full state variable is then the set $\boldsymbol{x} = \left\{\boldsymbol{T}_1^\top \ ... \ \boldsymbol{T}_n^\top\right\}^\top$ with derivatives given by (4.10). Defining $\boldsymbol{r}_i$ as the fixed location of the i^{th} terminal rod end with respect to the end-effector frame $\boldsymbol{T}_e = \begin{bmatrix} \boldsymbol{R}_e & \boldsymbol{p}_e \\ \boldsymbol{0} & 1 \end{bmatrix} \in \mathrm{SE}(3)$, the global positions of the rod ends would be

$$\boldsymbol{p}_{iL} = \boldsymbol{p}_e + \boldsymbol{R}_e \boldsymbol{r}_i.$$

We also consider that each terminal rod orientation $\boldsymbol{R}_{iL}$ is constrained to equal to the end-effector orientation as

$$\left(\log\left(\boldsymbol{R}_e^\top \boldsymbol{R}_{iL}\right)\right)^\vee = \boldsymbol{0}.$$

Since the end effector frame is not known a priori, we must eliminate $\boldsymbol{p}_e$ and $\boldsymbol{R}_e$ from the relations above to obtain $5n$ constraint equations:

$$\boldsymbol{\beta}_i(\boldsymbol{T}_L) = \begin{bmatrix} \boldsymbol{p}_{iL} - \boldsymbol{p}_{1L} - \boldsymbol{R}_{1L}\left(\boldsymbol{r}_i - \boldsymbol{r}_1\right) \\ \left(\log\left(\boldsymbol{R}_{1L}^\top \boldsymbol{R}_{iL}\right)\right)^\vee \end{bmatrix} = \boldsymbol{0}$$

for $i = 2 : n$.

We assume there is a globally defined force $\boldsymbol{F}$ applied at the origin of the end-effector frame. The potential energy of this loading is then

$$\phi(\boldsymbol{T}_L) = -\boldsymbol{F}^\top \boldsymbol{p}_e = -\boldsymbol{F}^\top\left(\boldsymbol{p}_{1L} - \boldsymbol{R}_{1L}\boldsymbol{r}_1\right).$$

The total potential energy of the system is then given by

$$J = \phi(\boldsymbol{x}_L) + \sum_{i=1}^{n} \int_0^{L_i} \frac{1}{2} \left(\boldsymbol{u}_i^\top \boldsymbol{K}_i \boldsymbol{u}_i - \boldsymbol{f}_i^\top \boldsymbol{p}_i \right) ds_i,$$

where $\boldsymbol{f}_i \in \mathbb{R}^3$ is the global distributed force on rod i. Similar to the planar case, we rewrite this expression by expressing each integral in terms of a common integration variable σ, where $s_i = \sigma L_i$ and $ds_i = L_i d\sigma$,

$$J = \phi(\boldsymbol{x}_L) + \int_0^1 \sum_{i=1}^{n} \frac{1}{2} \left(\boldsymbol{u}_i^\top \boldsymbol{K}_i \boldsymbol{u}_i - \boldsymbol{f}_i^\top \boldsymbol{p}_i \right) L_i d\sigma$$

and rewrite the state derivatives with respect to σ as

$$\boldsymbol{x}_\sigma = \begin{bmatrix} L_1 \boldsymbol{T}_1 \widehat{\boldsymbol{\xi}}_1(\boldsymbol{u}_1) \\ \vdots \\ L_n \boldsymbol{T}_n \widehat{\boldsymbol{\xi}}_n(\boldsymbol{u}_n) \end{bmatrix}.$$

Following the development in Section 4.4, the first-order necessary conditions are then

$$\begin{aligned}
\boldsymbol{T}_{i_\sigma} &= L_i \boldsymbol{T}_i \widehat{\boldsymbol{\xi}}_i(\boldsymbol{u}_i) \\
\boldsymbol{\lambda}_{i_\sigma} &= L_i \left(\mathrm{ad}^\top_{\boldsymbol{\xi}_i(\boldsymbol{u}_i)} \boldsymbol{\lambda}_i - \begin{bmatrix} \boldsymbol{R}_i^\top \boldsymbol{f}_i \\ \boldsymbol{0} \end{bmatrix} \right) \\
\boldsymbol{u}_i &= \boldsymbol{K}_i^{-1} \frac{\partial \boldsymbol{\xi}_i}{\partial \boldsymbol{u}_i}^\top \lambda_i \\
\boldsymbol{T}_i(0) &= \boldsymbol{T}_{i0} \\
\boldsymbol{\beta}(\boldsymbol{T}_L) &= \boldsymbol{0} \\
\boldsymbol{0} &= -\boldsymbol{F} + \sum_{i=1}^{n} \boldsymbol{R}_{iL} \boldsymbol{\lambda}_{i1-3L} \\
\boldsymbol{0} &= -\sum_{i=1}^{n} \left[\boldsymbol{\lambda}_{i4-6L} + \boldsymbol{r}_i \times \boldsymbol{\lambda}_{i1-3L} \right].
\end{aligned}$$

Except for the rigid rod attachments as opposed to shaft collars, this is the same as the system derived in Chapter 2.2.

4.5.1 Second-Order Sufficient Conditions

The stability test for coupled spatial rods parallels our development of the coupled planar rods case in Section 4.3.2. We need to calculate the $6n \times 6n$ matrix $\boldsymbol{b}_{\lambda(\sigma)}$ at all points $\sigma \in [0\ 1)$. Again, $\boldsymbol{E}_{\lambda(\sigma)}$ can be written in block form as

$$\boldsymbol{E}_{\lambda(\sigma)} = \begin{bmatrix} \boldsymbol{E}_{\lambda_1(\sigma)} & \dots & \boldsymbol{E}_{\lambda_n(\sigma)} \end{bmatrix},$$

where $\boldsymbol{E}_{\lambda_i(\sigma)}$ is a $6n \times 6$ matrix which can be calculated using the 6×6 transition matrix of the i^{th} rod $\boldsymbol{\Phi}_i$ as follows:

$$\boldsymbol{E}_{\lambda_i(\sigma)} = \boldsymbol{D}_i(\boldsymbol{\Phi}_i^{-1})_{12} + \frac{\partial \boldsymbol{E}}{\partial \boldsymbol{\lambda}_{iL}}(\boldsymbol{\Phi}_i^{-1})_{22},$$

where $\boldsymbol{D}_i$ is the $\boldsymbol{D}$ matrix for rod i as in (4.9).

Thus, after finding a solution for the first-order conditions, the conjugate point test is performed by solving for the transition matrices of each individual rod $\boldsymbol{\Phi}_i(\sigma)$, computing $\boldsymbol{D}_i$ and $\boldsymbol{E}_{\boldsymbol{\lambda}_L,i}$ from the BVP, calculating $\boldsymbol{E}_\lambda(\sigma)$, and checking for singularities in $\boldsymbol{E}_\lambda(\sigma)$ at a large number of discrete points over $0 \leq \sigma < 1$.

4.6 Validation and Application

We implemented a C++ algorithm that solves the first-order optimality conditions for a parallel continuum robot and subsequently performs our second-order Jacobi conjugate-point test. We used a shooting method comprised of classical 4th-order Runge-Kutta integration of the first-order necessary conditions of a single rod and Levenberg-Marquardt optimization for the coupled boundary-value constraints as described in [13, 110]. Figure 4.7 shows the physical robot, the model solution, and the Jacobi conjugate-point test for an example configuration. The conjugate point test is performed as described in Section VI. The test computation time is about 2 milliseconds for a model with 40 fourth-order Runge-Kutta steps per rod on an Intel i7-4770 processor. Figure 4.7 shows a translation movement of

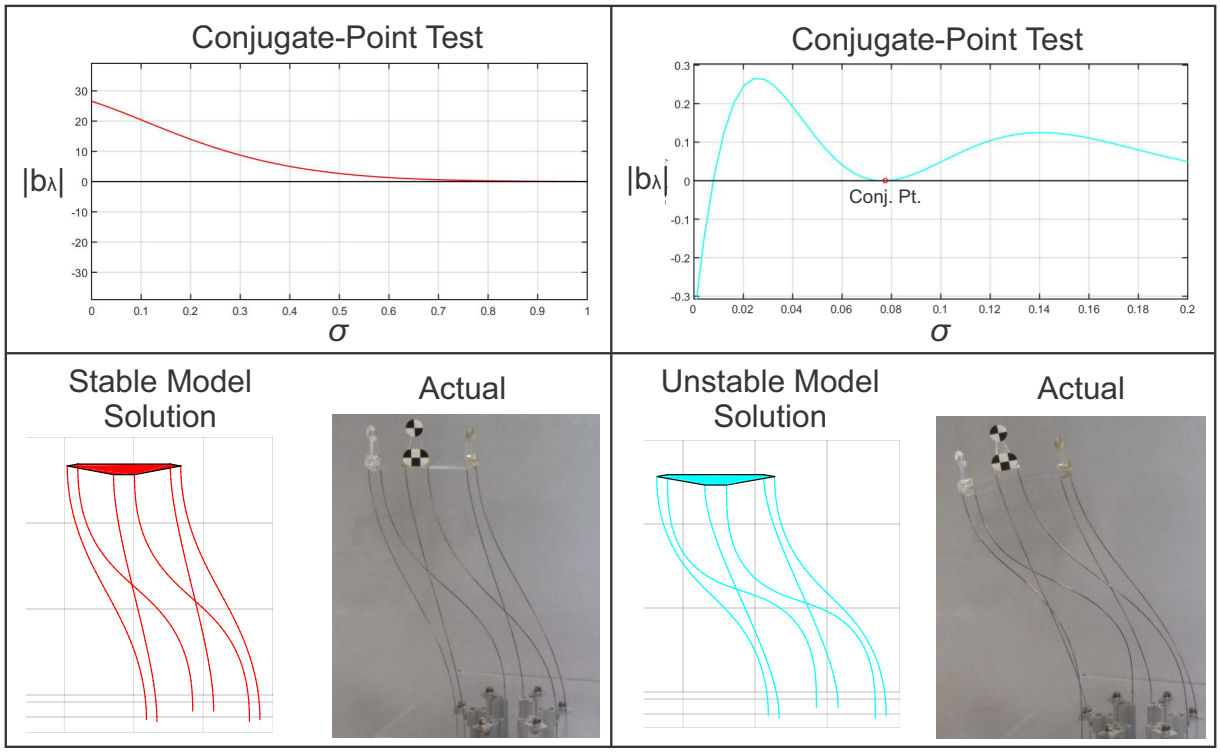


Figure 4.7: On the left, a model solution of the first-order necessary conditions represents a stable configuration as evidenced by the absence of any conjugate points on the interval. The physical robot configuration is stable and corresponds to the stable model solution. On the right, a model solution to the first-order necessary equations is shown to be unstable and thus not physically achievable (not a local energy minimizer) as evidenced by the first conjugate point on the interval (indicated by the circle). The physical robot (under the same actuation conditions) is of course at a stable configuration, but it arrived there after an unstable transition.

the robot during which our test predicts instability at the same moment when the physical robot shape diverges from the model solution. The accompanying video shows a detailed time-lapse breakdown of this case.

