UNIVERSITY OF CALIFORNIA SAN DIEGO

Modeling, Estimation, and Control for Continuum Robots in Unknown, Constrained Environments

A dissertation submitted in partial satisfaction of the
requirements for the degree of Doctor of Philosophy

in

Engineering Sciences (Mechanical Engineering)

by

Connor Watson

Committee in charge:

Professor Tania Morimoto, Chair
Professor Miroslav Krstic
Professor Michael Tolley
Professor Michael Yip

2023

The Dissertation of Connor Watson is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2023

DEDICATION

For mom.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I owe many thanks to many people who have helped me in completing this dissertation. First, I would like to thank the members of my committee. I would like to thank Professor Tolley for his insightful questions that have helped to improve the quality of this research as well as for giving me the opportunity to TA one of my first classes in the graduate program, which was a valuable experience I have carried with me for the rest of my time here. I would like to thank Professors Yip and Krstic for the many EFRI meetings that they attended with me. I have learned a lot from those meetings and believe that they have played an integral role in my development as a researcher.

Next, I would like to thank my advisor, Professor Morimoto. I will forever be thankful to her for giving me the chance to pursue graduate level research back in the Fall of 2018. Since that time, I have come to know her as an excellent mentor and researcher that I am very fortunate to have gotten the chance to learn from and work with. Without her help this work would not have been possible.

Finally, I would like to acknowledge all of the friends and family that have supported me in my efforts to pursue this research. To properly thank them all would take another 100 pages, but in order save paper, I will just say thank you all very much.

Chapter 2 is, in part, a reprint of material from C. Watson, R. Obregon, T.K. Morimoto, ”*Closed-Loop Position Control for Growing Robots Via Online Jacobian Corrections*.” IEEE Robotics and Automation Letters, vol. 6, no. 4, pp. 6820-6827, 2021. Chapter 3 is, in part, a reprint of material from C. Watson, T.K. Morimoto, ”*Learning Non-Parametric Models in Real Time via Online Generalized Product of Experts*.” IEEE Robotics and Automation Letters, vol. 7, no. 4, pp. 9326-9333, 2022. The dissertation author is the primary author of both of these papers.

Chapter 4 contains material that will be submitted for publication under the title ”*Disturbance Rejection for Cable-Driven Continuum Robots without Integrated Sensing*.” in 2023. The

dissertation author is the primary author of this work. Chapter 5 is, in part, a reprint of material from C. Watson, A. Nguyen, and T.K. Morimoto. ”*Image Segmentation for Continuum Robots from a Kinematic Prior*.” IEEE/RAS Int. Conf. on Robotics and Automation (ICRA), 2023. The dissertation author is the primary author of this work.

ABSTRACT OF THE DISSERTATION

Modeling, Estimation, and Control for Continuum Robots in Unknown, Constrained Environments

by

Connor Watson

Doctor of Philosophy in Engineering Sciences (Mechanical Engineering)

University of California San Diego, 2023

Professor Tania Morimoto, Chair

Continuum robots achieve high dexterity in constrained spaces by bending along their entire length. Their unique modes of actuation, small size, and inherent compliance make them well-suited for "keyhole" applications in delicate environments, such as minimally invasive surgery, where the robot must perform a dexterous task through a small opening without causing harm to its surroundings. The full potential of continuum robots has yet to be realized, however, as the advantages offered by the mechanics of these robots pose challenges for their modeling, estimation, and control. In particular, real world phenomena such as friction, material variability, and fabrication tolerances play a large role in determining the ultimate shape of a continuum robot, but are notoriously difficult to model analytically. Further, their small form factor and flexible nature make it difficult to integrate sensors that can directly measure their states. These problems are compounded when the robot comes into contact with unknown obstacles, which change the robot shape in ways that cannot be predicted *a priori*.

This work looks at three different algorithmic approaches to address these challenges. First, we consider how a low-parameter, numerically stable representation of the differential kinematics for a growing robot can be used together with a real time estimation technique to steer the robot through highly constrained environments. Second, we show how traditional Gaussian Process models can be adapted to produce high accuracy input-output maps for a wide range of continuum robot designs using data that is received sequentially over time. Finally, we develop a simplified dynamics model that facilitates the design of an observer for continuum robot control in the extreme case when the robot does not have any integrated sensing. All of our methods are validated through hardware experiments and their associated trade-offs are discussed. Overall, this work demonstrates how we can better design models and algorithms to extend the capabilities of continuum robots and further unlock their potential.

Chapter 1

Introduction

1.1 Motivation

Continuum robots are a class of robot that can change shape continuously along their lengths, achieving motions that often draw comparisons to animal physiology, such as elephant trunks, octopus tentacles or snake bodies [1] [2]. This unique feature makes them well suited to “keyhole” applications, where the robot would need to perform a dexterous task through a small opening or in a constrained environment. Many such applications have been proposed including search and rescue [3][4], maintenence of nuclear power plants [5], and machine inspection [6], but by far the most popular has been as tools used in minimally invasive surgical procedures [7][8] [9]. In some ways this is not surprising, given that “continuum” instruments like endoscopes, catheters, and guidewires have been used in medicine long before robotics [10]. The fact that rigid robot designs have already been shown to be successful in revolutionizing laparoscopic surgery outside of the laboratory [11] [12] further hints at the exciting possibilities of applying continuum robots in this area.