4.6.1 Calibration and Measurement

We performed a simple calibration procedure with a cantilevered spring steel (ASTM A228) rod to determine the equivalent Young's modulus of the rods in our prototype robot as shown in Figure 4.8. Taking the length, radius, and tip load as known constants, we measured the vertical tip deflection with calipers and used the bisection method to determine the correct Young's modulus which causes the model to predict the measured displacement of 52.27mm. This turned out to be $E = 183.41$ GPa. We assumed a Poisson's ratio of 0.3, which results in a shear modulus of 70.54 GPa.

Tracking marker rigid transformations were calibrated by comparing the model-based simulated end-effector pose with the true end-effector pose as determined by a MicronTracker 3 stereoscopic camera model XB3-BW-H360 from Claron Technology, Inc. Tracking markers

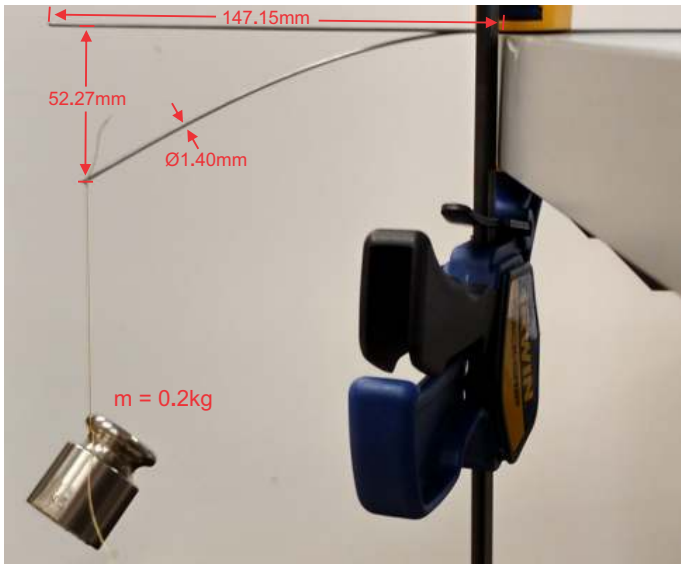


Figure 4.8: Two rods with diameter 1.40mm were clamped to form cantilevers of equal length 147.15mm. A mass of 0.2kg was attached at the distal end of a rod and the vertical displacement was 52.27mm. The mass was known within ANSI/ASTM Class 7 tolerances and lengths were measured with calipers. The resulting Young's modulus was 183.41 GPa.

were placed at the base and end effector so that the relative transformation between the two could be measured. The position offsets from robot frames to tracker marker frames were measured with calipers, while the rotation offsets were optimized to minimize the rotation error described by (4.11).

4.6.2 Experimental Validation

We performed experiments in which we commanded the robot to move from a stable configuration along several precomputed paths in actuation space that produced an unstable model configuration during quasistatic simulation. The weight of the rods, platform, and markers was considered negligible during the precomputation stage when the motion paths were generated.

Six motions were performed– pure translation in the x and y directions, pure rotation about the x, y, and z axes, and translation in the x-axis with simultaneous rotation about the y-axis (a bending motion). This set of motions was repeated at three different heights: $p_z = 0.2\text{m}, 0.25\text{m}, 0.3\text{m}$. Each trial at a specific height was repeated five times to evaluate repeatability. The results of these trials are shown in Figure 4.9. The errors are calculated by:

$$\begin{aligned} \text{Position Error} &= \|\boldsymbol{p}_{sim} - \boldsymbol{p}_{meas}\| \\ \text{Rotation Error} &= \left\| \left[\log\left(\boldsymbol{R}_{sim}\boldsymbol{R}_{meas}^{\top}\right)\right]^{\vee} \right\|. \end{aligned} \tag{4.11}$$

As the reader may infer from the graphs in Figure 4.9, large-scale pose transitions are observed for pure translations in the x and y directions and pure rotation about the z-axis. The numerical conjugate-point test triggered in close proximity to these buckling events. The accompanying video shows a detailed breakdown for the experimental translation in the x translation case.

The experiments were also informative in the other three cases where large-scale unstable pose transitions were not observed. In the lower three tests of Figure 4.9, the errors remain low despite the appearance of conjugate points (except for X-Rotation, in which no conjugate points appeared). This discrepancy could be due to small construction and assembly

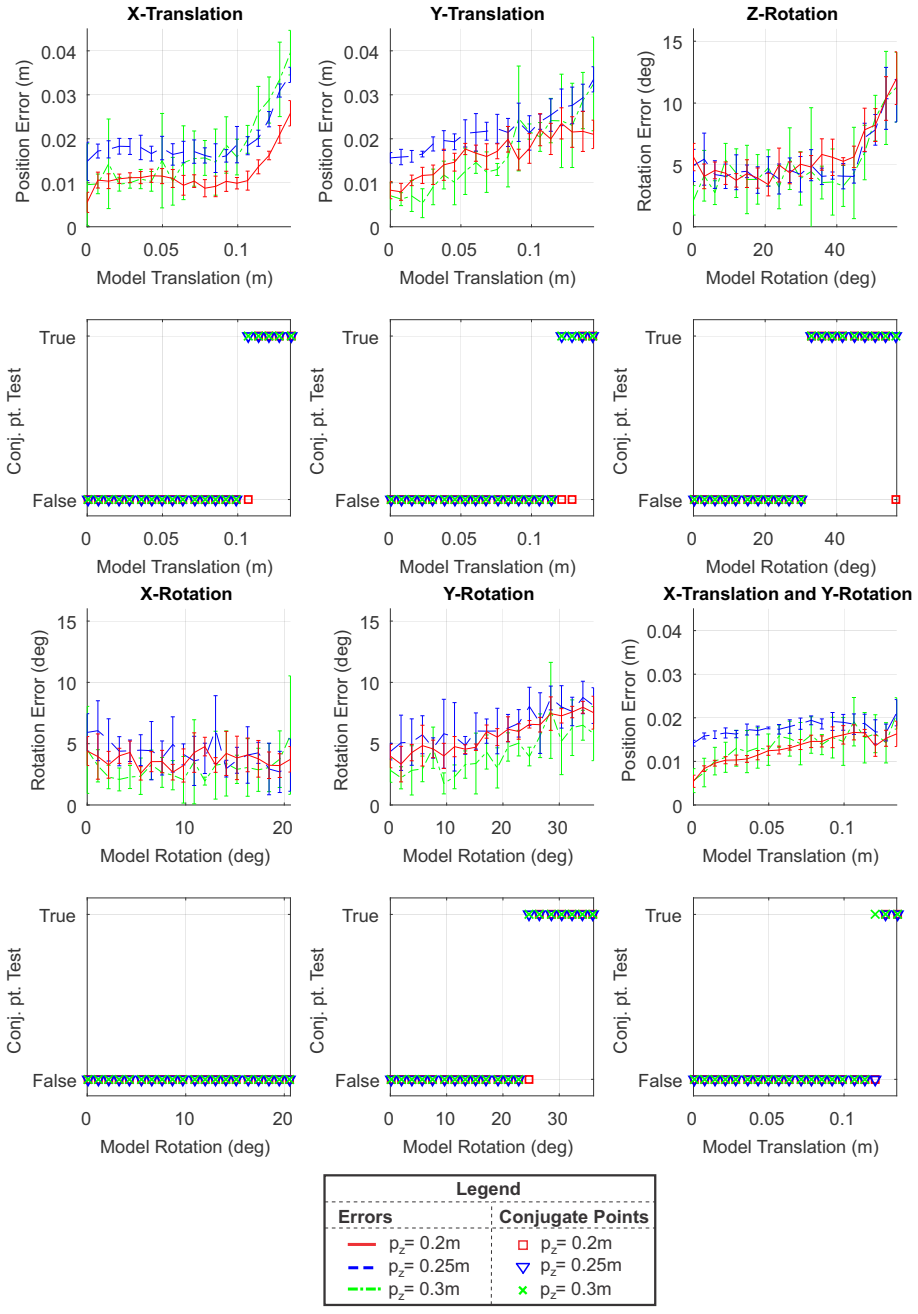


Figure 4.9: The robot was either translated or rotated toward a conjugate point, and in the lower right case, there was a simultaneous translation and rotation such that the rotation in radians was 5.4 times the translation in meters. The model pose was compared against the actual robot pose as measured by a stereoscopic camera. The six motions were performed at three different heights p_z. For each motion at a specific height, the motion was repeated five times, and the mean error is shown with the standard deviation. In every case where the two metrics suddenly diverged, this corresponded to the presence of a conjugate point. Thus, the conjugate point test is effective for assessing stability. However, the conjugate point test is conservative in that it can detect instability even when large-scale pose transitions do not occur, as seen in the bottom three cases.

tolerances because the stability boundary could be sensitive to small changes in these parameters, as discussed in the next section. Note also that instability does not necessarily imply a large dynamic pose divergence. The experiment is limited to measurements in the finite-dimensional space SE(3), but an unstable transition can be subtle and primarily affecting a rod's continuous path as opposed to being manifested at the end effector where the pose is measured.

In addition to quantifiable data, observations of the robot may yield insight. The three cases with large errors exhibited a steady increase in the error rather than a sudden one. For the three cases where the error did not drastically increase, there was also no significant deviation between the model configuration and the physical robot, although perturbing the robot in these configurations resulted in large vibrations of the system with very long settling times, indicating a system on the verge of stability.

4.6.3 Sensitivity to Parameters

In practice, we may be concerned about the behavior of the conjugate-point test with respect to changes in joint variables (studied above) and also changes in external loading and other problem parameters such as Young's Modulus. Even in classical column buckling problems, the practical buckling limit is significantly lower than the theoretical buckling limit when minor load eccentricities are considered. We demonstrate sensitivity of robot stability to end-effector loads in the three directions in Figure 4.10 by plotting a region of the stability boundary (as computed by a brute-force simulation using our numerical test) for our continuum Stewart-Gough robot in a configuration with equal leg lengths. The figure shows that for approximately vertical loads, small changes in load direction can greatly affect the magnitude of the critical load, as is the case for classic straight column buckling.

Avoiding instability in the presence of such parameter sensitivity is a difficult problem. One potential solution is to simulate a small change in every parameter affecting the robot (e.g. actuator positions, external loads, elastic moduli, etc.) to determine if small errors or movements in these values would result in instability. Such a discretized test would probably result in safer operation, but it could still fail to rigorously guarantee detection of impending

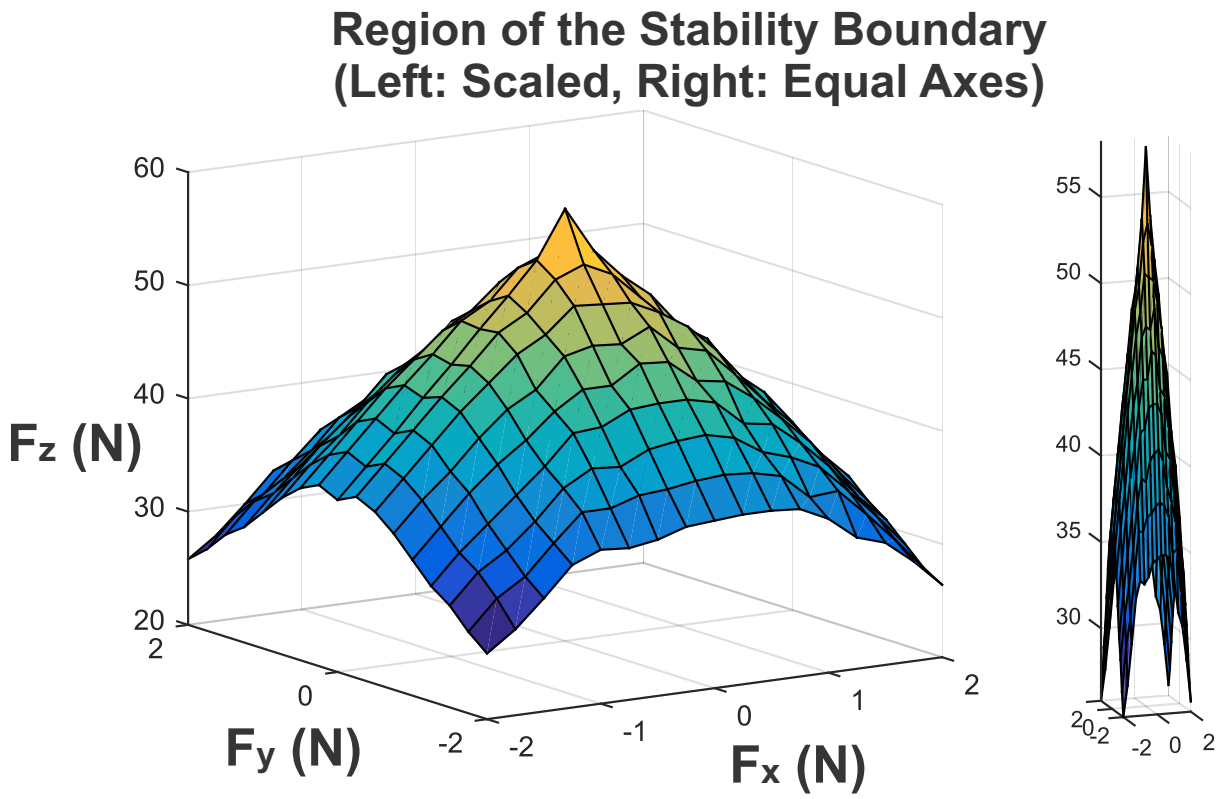


Figure 4.10: A region of the stability boundary with respect to 3D end-effector forces was generated for a continuum Stewart-Gough robot having equal leg lengths using a brute-force search. The stability boundary has an elongated, conical shape, indicating that for approximately vertical loads, small changes in load direction can greatly affect the magnitude of the critical load.

instabilities, and it still would not contain any information about how "far" the system is from instability.

We also investigated the sensitivity of conjugate point location to variations in Young's modulus. This was explored in simulation, as shown in Figure 4.11. Two configurations were considered– the nominal configuration where all legs have equal arc length of 0.3m and a bent configuration where the legs were given arc lengths of $L = \begin{bmatrix} 0.32 & 0.32 & 0.30 & 0.29 & 0.29 & 0.30 \end{bmatrix}$ m. The derivative of the conjugate point location with respect to Young's modulus $d\sigma_c/dE$, with units of GPa^{-1}, was found for various compressive loads by a first order finite difference of $\sigma_c(E, F_z)$. The results show that the derivative has a small magnitude and near-linear behavior and that the sensitivity to Young's modulus is zero when the load is zero. This is intuitive because changing Young's modulus merely scales the energy functional in the unloaded case.

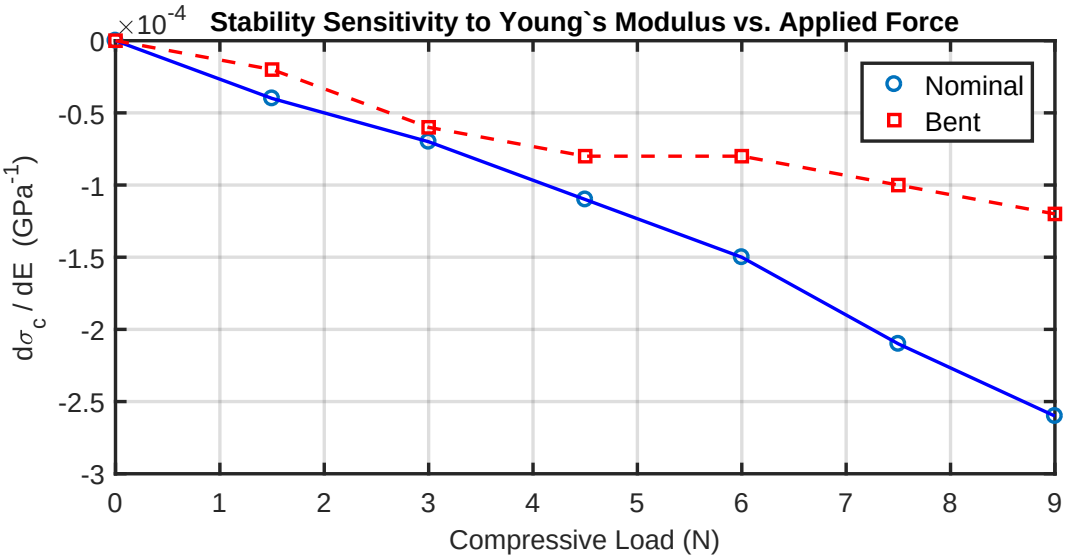


Figure 4.11: The sensitivity of conjugate point location with respect to stiffness was studied in simulation for two configurations under varying compressive load. The magnitude of $d\sigma_c/dE$ is relatively small, indicating that measurement errors in the stiffness calculation would have a minor impact.

4.6.4 Use of a Stability Heuristic

In addition to detecting if a conjugate point falls inside the integration interval, it can be helpful to consider the location σ_{cp} where a conjugate point occurs, even if σ_{cp} falls outside the interval [0 1) ($\sigma_{cp} < 0$). In this case, the value $d = -\sigma_{cp}$ indicates nearness to a critically stable system state with respect to changes in integration length. Using d as a heuristic stability metric, one could numerically compute the sensitivity (gradient) of d to changes in any problem parameters (e.g. actuator positions, external loads, elastic moduli, etc.) via finite difference approximations using the same sampling technique described above. Combined with knowledge of parameter uncertainty, this sensitivity vector could be used to obtain an estimation of how much change in each parameter (and in what direction) can be tolerated before the system becomes unstable. However, we note again that this heuristic should be used with caution because we have not proven that conjugate point location is continuous in every possible problem parameter. We leave further exploration of this topic to future work.

4.7 Conclusion

Starting from a rich literature on optimal control and stability assessment, we have derived a sufficient numerical test for the stability of Cosserat rods with arbitrary terminal constraints, including the multi-rod structures of parallel continuum robots. We validated the approach in simulation by comparing our results to classical results in the special cases of straight column buckling and sway frame buckling. We have further implemented the test to assess stability of a 6 DOF prototype parallel continuum robot, and the experimental data supports the effectiveness of the test.

Parallel continuum robot research and applications were previously hindered by the inability to recognize unstable model solutions. Our test provides this capability, which will enable robust model-based design, motion planning, and control in future work toward applications in robotic surgery and human-robot interaction. We also hope that other applications in robotics and elsewhere will benefit from our simple and general derivation

of the first- and second-order conditions for spatial elastic rods with arbitrary terminal constraints. For example, our approach could be adapted to assess the stability of other continuum robots and long, elastic objects in cases whenever partial end-pose constraints or coupling occur, such as contact with rigid fixtures or two robots manipulating the same object.

Chapter 5

Conclusions

Mathematical models of continuum robots may be used to predict and control robot motions. Dynamic behaviors can be accurately simulated at interactive rates with an appropriate discretization strategy and a careful implementation. The scheme employed here of implicit time semi-discretization and spatial shooting is accurate, efficient, and generalizes well to various classes of continuum robot. The approach to static and dynamic problems is unified since solving for a static robot state utilizes the same process as solving a single time-step of the dynamics problem. The elastic stability of robots may be studied by dynamic simulation or variational approaches.

5.1 Potential Future Work

While the topic of forward dynamic simulation has been addressed, dynamic control is a subject for future work. With dynamic control schemes one could accomplish manipulation tasks faster than quasi-static control schemes allow, and possibly provide stabilizing control inputs to unstable robots. However, it should be noted that boundary control of nonlinear PDE systems is nontrivial.

In general the issue of environmental and self-contact constraints has been ignored in this dissertation. For some tasks this omission is acceptable since contact is not likely to occur, e.g. parallel continuum robot teleoperation. However, contact is of theoretical interest and practically relevant for many tasks. The shooting approach may be fundamentally poorly

suited to model contacts at arbitrary locations since the robot connectivity graph is unknown for such problems. Investigation of arbitrary continuous-contact models and discretizations would likely be an appropriate topic for future works.

Much of the effort required to implement a continuum robot simulation comes from the tedium of implementing a discretized model in an imperative programming language. The differential equations and boundary conditions describing a continuum robot are terse compared to the hundreds or thousands of lines of code which ultimately define an imperative program to simulate a robot. A mathematical declarative-programming paradigm has the potential to close this gap, simultaneously reducing implementation effort and allowing a greater focus on the mathematical principles which are the core of the robot model.