This great potential has made them the object of intense research over the past two decades, resulting in a vast literature proposing new continuum robot designs [13][8][14][3][15], mathematical modeling approaches [16] [17] [18] and control strategies [19] [20]. While progress has been significant, challenges remain in each of these areas that must be solved before continuum robots can have the same transformative effects on the real world that their rigid counterparts have had. More specifically, these challenges arise due to the small form factor and flexible nature of continuum robots. Natural flexibility makes the shape of the robot much more sensitive to factors like external loading, variations in material parameters, fabrication tolerances and more. While these issues could be mitigated by sensor feedback, physically incorporating sensors into a small, compliant continuum robot body is often a challenge on its own. Consequently, algorithms developed for predicting and controlling continuum robot motion must be designed to work with incomplete information and to be somewhat robust to the aforementioned uncertainties that effect the robot shape. The estimation and control challenges are exacerbated when the robot is constrained by an unknown environment and remains an unsolved problem. While there is no single, perfect algorithm for this task, this dissertation proposes a number of possible approaches to the problem of predicting and controlling continuum robots subjected to unknown forces from their environments, each of which has tradeoffs with respect to the others in aspects such as the amount of sensor information the algorithm requires, accuracy, and generality across robot designs.

1.2 Prior Work

1.3 Design, Actuation, and Sensing

There are a large number of continuum robot designs that have emerged in the past two decades [10]. While design encompasses all aspects of a continuum robot's construction including geometry and material composition, it is helpful to distinguish groups of related continuum robot designs by their primary means of actuation. This actuation may be either intrinsic–embedded into the body of the robot itself–or extrinsic–mechanically transmitted to the robot body from an external source. The robot's actuating principle determines how it will vary its curvature in space and is therefore vital to its functionality. To date, researchers have proposed continuum robot designs that are actuated by tendons [21], flexible auxiliary backbones [22], concentric tubes [14], magnets [23], or various soft structures [24]. No one mode of actuation has emerged as superior to the others for every application, as each has significant advantages and disadvantages that must be considered on a case-by-case basis. From the perspective of robot control, it is important to note that *none* of the proposed designs are ideal. The ideal actuator would allow the user to control the curvature of the robot everywhere along its backbone *independently*, thereby making it simple to command the robot to form arbitrary shapes. Instead, however, the current means of actuation only allow for changing the robot curvature over large *regions* of the robot backbone, sometimes in coupled and highly nonlinear ways.

Limited control authority is not the only physical limitation imposed on algorithm design for these robots, however. Due to their small form factor and natural compliance, it is often difficult to integrate sensors into the backbone of the robot to measure important aspects of the robot state. The ideal sensor would provide information, such as position and applied external loading, all along its length, i.e. in a distributed fashion. Arguably, something close to this can actually be achieved with Fiber Bragg Grating sensors [25], which measure strain in multiple dimensions all along the robot backbone. While use of these sensors has seen some limited exploration in the literature, they remain prohibitively expensive for widespread adoption and can be challenging to successfully integrate with some of the softer continuum robot designs. More recently, researchers have explored another means of distributed sensing for continuum robots using external cameras [26, 27, 28, 29]. In order for these approaches to be used in a control algorithm, however, there must be improved robustness to imperfect imaging conditions, increased speed for real-time performance, and quantified observability as function of the relative camera pose. Therefore, only point-wise measurements of the robot position and orientation, provided by a magnetic or electromagnetic tracking system, are typically available [30].

These two problems of under actuation and limited sensing add difficultly to the task of developing useful algorithms for continuum robots and necessarily inform the steps of modeling, estimation, and control design.

1.3.1 Modeling

Whether data-driven or analytically derived, some computational representation of continuum robot motion is necessary for control. Typically, it is helpful to distinguish between joint, configuration, and task spaces when discussing different robotics modeling approaches [16]. Unlike for rigid robots, there is a significant distinction between the joint and configuration spaces for a continuum robot. While the joint space is the set of all possible joint inputs to the robot, the configuration space is the set of all shapes that the robot centerline can take. This set of all configurations is a functional space and does not map uniquely onto joint space when external loads are applied to the robot body. The task space is defined by the operation that a continuum robot is supposed to perform, but is usually considered to be the tip position of the robot, its orientation, or both.

The simplest class of models for continuum robots that have been developed to date are constant curvature kinematic models [16]. These models map from configuration to task space assuming that the shape of the robot is well-described by a circular arc. Their advantages include that they can be straightforwardly derived by geometric reasoning, the approximation is reasonably accurate for many kinds of robots when not subjected to external loading, and that their computational cost is negligible. The primary downsides of these models are that they do not perform as well when the continuum robot is subjected to large external loads and that they do not address the joint-configuration space mapping at all.

The next class of models is based on a quasi-static assumption. There are two main approaches that make this assumption: (i) the constant curvature kineto-static models, which were first popularized in [31], and (ii) the Cosserat-rod models, which were first employed in [32]. Both of these approaches provide mappings from joint space to configuration space *and* configuration space to task space. The primary differences between the two can be thought of as a trade-off between accuracy and computational cost. While kineto-static approaches employ a constant curvature assumption and retain minimal computational requirements, Cosserat rod models of continuum robots define the shape of the robot as an oriented curve in space that must be determined via the solution of a system of ordinary differential equations (ODEs). This solution is expensive and makes using this model inside a real-time control loop challenging. The accuracy of these models is unmatched by any other analytical approach in the literature, however, and they provide an elegant framework for modeling how arbitrary, known loads applied to the robot body will affect its shape [18].