Bibliography

[1] Anderson, V. C. and Horn, R. C. (1967). Tensor Arm Manipulator Design. *Transactions of the ASME*, 67. 3

[2] Antman, S. S. (2005). *Nonlinear Problems of Elasticity Second Edition*. Springer, New York, NY. 4, 12, 40, 109, 112

[3] Benosman, M. and Le Vey, G. (2004). Control of Flexible Manipulators: A Survey. *Robotica*, 22(5):533–545. 4

[4] Bergeles, C. and Dupont, P. E. (2013). Planning Stable Paths for Concentric Tube Robots. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3077–3082. IEEE. 8

[5] Bergeles, C., Gosline, A. H., Vasilyev, N. V., Codd, P. J., del Nido, P. J., and Dupont, P. E. (2015). Concentric Tube Robot Design and Optimization Based on Task and Anatomical Constraints. *IEEE Transactions on Robotics*, 31(1):67–84. 1, 8

[6] Bergou, M., Wardetzky, M., Robinson, S., Audoly, B., and Grinspun, E. (2008). Discrete Elastic Rods. *ACM Transactions on Graphics*, 27(3):1. 5

[7] Bertails, F., Audoly, B., Querleux, B., Leroy, F., Lévêque, J.-L., and Cani, M.-P. (2005). Predicting Natural Hair Shapes by Solving the Statics of Flexible Rods. In *Eurographics (short papers)*. Eurographics. 5

[8] Black, C. (2017). *Modeling, Analysis, Force Sensing and Control of Continuum Robots for Minimally Invasive Surgery*. PhD thesis, The University of Tennessee, Knoxville. 2, 5

[9] Black, C. B., Till, J., and Rucker, D. C. (2018). Parallel Continuum Robots: Modeling, Analysis, and Actuation-Based Force Sensing. *IEEE Transactions on Robotics*, 34(1):29–47. 2, 5, 20, 35, 36

[10] Borum, A. and Bretl, T. (2015). The Free Configuration Space of a Kirchhoff Elastic Rod is Path-connected. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2958–2964. IEEE. 9

[11] Bretl, T. and McCarthy, Z. (2013). Equilibrium Configurations of a Kirchhoff Elastic Rod under Quasi-static Manipulation. In *Algorithmic Foundations of Robotics X*, pages 71–87. Springer, Berlin, Heidelberg. 9

[12] Bretl, T. and McCarthy, Z. (2014). Quasi-static Manipulation of a Kirchhoff Elastic Rod Based on a Geometric Analysis of Equilibrium Configurations. *The International Journal of Robotics Research*, 33(1):48–68. 5, 9, 10, 18, 114

[13] Bryson, C. E. and Rucker, D. C. (2014). Toward Parallel Continuum Manipulators. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 778–785. IEEE. x, 2, 5, 11, 20, 21, 22, 35, 129

[14] Buckingham, R. O. and Graham, A. C. (2010). Dexterous Manipulators for Nuclear Inspection and Maintenance — Case Study. In *2010 1st International Conference on Applied Robotics for the Power Industry (CARPI 2010)*, pages 1–6. IEEE. 3

[15] Burgner, J., Rucker, D. C., Gilbert, H. B., Swaney, P. J., Russell, P. T., Weaver, K. D., and Webster, R. J. (2014). A Telerobotic System for Transnasal Surgery. *IEEE/ASME Transactions on Mechatronics*, 19(3):996–1006. 7, 90

[16] Burgner-Kahrs, J. (2015). Task-specific Design of Tubular Continuum Robots for Surgical Applications. In *Soft Robotics*, pages 222–230. Springer Berlin Heidelberg, Berlin, Heidelberg. 1

[17] Burgner-Kahrs, J., Rucker, D. C., and Choset, H. (2015). Continuum Robots for Medical Applications: A Survey. *IEEE Transactions on Robotics*, 31(6):1261–1280. 1

[18] Camarillo, D., Milne, C., Carlson, C., Zinn, M., and Salisbury, J. (2008). Mechanics Modeling of Tendon-Driven Continuum Manipulators. *IEEE Transactions on Robotics*, 24(6):1262–1273. 2

[19] Cardoso, J. R. and Leite, F. S. (2010). Exponentials of Skew-symmetric Matrices and Logarithms of Orthogonal Matrices. *Journal of Computational and Applied Mathematics*, 233(11):2867–2875. 11

[20] Celaya, E. A. and Jos, J. (2013). BDF-α : A Multistep Method with Numerical Damping Control. *Universal Journal of Computational Mathematics*, 1(3):96–108. 44

[21] Chajes, A. (1974). *Principles of Structural Stability Theory*. Prentice-Hall. 118

[22] Chirikjian, G. S. (2012). Variational Calculus on Lie Groups. In *Stochastic Models, Information Theory, and Lie Groups, Volume 2*, pages 129–154. Birkhäuser Boston, Boston. 121

[23] Choset, H., Zenati, M., Ota, T., Degani, A., Schwartzman, D., Zubiate, B., and Wright, C. (2011). Enabling Medical Robotics for the Next Generation of Minimally Invasive Procedures: Minimally Invasive Cardiac Surgery with Single Port Access. In *Surgical Robotics: Systems Applications and Visions*, pages 257–270. Springer US, Boston, MA. 3

[24] Curtiss, C. F. and Hirschfelder, J. O. (1952). Integration of Stiff Equations. *Proceedings of the National Academy of Sciences of the United States of America*, 38(3):235–43. 90

[25] Ding, J., Goldman, R. E., Xu, K., Allen, P. K., Fowler, D. L., and Simaan, N. (2013). Design and Coordination Kinematics of an Insertable Robotic Effectors Platform for Single-Port Access Surgery. *IEEE/ASME Transactions on Mechatronics*, 18(5):1612–1624. 2

[26] Duan, Y., Li, D., and Pai, P. F. (2013). Geometrically Exact Physics-based Modeling and Computer Animation of Highly Flexible 1D Mechanical Systems. *Graphical Models*, 75(2):56–68. 5

[27] Dupont, P., Lock, J., Itkowitz, B., and Butler, E. (2010). Design and Control of Concentric-Tube Robots. *IEEE Transactions on Robotics*, 26(2):209–225. 1, 4, 7, 8, 96

[28] Falk, V. (2002). Manual Control and Tracking–a Human Factor Analysis Relevant for Beating Heart Surgery. *The Annals of thoracic surgery*, 74(2):624–8. 21

[29] Gatti-Bono, C. and Perkins, N. (2002). Physical and Numerical Modelling of the Dynamic Behavior of a Fly Line. *Journal of Sound and Vibration*, 255(3):555–577. 6

[30] Gibson, S. F. F. and Mirtich, B. (1997). A Survey of Deformable Modeling in Computer Graphics. *Merl - a Mitsubishi Electric Research Laboratory*, pages 1–31. 5

[31] Gilbert, H. B., Hendrick, R. J., and Webster III, R. J. (2016a). Elastic Stability of Concentric Tube Robots: A Stability Measure and Design Test. *IEEE Transactions on Robotics*, 32(1):20–35. 8

[32] Gilbert, H. B., Rucker, D. C., and Webster III, R. J. (2016b). Concentric Tube Robots: The State of the Art and Future Directions. In Inaba, M., , and Corke, P., editors, *Robotics Research*, pages 253–269. Springer International Publishing. ix, 1, 2

[33] Gilbert, H. B., Webster, R. J., and III (2016c). Rapid, Reliable Shape Setting of Superelastic Nitinol for Prototyping Robots. *IEEE Robotics and Automation Letters*, 1(1):98–105. 1

[34] Goldberg, D. (1991). What Every Computer Scientist Should Know about Floating-point Arithmetic. *ACM Computing Surveys*, 23(1):5–48. 48

[35] Gravagne, I. A., Rahn, C. D., and Walker, I. D. (2003). Large Deflection Dynamics and Control for Planar Continuum Robots. *IEEE/ASME Transactions on Mechatronics*, 8(2):299–307. 4

[36] Grissom, M. D., Chitrakaran, V., Dienno, D., Csencits, M., Pritts, M., Jones, B., McMahan, W., Dawson, D., Rahn, C., and Walker, I. (2006). Design and Experimental Testing of the OctArm Soft Robot Manipulator. In Gerhart, G. R., Shoemaker, C. M., and Gage, D. W., editors, *Proceedings of SPIE - The International Society for Optical Engineering*, volume 6230, page 62301F. International Society for Optics and Photonics. 3

[37] Guennebaud, G., Jacob, B., and Others (2010). Eigen v3. http://eigen.tuxfamily.org. 32, 90

[38] Ha, J., Park, F. C., and Dupont, P. E. (2014). Achieving Elastic Stability of Concentric Tube Robots through Optimization of Tube Precurvature. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 864–870. IEEE. 1, 8

[39] Ha, J., Park, F. C., and Dupont, P. E. (2016). Elastic Stability of Concentric Tube Robots Subject to External Loads. *IEEE Transactions on Biomedical Engineering*, 63(6):1116–1128. 8, 9

[40] Hadap, S. (2006). Oriented Strands: Dynamics of Stiff Multi-body System. *Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 91–100. 5

[41] Hartenberg, R. S. and Denavit, J. (1955). A Kinematic Notation for Lower Pair Mechanisms Based on Matrices. *Journal of Applied Mechanics*, 77(2):215–221. 4

[42] Healey, T. J. and Mehta, P. G. (2005). Straightforward Computation of Spatial Equilibria of Geometrically Exact Cosserat Rods. *International Journal of Bifurcation*

and Chaos, 15(03):949–965. 8

[43] Hendrick, R. J., Gilbert, H. B., and Webster, R. J. (2015). Designing Snap-free Concentric Tube Robots: A Local Bifurcation Approach. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2256–2263. IEEE. 1, 8

[44] Hoffman, K. A. (2004). Methods for Determining Stability in Continuum Elastic-Rod Models of DNA. *Philosophical Transactions: Mathematical, Physical and Engineering Sciences*, 362:1301–1315. 8

[45] Holm, D. D., Schmah, T., and Stoica, C. (2009). *Geometric Mechanics and Symmetry: from Finite to Infinite Dimensions*. Oxford University Press. 9, 119, 121, 122, 123

[46] Holsapple, R., Venkataraman, R., and Doman, D. (2003). A Modified Simple Shooting Method for Solving Two-point Boundary-value Problems. *IEEE Aerospace Conference Proceedings*, 6(0):2783–2790. 49

[47] Hopkins, J. B., Rivera, J., Kim, C., and Krishnan, G. (2015). Synthesis and Analysis of Soft Parallel Robots Comprised of Active Constraints. *Journal of Mechanisms and Robotics*, 7(1):011002. 2, 9, 113

[48] Hošovský, A., Piteľ, J., Židek, K., Tóthová, M., Sárosi, J., and Cveticanin, L. (2016). Dynamic Characterization and Simulation of Two-link Soft Robot Arm with Pneumatic Muscles. *Mechanism and Machine Theory*, 103:98–116. 76

[49] Hull, D. G. (2003). *Optimal Control Theory for Applications*. Mechanical Engineering Series. Springer New York, New York, NY. 102, 103, 104, 105

[50] Jo, J.-W. and Prussing, J. E. (2000). Procedure for Applying Second-Order Conditions in Optimal Control Problems. *Journal of Guidance, Control, and Dynamics*, 23(2):241–250. 104

[51] Katzschmann, R. K., de Maille, A., Dorhout, D. L., and Rus, D. (2016). Cyclic Hydraulic Actuation for Soft Robotic Devices. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3048–3055. IEEE. 76

[52] Kazanzides, P., Chen, Z., Deguet, A., Fischer, G. S., Taylor, R. H., and DiMaio, S. P. (2014). An Open-source Research Kit for the da Vinci® Surgical System. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6434–6439. IEEE.

21

[53] Lan, C.-C. and Lee, K.-M. (2006). Generalized Shooting Method for Analysing Compliant Mechanisms with Curved Members. *Journal of Mechanical Design*, 128(July 2006):765–775. 6

[54] Lan, C. C., Lee, K. M., and Liou, J. H. (2009). Dynamics of Highly Elastic Mechanisms Using the Generalized Multiple Shooting Method: Simulations and Experiments. *Mechanism and Machine Theory*, 44(12):2164–2178. 6

[55] Lang, H., Linn, J., and Arnold, M. (2011). Multi-body Dynamics Simulation of Geometrically Exact Cosserat Rods. *Multibody System Dynamics*, 25(3):285–312. 5

[56] Lazarus, A., Miller, J., and Reis, P. (2013). Continuation of Equilibria and Stability of Slender Elastic Rods Using an Asymptotic Numerical Method. *Journal of the Mechanics and Physics of Solids*, 61(8):1712–1736. 8

[57] Lee, J., Kim, J., Lee, K.-K., Hyung, S., Kim, Y.-J., Kwon, W., Roh, K., and Choi, J.-Y. (2014). Modeling and Control of Robotic Surgical Platform for Single-port Access Surgery. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3489–3495. IEEE. 3

[58] Li, C. and Rahn, C. D. (2002). Design of Continuous Backbone, Cable-Driven Robots. *Journal of Mechanical Design*, 124(2):265. 8

[59] Linn, J., Lang, H., and Tuganov, A. (2012). Geometrically Exact Cosserat Rods with Kelvin-Voigt Type Viscous Damping. *Mechanical Sciences*, 4(1):79–96. 43, 87

[60] Maddocks, J. (1984). Stability of Nonlinearly Elastic Rods. *Archive for Rational Mechanics and Analysis*, 85(4):311–354. 8

[61] Mahl, T., Hildebrandt, A., and Sawodny, O. (2014). A Variable Curvature Continuum Kinematics for Kinematic Control of the Bionic Handling Assistant. *IEEE Transactions on Robotics*, 30(4):935–949. ix, 2, 3

[62] Mahony, R., Kumar, V., and Corke, P. (2012). Multirotor Aerial Vehicles: Modeling, Estimation, and Control of Quadrotor. *IEEE Robotics & Automation Magazine*, 19(3):20–32. 11

[63] Majidi, C., O'Reilly, O. M., and Williams, J. A. (2013). Bifurcations and Instability in the Adhesion of Intrinsically Curved Rods. *Mechanics Research Communications*, 49:13–16. 8

[64] Matthews, D. and Bretl, T. (2012). Experiments in Quasi-static Manipulation of a Planar Elastic Rod. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5420–5427. IEEE. 9

[65] McMahan, W., Chitrakaran, V., Csencsits, M., Dawson, D., Walker, I., Jones, B., Pritts, M., Dienno, D., Grissom, M., and Rahn, C. (2006). Field Trials and Testing of the OctArm Continuum Manipulator. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 2336–2341. IEEE. ix, 2, 3

[66] Mehling, J., Diftler, M., Chu, M., and Valvo, M. (2006). A Minimally Invasive Tendril Robot for In-Space Inspection. In *The First IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics, 2006. BioRob 2006.*, pages 690–695. IEEE. 3

[67] Mochiyama, H., Kinoshita, A., and Takasu, R. (2013). Impulse Force Generator Based on Snap-through Buckling of Robotic Closed Elastica: Analysis by Quasi-Static Shape Transition Simulation. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4583–4589. IEEE. 8

[68] Morimoto, T. K. and Okamura, A. M. (2016). Design of 3-D Printed Concentric Tube Robots. *IEEE Transactions on Robotics*, 32(6):1419–1430. 1

[69] Murray, R. M., Li, Z., and Sastry, S. (1994). *A Mathematical Introduction to Robotic Manipulation*. CRC Press. 11, 84, 119, 120

[70] Nizette, M. and Goriely, A. (1999). Towards a Classification of Euler-Kirchhoff Filaments. *Journal of Mathematical Physics*, 40(6):2830–2866. 31

[71] Nocedal, J. and Wright, S. J. (2006). *Numerical Optimization*. Springer. 27

[72] Oliver-Butler, K., Epps, Z. H., and Rucker, D. C. (2017). Concentric Agonist-Antagonist Robots for Minimally Invasive Surgeries. In Webster, R. J. and Fei, B., editors, *Proc. SPIE 10135, Medical Imaging 2017: Image-Guided Procedures, Robotic Interventions, and Modeling*, volume 10135, page 1013511. International Society for Optics and Photonics.

ix, 2

[73] Oliver-Butler, K., Till, J., and Rucker, C. (2019). Continuum Robot Stiffness Under External Loads and Prescribed Tendon Displacements. *IEEE Transactions on Robotics*, In Press:1–17. 2, 39, 67, 70

[74] Onal, C. D., Chen, X., Whitesides, G. M., and Rus, D. (2017). Soft Mobile Robots with On-Board Chemical Pressure Generation. In *Robotics Research*, pages 525–540. Springer, Cham. 76

[75] O'Reilly, O. M. and Peters, D. M. (2011). Nonlinear Stability Criteria for Tree-like Structures Composed of Branched Elastic Rods. *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, 468(2137). 8, 9

[76] Orekhov, A. L., Aloi, V. A., and Rucker, D. C. (2017). Modeling Parallel Continuum Robots with General Intermediate Constraints. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6142–6149. IEEE. 2

[77] Orekhov, A. L., Black, C. B., Till, J., Chung, S., and Rucker, D. C. (2016). Analysis and Validation of a Teleoperated Surgical Parallel Continuum Manipulator. *IEEE Robotics and Automation Letters*, 1(2):828–835. ix, 2, 20, 36, 37

[78] Orekhov, A. L., Bryson, C. E., Till, J., Chung, S., and Rucker, D. C. (2015). A Surgical Parallel Continuum Manipulator with a Cable-driven Grasper. In *2015 37th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pages 5264–5267. IEEE. 2, 20, 36

[79] Polygerinos, P., Correll, N., Morin, S. A., Mosadegh, B., Onal, C. D., Petersen, K., Cianchetti, M., Tolley, M. T., and Shepherd, R. F. (2017). Soft Robotics: Review of Fluid-Driven Intrinsically Soft Devices; Manufacturing, Sensing, Control, and Applications in Human-Robot Interaction. *Advanced Engineering Materials*, 19(12). 76

[80] Ponten, R. (2017). *Design of a Robotic Instrument Manipulator for Endoscopic Deployment*. PhD thesis, University of Tennessee, Knoxville. 2

[81] Ponten, R., Black, C. B., Russ, A. J., and Rucker, D. C. (2017). Analysis of a Concentric-Tube Robot Design and Feasibility for Endoscopic Deployment. In Webster, R. J. and Fei, B., editors, *Proc. SPIE 10135, Medical Imaging 2017: Image-Guided Procedures,*

Robotic Interventions, and Modeling., volume 10135, page 1013514. International Society for Optics and Photonics. 2

[82] Prussing, J. E. and Sandrik, S. L. (2005). Second-Order Necessary Conditions and Sufficient Conditions Applied to Continuous-Thrust Trajectories. *Journal of Guidance, Control, and Dynamics*, 28(4):812–816. 104

[83] Renda, F., Boyer, F., Dias, J., and Seneviratne, L. (2017). Discrete Cosserat Approach for Multi-Section Soft Robots Dynamics. *arXiv:1702.03660*. 5

[84] Renda, F., Cacucciolo, V., Dias, J., and Seneviratne, L. (2016). Discrete Cosserat Approach for Soft Robot Dynamics: A New Piece-wise Constant Strain Model with Torsion and Shears. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5495–5502. IEEE. 5

[85] Renda, F., Cianchetti, M., Giorelli, M., Arienti, A., and Laschi, C. (2012). A 3D Steady-State Model of a Tendon-Driven Continuum Soft Manipulator Inspired by the Octopus Arm. *Bioinspiration & Biomimetics*, 7(2):025006. 4, 5

[86] Renda, F., Giorelli, M., Calisti, M., Cianchetti, M., and Laschi, C. (2014). Dynamic Model of a Multibending Soft Robot Arm Driven by Cables. *IEEE Transactions on Robotics*, 30(5):1109–1122. 5

[87] Riojas, K. E., Hendrick, R. J., and Webster, R. J. (2018). Can Elastic Instability Be Beneficial in Concentric Tube Robots? *IEEE Robotics and Automation Letters*, 3(3):1624–1630. xiii, 8, 80, 89, 91, 95

[88] Rivera, J. A. and Kim, C. J. (2014). Spatial Parallel Soft Robotic Architectures. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 548–553. IEEE. 2, 9, 113

[89] Rivera-Serrano, C. M., Johnson, P., Zubiate, B., Kuenzler, R., Choset, H., Zenati, M., Tully, S., and Duvvuri, U. (2012). A Transoral Highly Flexible Robot. *The Laryngoscope*, 122(5):1067–1071. 3

[90] Robinson, G. and Davies, J. (1999). Continuum Robots - A State of the Art. In *Proceedings 1999 IEEE International Conference on Robotics and Automation*, volume 4, pages 2849–2854. IEEE. 1

[91] Rone, W. S. and Ben-Tzvi, P. (2014). Continuum Robot Dynamics Utilizing the Principle of Virtual Power. *IEEE Transactions on Robotics*, 30(1):275–287. 5

[92] Rucker, C. (2018). Integrating Rotations using Non-Unit Quaternions. *IEEE Robotics and Automation Letters*, 3(4):2979–2986. 18, 89

[93] Rucker, D. C. (2011). *The Mechanics of Continuum Robots: Model-based Sensing and Control*. PhD thesis, Vanderbilt University. ii, 5

[94] Rucker, D. C., Jones, B. A., and Webster III, R. J. (2010a). A Geometrically Exact Model for Externally Loaded Concentric-Tube Continuum Robots. *IEEE Transactions on Robotics*, 26(5):769–780. 4, 5, 7, 37, 82

[95] Rucker, D. C. and Webster, R. J. (2011). Computing Jacobians and Compliance Matrices for Externally Loaded Continuum Robots. In *2011 IEEE International Conference on Robotics and Automation*, pages 945–950. IEEE. 7, 106

[96] Rucker, D. C., Webster, R. J., Chirikjian, G. S., Cowan, N. J., and Cowan, N. J. (2010b). Equilibrium Conformations of Concentric-tube Continuum Robots. *The International Journal of Robotics Research*, 29(10):1263–1280. 8

[97] Rucker, D. C. and Webster III, R. J. (2011). Statics and Dynamics of Continuum Robots With General Tendon Routing and External Loading. *IEEE Transactions on Robotics*, 27(6):1033–1044. ix, 2, 4, 5, 7, 41, 62, 67

[98] Rus, D. and Tolley, M. T. (2015). Design, Fabrication and Control of Soft Robots. *Nature*, 521(7553):467–475. 4

[99] Shaikh, S. N. and Thompson, C. C. (2010). Natural Orifice Translumenal Surgery: Flexible Platform Review. *World Journal of Gastrointestinal Surgery*, 2(6):210–6. 3