A still more complicated class of continuum robot models, that has only recently begun to receive attention in the literature, attempts to capture dynamic behavior of the robot. Again, there are two subcategories of models within this class, one that is based on a minimal parameterization of the robot shape [33][34] and the other based on Cosserat rods [17]. The first category of methods tends to pose the same accuracy-computation tradeoff as previously described and also tends to ignore any actuator dynamics, which may be non-trivial for cable-driven and pneumatically actuated robots. The Cosserat rod-based methods on the other hand readily accommodate very complicated actuator dynamics, including actuating cables with non-uniform tension along their length and pneumatic actuators driven with compressible fluids. Whereas in the quasi-static case these models required solving a system of ODEs, in the dynamic case, the motion of the continuum robot is described by the solution to a system of partial differential equations (PDEs) [17]. The computational and control-theoretic challenges associated with using these models are significant and continue to be actively investigated [35].

Aside from analytical models, many researchers have looked at how machine learning methods can be used to capture any of the mappings between joint space, configuration space, and task space assuming both quasi-static [36] [37] and dynamic [38] robot motion. Because these approaches develop models directly from data, they tend to offer high accuracy without significant computational cost. This makes them attractive for applications where high precision is important. The disadvantages to these approaches include the large amount of training data required, the lack of insight they afford a user, and the difficulties with establishing guarantees when using these models inside a control loop.

1.3.2 Estimation and Control

Most of the early literature regarding control for continuum robots is concerned with controlling the tip of the robot in free space [16]. With a reasonably accurate model of the robot kinematics, this amounts to regulating the joint positions of the robot such that the error between the current tip position and a desired value decreases via the solution of an inverse kinematics problem and can be done in open loop [39]. It is even possible to learn an inverse kinematic mapping directly for control depending on the robot and the representation of the tip pose used [40]. Additional information from sensors can be used for closed-loop control of the tip pose, or to unlock functionality such as dynamic control of the robot [41] or control over aspects of its shape [42] in the quasi-static case. When the robot comes in to contact with an obstacle, however, an additional estimation step must be included to determine how this interaction affects the robot shape and ultimately its tip position. While there are many features that distinguish these proposed approaches from one another, largely, they can be classified as being either analytical or data-driven methods.

The analytical methods to date assume that any disturbance is being applied only at the tip of the robot. This simplifying assumption, together with tip position measurements, makes it straightforward to estimate a wrench applied at the tip of the robot using either a full Cosserat rod model of the robot kinematics or a local model of the robot statics. This estimate can then be used in a stiffness or hybrid tip position-force controller [43] [44]. If tip position measurements are unavailable however, the problem becomes considerably more difficult, and has received less attention in the literature. Nonetheless, some researchers have shown that it is possible to do hybrid tip position-force control using joint feedback alone [45].

An alternative group of methods for solving this problem are more data driven in nature. They construct a model of the robot's Jacobian that is valid only in a small region of the robot's task space. This model is then updated online via optimization [19], [46], [47] or using an Extended Kalman Filter [48] using measurements of the robot's end-effector position. These approaches are not limited by an assumption that the disturbance is applied to the tip of the robot but suffer a drawback in that they require measurements of the robot tip position.

1.4 Contributions

The main contributions of this dissertation can be summarized as follows:

1. **A method for closed-loop position control of growing robots via online Jacobian**

corrections: This method is based on leveraging real-time position and orientation measurements of the robot tip in order to update a local model of the velocity kinematics of the robot in an online manner via corrective rotations and magnitude adjustments. This control strategy is the first method for autonomous position control of growing robots that does not require a map of its environment.

2. **A Gaussian Process-based online learning algorithm, which we call online generalized product of experts (OGPoE):** This algorithm offers a combination of benefits unavailable from other online learning algorithms, including the simultaneous online optimization of local model parameters, a clear and interpretable data partitioning strategy, and principled fusion of multiple model predictions without the need for joint training.

3. **A constant curvature dynamic model for cable-driven continuum robots:** This model is the simplest dynamic model for these robots to date. We use the model to formulate an observer with provable guarantees for state estimation and design a controller to reject low-frequency disturbances affecting the robot.

4. **A geometric segmentation method for extracting a continuum robot shape from an image:** This segmentation method uses a model of the robot's kinematics. Utilizing this prior knowledge eliminates the need for hand-labeled training data and effectively segments the robot despite challenging background conditions and partial occlusion.

1.5 Dissertation Overview

This dissertation is organized as follows. This introductory chapter has provided the general motivation for the work, a survey of relevant literature on the modeling, estimation, and control of continuum robots, and a summary of the contributions. Chapter 2 describes a new approach for controlling continuum robots based on leveraging real-time position and orientation measurements. The control strategy is demonstrated in both the constrained and unconstrained cases using a soft growing robot. Chapter 3 describes online generalized product of experts (OGPoE), which is a Gaussian Process-based online learning algorithm. We demonstrate how OGPoE can be used for online learning tasks that involve physical hardware and dynamic environments, including demonstrating online learning of the kinematics of a concentric tube robot — both in free space and in the presence of dynamic environmental constraints. Chapter 4 describes the development of a dynamic model specific to cable-driven continuum robots. The model is used to design an observer to estimate the full state of the robot using only partial state feedback, and the observer is then incorporated into a control loop designed to reject a constant disturbance applied to the robot body. Chapter 5 describes an algorithm to segment a continuum robot from a noisy image without requiring any training data. Finally, Chapter 6 summarizes the results, reviews the contributions of this dissertation, and provides perspectives for possible future work.