[100] Shampine, L. F. and Reichelt, M. W. (1997). The MATLAB ODE Suite. *SIAM Journal on Scientific Computing*, 18(1):1–22. 18

[101] Shi, Y., Borovik, A. E., and Hearst, J. E. (1995). Elastic Rod Model Incorporating Shear and Extension, Generalized Nonlinear Schrödinger Equations, and Novel Closed-form Solutions for Supercoiled DNA. *The Journal of Chemical Physics*, 103(8):3166–3183. 31

[102] Shi, Y. and Hearst, J. E. (1994). The Kirchhoff Elastic Rod, the Nonlinear Schrödinger Equation, and DNA Supercoiling. *The Journal of Chemical Physics*, 101(6):5186–5200. 31

[103] Simaan, N., Xu, K., Kapoor, A., Wei, W., Kazanzides, P., Flint, P., and Taylor, R. (2009). Design and Integration of a Telerobotic System for Minimally Invasive Surgery of the Throat. *The International Journal of Robotics Research*, 28(9):1134–1153. 2

[104] Spillmann, J. and Teschner, M. (2007). CORDE: Cosserat Rod Elements for the Dynamic Simulation of One-Dimensional Elastic Objects. In *Eurographics/SIGGRAPH Symposium on Computer Animation*. 4, 5, 8

[105] Steigmann, D. J. and Faulkner, M. G. (1993). Variational Theory for Spatial Rods. *Journal of Elasticity*, 33(1):1–26. 8

[106] Sueda, S., Jones, G. L., Levin, D. I. W., and Pai, D. K. (2011). Large-scale Dynamic Simulation of Highly Constrained Strands. *ACM Transactions on Graphics*, 30(4):1. 5

[107] Tang, W., Lagadec, P., Gould, D., Wan, T. R., Zhai, J., and How, T. (2010). A Realistic Elastic Rod Model for Real-time Simulation of Minimally Invasive Vascular Interventions. *Visual Computer*, 26(9):1157–1165. 5

[108] Tang, W., Wan, T. R., Gould, D. A., How, T., and Nigel, J. W. (2012). A Stable and Real-Time Nonlinear Elastic Approach to Simulating Guidewire and Catheter Insertions Based on Cosserat Rod. *IEEE Transactions on Biomedical Engineering*, 59(8):2211–2218. 5

[109] Till, J., Aloi, V., and Rucker, C. (2019a). Real-Time Dynamics of Soft and Continuum Robots based on Cosserat-Rod Models. *International Journal of Robotics Research*, In Press. 8, 19, 39, 50, 52

[110] Till, J., Bryson, C. E., Chung, S., Orekhov, A., and Rucker, D. C. (2015). Efficient Computation of Multiple Coupled Cosserat Rod Models for Real-Time Simulation and Control of Parallel Continuum Manipulators. In *Proc. IEEE Conference on Robotics and Automation*, pages 5067–5074, Seattle, Washington. x, 20, 22, 30, 129

[111] Till, J., Riojas, K. E., Webster, R. J., and Rucker, C. (2019b). A Dynamic Model for Concentric Tube Robots. *IEEE Transactions on Robotics*, Submitted. 8, 39, 80

[112] Till, J. and Rucker, D. C. (2017a). Elastic Rod Dynamics: Validation of a Real-Time Implicit Approach. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3013–3019, Vancouver, Canada. IEEE. 39, 52, 90, 94

[113] Till, J. and Rucker, D. C. (2017b). Elastic Stability of Cosserat Rods and Parallel Continuum Robots. *IEEE Transactions on Robotics*, 33(3):718–733. 101

[114] Tobias, I., Swigon, D., and Coleman, B. D. (2000). Elastic Stability of DNA Configurations. I. General Theory. *Physical Review E*, 61(1):747–58. 8

[115] Trivedi, D., Lotfi, A., and Rahn, C. (2008). Geometrically Exact Models for Soft Robotic Manipulators. *IEEE Transactions on Robotics*, 24(4):773–780. 4

[116] Tunay, I. (2013). Spatial Continuum Models of Rods Undergoing Large Deformation and Inflation. *IEEE Transactions on Robotics*, 29(2):297–307. 5

[117] Walker, I. D. (2013). Continuous Backbone "Continuum" Robot Manipulators. *ISRN Robotics*, 2013. 1, 4

[118] Webster, R., Okamura, A., and Cowan, N. (2006). Toward Active Cannulas: Miniature Snake-Like Surgical Robots. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2857–2863. IEEE. 1

[119] Webster, R., Romano, J., and Cowan, N. (2009). Mechanics of Precurved-Tube Continuum Robots. *IEEE Transactions on Robotics*, 25(1):67–78. 8

[120] Webster, R. J. (2007). *Design and Mechanics of Continuum Robots for Surgery*. Dissertation, The Johns Hopkins University. 8

[121] Webster, R. J. and Jones, B. A. (2010). Design and Kinematic Modeling of Constant Curvature Continuum Robots: A Review. *The International Journal of Robotics Research*, 29(13):1661–1683. 1, 4

[122] Xu, K. and Simaan, N. (2008). An Investigation of the Intrinsic Force Sensing Capabilities of Continuum Robots. *IEEE Transactions on Robotics*, 24(3):576–587. 36

[123] Xu, R., Asadian, A., Naidu, A. S., and Patel, R. V. (2013). Position Control of Concentric-Tube Continuum Robots Using a Modified Jacobian-Based Approach. In *2013 IEEE International Conference on Robotics and Automation*, pages 5813–5818. IEEE. 7

[124] Yoon, H.-S. and Yi, B.-J. (2009). A 4-DOF Flexible Continuum Robot Using a Spring Backbone. In *2009 International Conference on Mechatronics and Automation*, pages 1249–1254. IEEE. 113

[125] Zadnq, G. (2001). Linear and Non-Linear Superelasticity in NiTi. *MRS Shape Memory Materials*, 683(9):201–209. 37

[126] Ziegler, H. (1977). *Principles of Structural Stability*. Birkhäuser Basel, Basel. 120

Appendices

A Static Rod IVP Solution in MATLAB

```
function RodIVP %Section 2.1.2
    %Independent Parameters
    E = 200e9;         %Young's modulus
    G = 80e9;          %Shear modulus
    r = 0.001;         %Cross-sectional radius
    rho = 8000;        %Density
    g = [9.81; 0; 0]; %Gravitational acceleration
    L = 0.5;           %Length (before strain)

    %Dependent Parameters
    A = pi*r^2;   %Cross-sectional area
    I = pi*r^4/4; %Area moment of inertia
    J = 2*I;      %Polar moment of inertia

    Kse = diag([G*A, G*A, E*A]); %Stiffness matrices, Equation (2.3)
    Kbt = diag([E*I, E*I, G*J]);

    %Measured base force and moment
    n0 = [0; 1; 0];
    m0 = [0; 0; 0];

    %Arbitrary base frame assignment
    p0 = [0;0;0];
    R0 = eye(3);

    %Numerical Integration
    y0 = [p0; reshape(R0,9,1); n0; m0]; %Combine states into single state vector
    [s,y] = ode45(@RodODE,[0 L],y0);    %Solve IVP with numerical integration

    %Visualization
    plot3( y(:,1), y(:,2), y(:,3) )
    axis([-L/2 L/2  -L/2 L/2  0 L])
    daspect([1 1 1])
    grid on
    title('Rod IVP Solution')
    xlabel('x (m)')
    ylabel('y (m)')
    zlabel('z (m)')

    %Subfunctions
    function ys = RodODE(s,y) %Equation (2.4)
        %Unpack state vector
        R = reshape(y(4:12),3,3);
        n = y(13:15);
        m = y(16:18);

        %Constitutive equation
        v = Kse^-1*R.'*n + [0;0;1];
        u = Kbt^-1*R.'*m;

        %Static Cosserat rod equations - system of nonlinear ODEs
        ps = R*v;
        Rs = R*hat(u);
        ns = -rho*A*g;
        ms = -hat(ps)*n;

        %Pack state vector derivative
        ys = [ps; reshape(Rs,9,1); ns; ms];
    end

    function skew_symmetric_matrix = hat(y) %Equation (1.1)
        skew_symmetric_matrix = [  0   -y(3)  y(2) ;
                                 y(3)   0   -y(1) ;
                                -y(2)  y(1)   0  ];
    end
end
```

B Static Rod BVP Solution in MATLAB

```
function RodBVP %Section 2.1.2
    %Parameters
    E = 200e9;
    G = 80e9;
    r = 0.001;
    rho = 8000;
    g = [9.81; 0; 0];
    L = 0.5;
    A = pi*r^2;
    I = pi*r^4/4;
    J = 2*I;
    Kse = diag([G*A, G*A, E*A]);
    Kbt = diag([E*I, E*I, G*J]);

    %Boundary Conditions
    p0 = [0;0;0];
    R0 = eye(3);
    pL = [0; -0.1*L; 0.8*L];
    RL = eye(3);

    %Main Simulation
    init_guess = zeros(6,1);
    global y; %Forward declaration for future scoping rule changes
    fsolve(@RodShootingMethod, init_guess); %Use convex optimization to solve ICs

    %Visualization
    plot3(y(:,1),y(:,2),y(:,3));  title('Rod BVP Solution');  axis([-L/2 L/2 -L/2 L/2 0 L])
    grid on;  daspect([1 1 1]);  xlabel('x (m)');  ylabel('y (m)');  zlabel('z (m)');

    %Subfunctions
    function residual = RodShootingMethod(guess) %Optimization objective function
        n0 = guess(1:3);                    %Update guessed initial conditions
        m0 = guess(4:6);
        y0 = [p0; reshape(R0,9,1); n0; m0];

        [s,y] = ode45(@RodODE,[0 L],y0);  %Numerically solve the resulting IVP

        pL_shot = y(end,1:3)';              %Calculate distal constraint violation
        RL_shot = reshape(y(end,4:12),3,3);
        position_error = pL_shot - pL;
        rotation_error = inv_hat( RL_shot'*RL - RL_shot*RL' ); %Equation (1.3)
        residual = [position_error; rotation_error];
    end

    function ys = RodODE(s,y) %Equation (2.4)
        R = reshape(y(4:12),3,3);
        n = y(13:15);
        m = y(16:18);

        v = Kse^-1*R.'*n + [0;0;1];
        u = Kbt^-1*R.'*m;

        ps = R*v;
        Rs = R*hat(u);
        ns = -rho*A*g;
        ms = -hat(ps)*n;

        ys = [ps; reshape(Rs,9,1); ns; ms];
    end

    function skew_symmetric_matrix = hat(y) %Equation (1.1)
        skew_symmetric_matrix = [  0   -y(3)  y(2) ;
                                 y(3)   0   -y(1) ;
                                -y(2)  y(1)   0  ];
    end

    function R3 = inv_hat(skew) %Equation (1.2)
        R3 = [skew(3,2); skew(1,3); skew(2,1)];
    end
end
```