Chapter 2

Closed-loop Position Control for Growing Robots via Online Jacobian Corrections

2.1 Introduction

Recently, growing robots—which are capable of actively changing the length of their bodies—have emerged as a promising class of continuum robot [3] [49]. One method used to achieve this growing motion is to tuck, or fold, one end of a thin-walled, hollow tube back inside itself, and apply pressure. When pressurized with fluid, this tucked material everts from the tip, enabling the robot to move through the environment in a manner analogous to how plants grow. Because the body of the robot remains stationary with respect to the environment, these growing robots impart very low forces on their surroundings, offering the potential for even safer interactions. To-date, many of these growing robots have been passive, meaning that their deployed shape depends on the pre-fabricated shape and on the surrounding environment, but cannot be actively controlled [50, 51, 52]. In this case, the robot must rely on interactions with the environment for navigation [53, 54, 55]. Active steering of growing robots has been achieved using various pneumatic artificial muscle designs [51, 56, 24, 57], and control strategies all require either a human operator, the robot workspace to be obstacle free, or for every obstacle to be known *a priori* [58, 59, 60]. In this chapter, however, we seek to enable automatic control of the tip of a growing robot *without* these requirements.

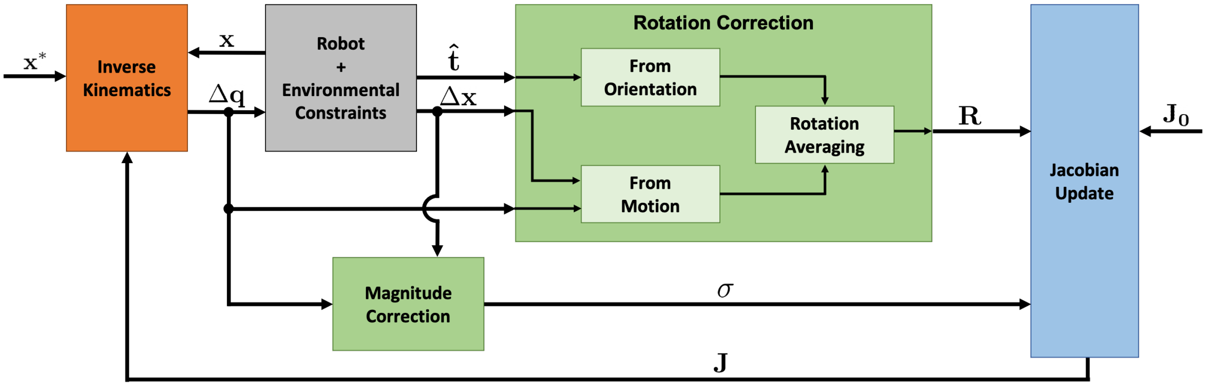


Figure 2.1. Flowchart of proposed closed-loop control method, including an online Jacobian update via rotation and magnitude corrections. These corrections are based on leveraging real-time position and orientation measurements of the tip of the robot.

2.1.1 Related Works

Previous works have solved the constrained control problem for other kinds of continuum robots by estimating a local model of the robot's Jacobian that is valid in a small region of the robot's configuration space. This model is then updated online via optimization using measurements of the robot's end-effector position [19], [46], [47]. Importantly, unlike other control methods, these updates to the Jacobian do not rely on assumptions about the underlying physics of the robot, as in [61], or require knowledge of the robot's environment, as in [62]. Although this strategy has been shown to successfully maneuver continuum robots in unknown, static environments, it risks developing a singularity in the model over time because the model updates are not guaranteed to preserve the structure of the original Jacobian [63]. More recent work has explored the use of images taken from an end-effector camera to estimate a 2D rotation matrix that could be used to iteratively update the Jacobian in a way that preserved its initial structure [63]. This approach allowed for the control of a continuum robot in a constrained environment, while preventing singularities. In this work, we build on this idea to estimate a 3D rotation matrix and a magnitude correction term using position and orientation measurements at the tip of the robot.

2.1.2 Contributions

The main contributions of this chapter are as follows. First, we extend the ideas originally presented in [63] to develop a new approach for controlling continuum robots based on leveraging real-time position and orientation measurements. This information on the robot pose is used to update a local model of the velocity kinematics of the robot in an online manner via corrective rotations and magnitude adjustments. This approach takes advantage of sensor information that is commonly available for these types of robots but that has been largely unused. A significant advantage, compared to methods that rely on a global model of the robot kinematics, is the ability to handle the many external contacts that continuum robots are likely to encounter during navigation in constrained environments, even if the exact robot configuration is unknown, while preserving the structure of the model estimate. Second, we combine this control strategy with a method for localizing the tip of a growing robot to demonstrate closed-loop position control for this class of robot that does not require a line of sight. This is the first method for autonomous position control of growing robots that does not require a map of its environment. We experimentally characterize the performance of the controlled growing robot in both the constrained and unconstrained cases.