C CSG Static BVP Solution in MATLAB

```
function ContinuumStewartGoughStaticBVP %Section 2.2.3
    %Properties
    E=200e9;  G=80e9;  rad=0.001;  rho=8000;  g=[0;0;-9.81];  ee_mass=0.1;  R0=eye(3);
    A=pi*rad^2;  I=pi*rad^4/4;  J=2*I;  Kse=diag([G*A,G*A,E*A]);  Kbt=diag([E*I,E*I,G*J]);

    %Given End Effector Pose and Wrench
    F = ee_mass*g;   M = [0;0;0];
    pE = [0; 0; 0.4];  bend = 10*pi/180;  RE = [ cos(bend)  0  sin(bend) ;
                                                    0       1      0     ;
                                                -sin(bend)  0  cos(bend)];
    %Hole Pattern (vector operations)
    scrib_R = 0.087;  alpha1 = 100*pi/180;  alpha2 = 120*pi/180 - alpha1;  i = 1:6;
    theta_B = (-alpha2 + (i-mod(i,2))*alpha2 + (i-1-mod(i-1,2))*alpha1)/2; %Equation (2.8)
    theta_E = (-alpha1 + (i-mod(i,2))*alpha1 + (i-1-mod(i-1,2))*alpha2)/2;
    p0 = scrib_R*[cos(theta_B); sin(theta_B); zeros(1,6)];
    r  = scrib_R*[cos(theta_E); sin(theta_E); zeros(1,6)];

    %Equations (1.1) and (1.2) (Written as anonymous functions for brevity)
    hat=@(y)[0,-y(3),y(2);y(3),0,-y(1);-y(2),y(1),0];  inv_hat_xy=@(y)[y(3,2);y(1,3)];

    %Main Simulation
    init_guess = [zeros(30,1); pE(3)*ones(6,1)]; %Initial guess; the form is given by (2.12)
    global p
    fsolve(@CSG_BVP_Function, init_guess); %Solve CSG BVP with shooting method

    %Visualization
    plot3( p{1}(1,:), p{1}(2,:), p{1}(3,:), 'b' );  hold on;
    plot3([p{6}(1,end) p{1}(1,end)],[p{6}(2,end) p{1}(2,end)],[p{6}(3,end) p{1}(3,end)],'r')
    for i = 2 : 6
        plot3(p{i}(1,:),p{i}(2,:),p{i}(3,:),'b');  ee_line = [p{i-1}(:,end), p{i}(:,end)];
        plot3(ee_line(1,:),ee_line(2,:),ee_line(3,:),'r')
    end
    hold off;  axis([-0.25 0.25  -0.25 0.25  0 0.5]);  daspect([1 1 1]);
    title('CSG BVP Solution'); grid on; xlabel('x (m)'); ylabel('y (m)'); zlabel('z (m)');

    %Subfunctions
    function E = CSG_BVP_Function(G) %Equation (2.13)
        EF = F;  EM = M; %Begin summing forces

        for i = 1 : 6 %Loop over each compliant robot link
            n0_range = 1+5*(i-1) : 3+5*(i-1);   m0_range = 4+5*(i-1) : 5*i;
            n0 = G(n0_range);  m0 = [G(m0_range); 0];  L = G(30+i);
            y0 = [p0(:,i); reshape(R0,9,1); n0; m0];

            [~,y] = ode45(@RodODE,[0 L],y0); %Numerically integrate this rod
            p{i} = y(:,1:3)'; %Store centerlines in cells for plotting

            pL_shot = y(end,1:3)';
            RL_shot = reshape(y(end,4:12),3,3);
            nL = y(end,13:15)';
            mL = y(end,16:18)';

            Ep = pL_shot - (pE + RE*r(:,i)); %Calculate geometric error
            ER = inv_hat_xy( RL_shot'*RE - RL_shot*RE' ); %Equation (1.3)
            geometric_error_range = 1+5*(i-1) : 5*i;
            E(geometric_error_range) = [Ep; ER];

            EF = EF - nL;    EM = EM - mL - cross( RE*r(:,i), nL ); %Continue summing forces
        end

        E(31:33) = EF;  E(34:36) = EM; %Force and moment summation are complete
    end

    function ys = RodODE(s,y) %Equation (2.4)
        R = reshape(y(4:12),3,3);  n = y(13:15);  m = y(16:18);
        v = Kse^-1*R.'*n + [0;0;1];  u = Kbt^-1*R.'*m;
        ps = R*v;  Rs = R*hat(u);  ns = -rho*A*g;  ms = -hat(ps)*n;
        ys = [ps; reshape(Rs,9,1); ns; ms];
    end
end
```

D Cantilever PDE: BDF-α and Shooting

```
function CantileverRod %Section 3.1.4
    hat=@(y)[0,-y(3),y(2);y(3),0,-y(1);-y(2),y(1),0];
    global p R j n m v u q w vt ut qt wt vh uh qh wh %Make vars available in whole program
    %Parameters
    L = 0.4;          %Length (before strain)
    N = 40;           %Spatial resolution
    E = 207e9;        %Young's modulus
    r = 0.0012;       %Cross-section radius
    rho = 8000;       %Density
    g = [-9.81;0;0]; %Gravity
    Bse = zeros(3);  %Material damping coefficients - shear and extension
    Bbt = zeros(3);  %Material damping coefficients - bending and torsion
    C = zeros(3);    %Square-law-drag damping coefficients
    dt = 0.002;       %Time step
    alpha = -0.48;    %BDF-alpha parameter
    STEPS = 50;       %Number of timesteps to completion
    vstar = @(s)[0;0;1]; %Value of v when static and absent loading
    ustar = @(s)[0;0;0]; %Precurvature
    %Boundary Conditions
    for i = 1 : STEPS
        p{i,1} = [0;0;0]; %Clamped base
        R{i,1} = eye(3);
        q{i,1} = [0;0;0];
        w{i,1} = [0;0;0];
    end
    nL = 0.2*g;  %Start with a weight hung at the tip
    mL = [0;0;0];

    %Dependent Parameter Calculations
    A = pi*r^2;                                  %Cross-sectional area
    J = diag([pi*r^4/4  pi*r^4/4  pi*r^4/2]);    %Inertia
    G = E/( 2*(1+0.3) );                         %Shear modulus
    Kse = diag([G*A, G*A, E*A]);                 %Stiffness matrix - shear and extension
    Kbt = diag([E*J(1,1), E*J(2,2), G*J(3,3)]); %Stiffness matrix - bending and torsion
    ds = L/(N-1);                                %Grid distance (before strain)
    c0 = (1.5 + alpha) / ( dt*(1+alpha) );       %BDF-α coefficients
    c1 = -2/dt;
    c2 = (0.5 + alpha) / ( dt*(1+alpha) );
    d1 = alpha / (1+alpha);

    %Main Simulation
    i = 1;
    fsolve(@staticIVP, zeros(6,1)); %Solve static BVP w/ shooting method
    applyStaticBDFalpha();
    visualize();
    nL = [0;0;0]; %Tip weight is released

    for i = 2 : STEPS
        fsolve(@dynamicIVP, [n{i-1,1}; m{i-1,1}]); %Solve semi-discretized PDE w/ shooting
        applyDynamicBDFalpha();
        visualize();
    end

    %Function Definitions
    function applyStaticBDFalpha() %Equation (3.4) for system at steady state
        for j = 1 : N-1
            vh{i+1,j} = (c1+c2)*v{i,j};
            uh{i+1,j} = (c1+c2)*u{i,j};
            qh{i+1,j} = [0;0;0];
            wh{i+1,j} = [0;0;0];
            q{i,j} = [0;0;0];
            w{i,j} = [0;0;0];
        end
    end

    function applyDynamicBDFalpha() %Equation (3.4)
        for j = 1 : N-1
            vh{i+1,j} = c1*v{i,j} + c2*v{i-1,j} + d1*vt{i,j};
            uh{i+1,j} = c1*u{i,j} + c2*u{i-1,j} + d1*ut{i,j};
            qh{i+1,j} = c1*q{i,j} + c2*q{i-1,j} + d1*qt{i,j};
```

```
71              wh{i+1,j} = c1*w{i,j} + c2*w{i-1,j} + d1*wt{i,j};
72          end
73      end
74
75      function E = staticIVP(G)
76          n{i,1} = G(1:3);
77          m{i,1} = G(4:6);
78
79          %Euler's method
80          for j = 1 : N-1
81              [ps, Rs, ns, ms, v{i,j}, u{i,j}] = staticODE(p{i,j},R{i,j},n{i,j},m{i,j});
82              p{i,j+1} = p{i,j} + ds*ps;
83              R{i,j+1} = R{i,j} + ds*Rs;
84              n{i,j+1} = n{i,j} + ds*ns;
85              m{i,j+1} = m{i,j} + ds*ms;
86          end
87          E = [ n{i,N} - nL;  m{i,N} - mL ];
88      end
89
90      function E = dynamicIVP(G)
91          n{i,1} = G(1:3);
92          m{i,1} = G(4:6);
93
94          %Euler's method
95          for j = 1 : N-1
96              [ps, Rs, ns, ms, qs, ws, v{i,j}, u{i,j}, ...
97                  vt{i,j}, ut{i,j}, qt{i,j}, wt{i,j}] = ...
98                  dynamicODE(p{i,j},R{i,j},n{i,j},m{i,j},q{i,j},w{i,j});
99              p{i,j+1} = p{i,j} + ds*ps;
100             R{i,j+1} = R{i,j} + ds*Rs;
101             n{i,j+1} = n{i,j} + ds*ns;
102             m{i,j+1} = m{i,j} + ds*ms;
103             q{i,j+1} = q{i,j} + ds*qs;
104             w{i,j+1} = w{i,j} + ds*ws;
105         end
106         E = [n{i,N} - nL;  m{i,N} - mL];
107     end
108
109     function [ps, Rs, ns, ms,  v, u] = staticODE(p,R,n,m) %Equation (2.4)
110         v = Kse\R'*n + vstar(ds*(j-1));
111         u = Kbt\R'*m + ustar(ds*(j-1));
112
113         ps = R*v;
114         Rs = R*hat(u);
115         ns = -rho*A*g;
116         ms = -hat(ps)*n;
117     end
118
119     function [ps,Rs,ns,ms,qs,ws, v,u,vt,ut,qt,wt] = dynamicODE(p,R,n,m,q,w) %Equation (3.5)
120         v = (Kse + c0*Bse)\(R'*n + Kse*vstar(ds*(j-1)) - Bse*vh{i,j});
121         u = (Kbt + c0*Bbt)\(R'*m + Kbt*ustar(ds*(j-1)) - Bbt*uh{i,j});
122         vt = c0*v + vh{i,j};
123         ut = c0*u + uh{i,j};
124         qt = c0*q + qh{i,j};
125         wt = c0*w + wh{i,j};
126         f = -R*C*q.*abs(q) + rho*A*g;
127
128         ps = R*v;
129         Rs = R*hat(u);
130         ns = rho*A*R*(hat(w)*q + qt) - f;
131         ms = rho*R*(hat(w)*J*w + J*wt) - hat(ps)*n;
132         qs = vt - hat(u)*q + hat(w)*v;
133         ws = ut - hat(u)*w;
134     end
135
136     function visualize()
137         for j = 1 : N,  x(j) = p{i,j}(1);  z(j) = p{i,j}(3);  end
138         plot(z,x);  axis([0 1.1*L  -0.55*L 0.55*L]);  daspect([1 1 1]);
139         title('Cantilever Rod');  xlabel('z (m)');  ylabel('x (m)');
140         grid on;  drawnow;  pause(0.05);
141     end
142 end
```