2.2 Methods

We consider the problem of controlling a continuum robot with tip position and orientation measurements in the presence of unknown environmental constraints. We propose to use the streaming robot pose measurements to correct a local model of the robot's velocity kinematics in real time, which can then be used for control. In the rest of this section, we more formally define the local model correction in terms of a rotation and magnitude adjustment, and detail how we calculate these adjustments from robot pose measurements.

2.2.1 Problem Definition

Small changes to the actuators of a continuum robot, $\Delta\mathbf{q}$, can be approximately related to changes in the robot's 3D end-effector position, $\Delta\mathbf{x}$, by the linear equation

$$\Delta\mathbf{x} \approx \mathbf{J}\Delta\mathbf{q}, \tag{2.1}$$

where $\mathbf{J}$ is the Jacobian matrix. If $\mathbf{J}$ has full column rank, then its pseudoinverse can be used to find the minimum norm signal, $\Delta\mathbf{q}$, that achieves a desired change in end-effector position. When operating in unconstrained conditions, a reasonable estimate of the Jacobian can often be developed through either analytical (e.g. [64], [8], [56]) or empirical (e.g. [65]) means. However, as the robot interacts with unmodeled obstacles in its environment, this estimate becomes scaled and rotated and can result in drastically different behavior than predicted by the linear relationship [19].

This problem is described as

$$\mathbf{J}_*\mathbf{J}^\dagger\Delta\mathbf{x} \neq \Delta\mathbf{x}, \tag{2.2}$$

in [63], where $\mathbf{J}_*$ is the true robot Jacobian and $\mathbf{J}^\dagger$ is the pseudoinverse of the estimated Jacobian. We seek to find a linear correction, $\mathbf{K}$, such that

$$\mathbf{J}_*\mathbf{J}^\dagger\mathbf{K}\Delta\mathbf{x} = \Delta\mathbf{x}, \tag{2.3}$$

where correction matrix, $\mathbf{K}$, has a singular value decomposition (SVD) given by,

$$\mathbf{K} = \mathbf{U}\begin{bmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{bmatrix}\mathbf{V}^T. \tag{2.4}$$

If we assume all the singular values to be equal, $\sigma_1 = \sigma_2 = \sigma_3$, then $\mathbf{K}$ may be expressed simply as the product of a scalar and a unitary matrix, the latter of which we may write as a rotation matrix

$$\mathbf{K} = \sigma\mathbf{U}\mathbf{I}\mathbf{V}^T = \sigma\mathbf{R}. \tag{2.5}$$

Assuming that all of the singular values of the correction matrix are equal, implies that the correction matrix does not shear the Jacobian estimate [63], meaning that the relative angles between the column vectors of the Jacobian estimate do not change. This assumption guarantees that the correction matrix does not introduce an artificial singularity to the Jacobian update. Although operating the robot near the limits of its workspace or in the presence of environmental interactions may violate this assumption, in practice, approximate Jacobians have proven to be effective for continuum robot control, in constrained and unconstrained settings [19] [56].

Physically, the scalar, σ, varies to compensate for changes in the system energy due to viscoelastic losses in actuation, environmental interaction, [63] and—in the case of growing robots—changes in the robot length. The rotation matrix, $\mathbf{R}$, accounts for how the end-effector's orientation affects the directions of the control inputs. By estimating this scalar—which we will refer to as the magnitude correction—and rotation matrix—which we will refer to as the rotation or orientation correction—we can determine the total correction, $\mathbf{K}$, which can then be used to update our Jacobian estimate each time new pose measurements are available (see Fig. 3.1).

2.2.2 Orientation Correction

We discuss two approaches for estimating the rotation correction, $\mathbf{R}$. The first approach uses successive measurements of the end effector's position, while the second approach uses successive measurements of its orientation. In an ideal setting, this rotation correction estimated from either method would be identical. In practice, however, any sensor measurements will be corrupted by noise, making it advantageous to combine both estimates. Accordingly, we also describe a straightforward way to combine rotations so that both approaches may be used simultaneously to compute the correction matrix.

Rotation Estimation from Position Sensing

For every change in end-effector position at timestep k predicted by the Jacobian and given by $\Delta\hat{\mathbf{x}}_k = \mathbf{J}_{k-1}\Delta\mathbf{q}_{k-1}$, there is an actual motion, $\Delta\mathbf{x}_k$, measured by a position sensor. In order to correct our local Jacobian model, we would like to find the rotation matrix that aligns these two vectors ($\Delta\hat{\mathbf{x}}_k$ and $\Delta\mathbf{x}_k$). We can calculate this rotation matrix by computing the axis, $\boldsymbol{\omega}_k$, and angle of rotation, θ_k, between these two vectors as given by,

$$\boldsymbol{\omega}_k = \frac{\Delta\mathbf{x}_k}{||\Delta\mathbf{x}_k||} \times \frac{\Delta\hat{\mathbf{x}}_k}{||\Delta\hat{\mathbf{x}}_k||}, \tag{2.6}$$

$$\theta_k = \cos^{-1}\left(\frac{\Delta\mathbf{x}_k \cdot \Delta\hat{\mathbf{x}}_k}{||\Delta\mathbf{x}_k||\;||\Delta\hat{\mathbf{x}}_k||}\right). \tag{2.7}$$

We can then employ Rodrigues' rotation formula,

$$\mathbf{R}_k = \mathbf{I} + (\sin\theta_k)[\boldsymbol{\omega}_k] + (1-\cos\theta_k)[\boldsymbol{\omega}_k]^2, \tag{2.8}$$

where $[\boldsymbol{\omega}]$ is the skew symmetric matrix representation of vector $\boldsymbol{\omega}$.

Rotation Estimation from Orientation Sensing

Another approach to determining the rotation correction is to directly sense the orientation of the end effector. If full 3D orientation measurements are available, as is the case with some electromagnetic sensors for instance, then the end effector orientation may be described by a 3x3 rotation matrix, $\mathbf{T}$. The rotation correction can then be determined simply by multiplying successive orientations of the end effector using

$$\mathbf{R}_k = \mathbf{T}_{k-1}^T \mathbf{T}_k. \tag{2.9}$$

For other sensing modalities, such as certain electromagnetic trackers and permanent magnet-based systems, it is impossible to obtain a reliable measurement of the twist about the axis of the continuum robot ([66]). A 2-degree-of-freedom orientation measurement is still useful, however, and may be expressed as a unit vector, $\hat{\mathbf{t}}$, pointing in the direction tangent to the robot backbone at the tip. Using two consecutive 2-degree-of-freedom orientation measurements, we can compute a rotation matrix correction to our Jacobian by calculating the matrix that aligns the measurement from the previous timestep, $\hat{\mathbf{t}}_{k-1}$, with the orientation measurement from the current timestep, $\hat{\mathbf{t}}_k$, using equations (2.6)-(2.8). It is important to note that rotation corrections computed from 2-degree-of-freedom measurements will not capture any distortions to the robot Jacobian that are due to twisting about its backbone, and therefore must be combined with the rotation estimates derived from changes in robot tip position.

Combining Rotation Matrices

In order to take advantage of the rotation estimates derived from both the position and orientation measurements of the robot pose, we seek a method for combining rotation matrices. Because we find rotation estimates derived from changes in position to be more susceptible to sensor noise than those derived from orientation sensed directly, our method should allow us to give more weight to the orientation-based estimate. One such method is to take a weighted average of the matrix elements and then re-orthonormalize the result using the SVD [67]. The resulting, composite rotation matrix, $\mathbf{R}_k$, can then be used in the rest of the algorithm.

$$\begin{aligned} \mathbf{R}_{sum} &= \sum_i w_i \mathbf{R}_i, \quad \text{where} \quad \sum_i w_i = 1 \\ \mathbf{R}_{sum} &= \mathbf{U}\Sigma\mathbf{V}^T \\ \mathbf{R}_k &= \mathbf{U}\mathbf{V}^T \end{aligned} \tag{2.10}$$

This method has been shown to be as effective as averaging over alternative representations of rotations, such as axis-angle or quaternion parameterizations, provided the individual rotations are sufficiently close to one another [67][68].

For our subsequent experiments with growing robots, which use a 2-degree-of-freedom orientation measurement, we found that the orientation-based rotation estimates were qualitatively less noisy across time-steps than the position-based estimates. We could not use this estimate alone, however, since it fails to capture rotations due to twisting along the backbone of the robot. Accordingly, we assigned a weight of 0.9 to the orientation-based estimate to be used in the averaging.

2.2.3 Magnitude Correction

In order to estimate the scalar portion of our correction matrix, we rewrite (2.1) with our corrected Jacobian estimate,

$$\Delta \mathbf{x}_k = \mathbf{K}_k^{-1} \mathbf{J}_{k-1} \Delta \mathbf{q}_{k-1} = \frac{1}{\sigma_k} \mathbf{R}_k^T \mathbf{J}_{k-1} \Delta \mathbf{q}_{k-1}, \tag{2.11}$$

and consider $\mathbf{R}_k^T \mathbf{J}_{k-1} \Delta \mathbf{q}_{k-1} = \Delta \hat{\mathbf{x}}_{R,k}$ to be the rotated prediction of end-effector motion based on our Jacobian estimate. If we perfectly estimate the rotation correction, $\mathbf{R}_k$, then the vectors $\Delta \mathbf{x}_k$ and $\Delta \hat{\mathbf{x}}_{R,k}$ should be colinear, differing only in magnitude. Accordingly, we can choose σ_k to resolve this difference and adjust the magnitude of our prediction using,

$$\sigma_k = \frac{||\Delta \hat{\mathbf{x}}_{R,k}||}{||\Delta \mathbf{x}_k||}. \tag{2.12}$$

To accommodate sensor noise present in real-world systems, we apply a low-pass filter to the update of this magnitude correction

$$\sigma_k = \alpha \sigma_{k-1} + (1 - \alpha) \frac{||\Delta \hat{\mathbf{x}}_{R,k}||}{||\Delta \mathbf{x}_k||}. \tag{2.13}$$

Larger values of the filter constant, α, lead to updates that are less affected by noise, but at the expense of introducing phase lag and delaying the Jacobian update. We use a value of $\alpha = 0.95$ in all of our experiments. Finally, we update the Jacobian to use for prediction at the next timestep as

$$\mathbf{J}_k = \frac{1}{\sigma_k} \mathbf{R}_k^T \mathbf{J}_{k-1}. \tag{2.14}$$