E Cantilever PDE: Implicit Midpoint and Shooting

```
function CantileverRod %Section 3.1.4
    hat=@(y)[0,-y(3),y(2);y(3),0,-y(1);-y(2),y(1),0]; global y z
    %Parameters
    L = 0.4;  E = 207e9;  r = 0.0012;  rho = 8000;  g = [-9.81;0;0];
    Bse = zeros(3);  Bbt = zeros(3);  C = zeros(3);  vstar = [0;0;1];
    N = 40;  dt = 0.002;  STEPS = 50;  M_tip = [0;0;0];  F_tip = 0.2*g; %Start with tip load
    %Boundary Conditions
    p0 = [0;0;0];  R0 = eye(3);  q0 = [0;0;0];  w0 = [0;0;0];
    %Dependent Parameter Calculations
    A = pi*r^2;  G = E/( 2*(1+0.3) );  ds = L/(N-1);
    Ixx = pi*r^4/4;  Iyy = Ixx;  Izz = 2*Ixx;  J = diag([Ixx  Iyy  Izz]);
    Kse = diag([G*A, G*A, E*A]);   Kbt = diag([E*Ixx, E*Iyy, G*Izz]);
    %Main Simulation
    i = 1;  G = fsolve(@getResidual, zeros(6,1)); %Solve static BVP
    z(:,N) = solveStaticConstitutiveLaw(y(:,N));
    ys(:,N) = f(y(:,end),zeros(24,1),z(:,end),zeros(6,1)); %Distal ys needed
    F_tip = [0;0;0]; %Tip weight is released
    for i = 2 : STEPS
        y_old = y;  ys_old = ys;  z_old = z;  visualize();
        G = fsolve(@getResidual, G); %Solve semi-discretized BVP
        ys(:,N) = implicitMidptODE(y(:,end),y_old(:,end),ys_old(:,end),z_old(:,end));
    end
    %Function Definitions
    function visualize()
        plot(y(3,:),y(1,:));  title('Cantilever Rod');  xlabel('z (m)');  ylabel('x (m)');
        axis([0 1.1*L  -0.55*L 0.55*L]);  grid on;  daspect([1 1 1]);  drawnow;
    end
    function E = getResidual(G)
        n0 = G(1:3);  m0 = G(4:6); %Reaction force and moment are guessed
        y(:,1) = [p0; reshape(R0,9,1); n0; m0; q0; w0];
        for j = 1 : N-1 %Euler Integration
            if i == 1 %First time step is static
                z(:,j) = solveStaticConstitutiveLaw(y(:,j));
                ys(:,j) = f(y(:,j),zeros(24,1),z(:,j),zeros(6,1));
            else      %Next time steps use PDE semi-discretization
                [ys(:,j),z(:,j)]=implicitMidptODE(y(:,j),y_old(:,j),ys_old(:,j),z_old(:,j));
            end
            y(:,j+1) = y(:,j) + ds*ys(:,j); %Euler's Method
        end
        nL = y(13:15,N);  mL = y(16:18,N);  E = [F_tip-nL;  M_tip-mL];
    end
    function [ys,z] = implicitMidptODE(y,y_old,ys_old,z_old) %Equation (3.7)
        z = solveDynamicConstitutiveLaw(y,y_old,z_old);
        ys = -ys_old + 2*f( (y+y_old)/2, (y-y_old)/dt, (z+z_old)/2, (z-z_old)/dt );
    end
    function z = solveStaticConstitutiveLaw(y)
        R = reshape(y(4:12),3,3);  n = y(13:15);  m = y(16:18);
        v = Kse\R'*n + vstar;  u = Kbt\R'*m;  z = [v;u];
    end
    function z = solveDynamicConstitutiveLaw(y,y_old,z_old) %Equation (3.8)
        Rbar = (reshape(y(4:12),3,3) + reshape(y_old(4:12),3,3))/2;
        nbar = (y(13:15) + y_old(13:15))/2;  mbar = (y(16:18) + y_old(16:18))/2;
        v_prev = z_old(1:3);  u_prev = z_old(4:6);
        v = (Kse/2+Bse/dt) \ ((-Kse/2+Bse/dt)*v_prev + Rbar'*nbar + Kse*vstar);
        u = (Kbt/2+Bbt/dt) \ ((-Kbt/2+Bbt/dt)*u_prev + Rbar'*mbar);
        z = [v;u];
    end
    function ys = f(y,yt,z,zt) %Equation (3.5)
        R = reshape(y(4:12),3,3);  n = y(13:15);  m = y(16:18);
        q = y(19:21);  w = y(22:24);  qt = yt(19:21);  wt = yt(22:24);
        v = z(1:3);  u = z(4:6);  vt = zt(1:3);  ut = zt(4:6);
        ps = R*v;
        Rs = R*hat(u);
        ns = R*(rho*A*(hat(w)*q + qt) + C*q.*abs(q)) - rho*A*g;
        ms = rho*R*(hat(w)*J*w + J*wt) - hat(ps)*n;
        qs = vt - hat(u)*q + hat(w)*v;
        ws = ut - hat(u)*w;
        ys = [ps; reshape(Rs,9,1); ns; ms; qs; ws];
    end
end
```

F Cantilever PDE: BDF-α and Midpoint Differences

```
function CantileverFiniteDifferenceSystem %Section 3.1.4
    hat=@(y)[0,-y(3),y(2);y(3),0,-y(1);-y(2),y(1),0];  M = 24;  global Yt
    %Parameters
    L = 0.4;  E = 207e9;  r = 0.0012;  rho = 8000;
    g = [-9.81;0;0];  C = zeros(3);  vstar = [0;0;1];
    M_tip = [0;0;0];  F_tip = 0.2*g; %Start with tip load
    N = 40;  dt = 0.002;  alpha = -0.48;  STEPS = 50;
    %Boundary Conditions
    p0 = [0;0;0];  R0 = eye(3);  q0 = [0;0;0];  w0 = [0;0;0];
    %Dependent Parameter Calculations
    A = pi*r^2;  G = E/( 2*(1+0.3) );  ds = L/(N-1);
    Ixx = pi*r^4/4;  Iyy = Ixx;  Izz = 2*Ixx;  J = diag([Ixx  Iyy  Izz]);
    Kse = diag([G*A, G*A, E*A]);   Kbt = diag([E*Ixx, E*Iyy, G*Izz]);
    ds = L/(N-1); %Distance between grid points
    c0 = (1.5 + alpha) / ( dt*(1+alpha) );  c1 = -2/dt;
    c2 = (0.5 + alpha) / ( dt*(1+alpha) );  d1 = alpha / (1+alpha);

    %Main simulation
    i = 1; Y = zeros(M*N,1); G = zeros(M*(N-1),1); %Guess the states which are not enforced
    G = fsolve(@finiteDifferenceResidual, G); %Solve static BVP
    Y_old = Y;  Y_older = Y;
    F_tip = [0;0;0]; %Tip weight is released
    for i = 2 : STEPS
        Yt_old = Yt;  Y_older = Y_old;  Y_old = Y;
        visualize();
        G = fsolve(@finiteDifferenceResidual, G); %Solve semi-discretized BVP
        %Note that dense Jacobian scales poorly, but OK for example.
    end

    %Subfunctions
    function E = finiteDifferenceResidual(G) %Equation (3.9)
        Y(1:12) = [p0; reshape(R0,9,1)]; %Strongly enforced initial pose
        Y(13:18) = G(1:6); %Guessed initial wrench
        Y(19:24) = [q0; w0]; %Strongly enforced initial twist
        Y(M+1:M*(N-1)) = G(7:6+M*(N-2)); %Guessed middle states
        Y(M*N-23 : M*N-12) = G(7+M*(N-2) : 18+M*(N-2)); %Guessed distal pose
        Y(M*N-11 : M*N-9) = F_tip; %Strongly enforced distal internal force
        Y(M*N-8 : M*N-6) = M_tip; %Strongly enforced distal internal moment
        Y(M*N-5 : M*N) = G(19+M*(N-2) : M*(N-1)); %Guessed distal twist
        if i == 1,  Yt = zeros(M*N,1);
        else,       Yt = c0*Y + c1*Y_old + c2*Y_older + d1*Yt_old;  end
        E = zeros(M*(N-1),1);
        for j = 1 : N-1      %Find the M*(N-1) finite difference errors
            left = 1 + (j-1)*M : j*M;  right = left+M;
            E(left) = f( 0.5*(Y(left)+Y(right)), 0.5*(Yt(left)+Yt(right)) ) ...
                    - (Y(right) - Y(left)) / ds;
        end
    end
    function ys = f(y,yt)
        R = reshape(y(4:12),3,3);  n = y(13:15);  m = y(16:18);
        q = y(19:21);  w = y(22:24);
        Rt = reshape(yt(4:12),3,3);  nt = yt(13:15);  mt = yt(16:18);
        qt = yt(19:21);  wt = yt(22:24);
        %Constitutive equation - material damping omitted for brevity
        v = Kse\R'*n+vstar;  u = Kbt\R'*m;
        vt = Kse\(Rt.'*n + R.'*nt);  ut = Kbt\(Rt.'*m + R.'*mt);
        %Rod State Derivatives
        ps = R*v;
        Rs = R*hat(u);
        ns = R*(rho*A*(hat(w)*q + qt) + C*q.*abs(q)) - rho*A*g;
        ms = rho*R*(hat(w)*J*w + J*wt) - hat(ps)*n;
        qs = vt - hat(u)*q + hat(w)*v;
        ws = ut - hat(u)*w;
        ys = [ps; reshape(Rs,9,1); ns; ms; qs; ws];
    end
    function visualize()
        plot(Y(3:M:end),Y(1:M:end));  title('Cantilever Rod');  xlabel('z (m)');
        ylabel('x (m)'); axis([0 1.1*L -0.55*L 0.55*L]); grid on; daspect([1 1 1]); drawnow
    end
end
```

Vita

John Till was born in Knoxville, Tennessee to parents David and Lynn. He was educated throughout grade school along with three siblings by his dedicated mother. From 2007-2010 he received training in mathematics, science, and the humanities at Chattanooga State Community College. He attended Tennessee Technological University to study mechanical engineering from 2010 to 2014, where he also haplessly stumbled his way into a minor in computer science. Upon completion of his B.S. degree, he entered the graduate program at the University of Tennessee, where he performed the work for this dissertation as a member of Caleb Rucker's REACH Lab.