2.2.4 Control Law

It is important to note that the methods described above to do online estimation of the continuum robot Jacobian do not depend on a particular choice of control law and may be used in conjunction with any Jacobian-based algorithm for position control.In our experiments, we compute the velocity we wish to command at the end-effector, $\Delta\mathbf{x}^*$, as the difference between the current end effector position, $\mathbf{x}$, and the target end-effector position, $\mathbf{x}^*$. The pseudo-inverse of the Jacobian estimate is then used to determine the control input for this commanded velocity:

$$\Delta\mathbf{q} = \mathbf{J}^{\dagger}\Delta\mathbf{x}^*. \tag{2.15}$$

2.3 System Overview

To empirically evaluate the proposed control strategy we use a steerable growing robot. The robot hardware, along with the test platform designed for assessing system errors, are described here.

2.3.1 Growing Robot Hardware

A steerable growing robot capable of 3-D motion was fabricated using three fabric artificial pneumatic muscles (fPAMs) as actuators. The fPAMs were made consistent with the design described in [57], and contract when the pressure inside is increased (see Fig. 2.2). When placed radially around the main tube of the robot at equal angles, the fPAMs can be used to bend, or steer, the robot, similar to tendons. The robot was fabricated from a lightweight, silicone-urethane impregnated ripstop fabric (Seattle Fabrics) and has a maximum deployed length of 445 mm and an inflated diameter of 20 mm, including the surrounding actuators. Three Proportion Air QB1X pressure regulators are used to modulate the airflow in and out of each fPAM. The robot material is inverted through an N52 ring magnet (12.7 mm OD × 6.4 mm ID × 9.5 mm L) and wrapped around a motorized spool housed inside a pressure vessel. The base of the robot is attached to the housing outlet (see Fig. 2.2). We control the deployed length of the robot by setting the position of the motorized spool and supplying a sufficiently high inlet pressure to the housing ($\sim$41 kPa), similar to the mechanism in [59]. In total, we can control 3 degrees-of-freedom of the robot tip: two that determine the heading and one that determines

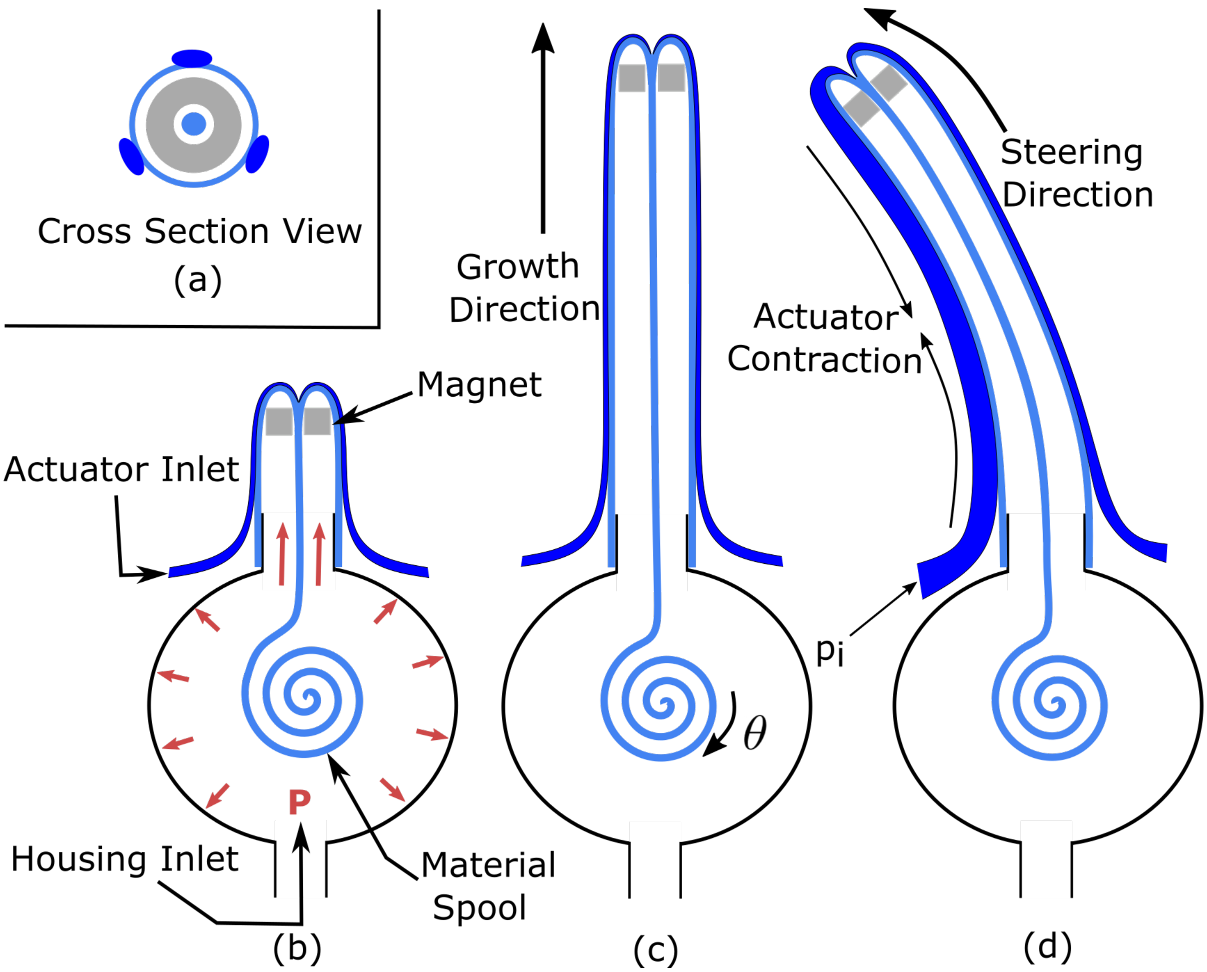


Figure 2.2. The overall system consists of a steerable growing robot with (a) a magnet at the tip and three, evenly spaced actuators. (b) The base of the robot is attached to the outlet of a pressure vessel, and the inverted material is wrapped around a spool inside the vessel. (c) As pressure is applied to the robot body, material everts from the tip, enabling growth. (d) The length of the robot is controlled by the spool position θ. As a pressure signal p_i is applied to the ith actuator, the robot bends, enabling steering.

deployed length.

We localize the distal end of the robot by tracking a permanent ring magnet at its tip. As the robot grows and bends, friction between the magnet and the inverted robot material, along with the pressure of the working fluid, keep the magnet at the tip of the robot [66]. Therefore, motion of the robot causes changes in the position and orientation of the magnet, which in turn causes changes in the surrounding magnetic field. We sense these changes with an array of

PNI3100 magnetoinductive sensors that are arranged in a four by two grid pattern beneath the robot (see Fig. 2.3). By solving a real time optimization problem that minimizes the difference between the sensor readings and a model of the field produced by the magnet, we can determine the position of the magnet center and its 2-degree-of-freedom orientation (see [66] for details). A laptop with an Intel i7-8650U processor runs the control loop and manages data collection using MATLAB.

2.3.2 Characterizing System Error

In order to assess the error from the combined localization and control approaches, we use a ground truth measurement of the robot tip position that is independent of the magnetic localization described above. Two Logitech C920 webcams are mounted to the test platform, one with an overhead view and the other with a view that is orthogonal to it (see Fig. 2.3). Both the camera intrinsics and extrinsics were calibrated in MATLAB using the Camera Calibrator and Stereo Camera Calibrator apps respectively. The mean reprojection error after calibration is less than one pixel. To obtain tip position measurements from images of the robot taken by these cameras, we manually label the robot tip in each image and triangulate the 3D coordinate of the tip position using the labels and the calibrated camera parameters.

This ground truth measurement allows us to more fully characterize the error of the entire closed-loop system. Specifically, we consider three points of interest and the vector differences between them. First is the actual tip position, $\bar{\mathbf{x}}$, as determined by the ground truth measurement. The second, which is used in the control loop, is the estimated tip position, $\mathbf{x}$, determined by localizing the permanent magnet. Third is the target tip position, $\mathbf{x}^*$, which corresponds to the goal position that the robot must reach in each task. Although all three of these points would coincide in a perfect system, in reality, we must investigate the error between each pair of points. We label these errors as follows: the controller error, defined as the difference between the target tip position and the estimated tip position; the localization error, defined as the difference between the estimated tip position and the actual tip position; and the

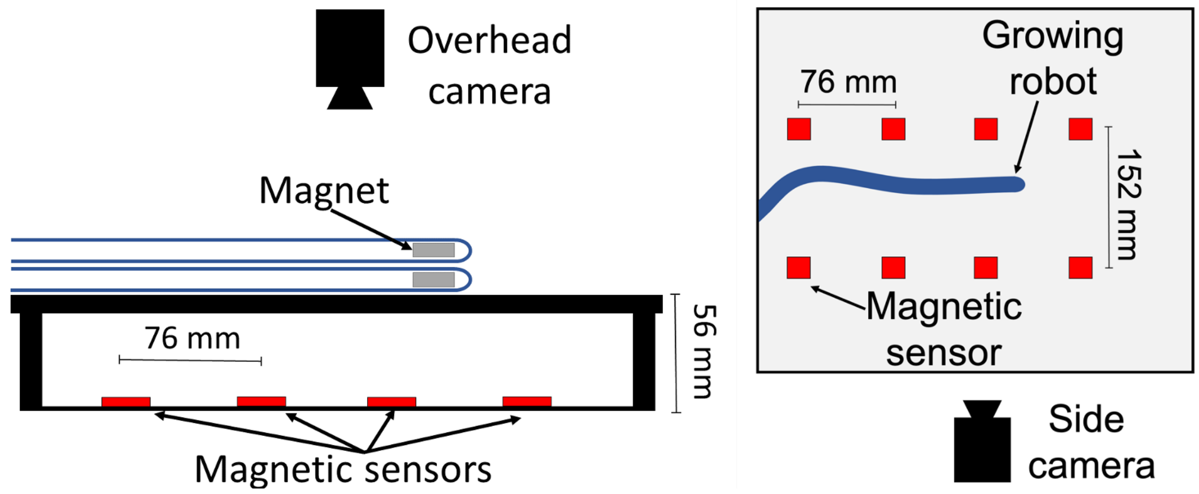


Figure 2.3. The experimental setup includes an array of magnetic sensors, used for localizing the tip of the growing robot, along with two orthogonal cameras, used to obtain a ground truth measurement of the robot tip position.

total error, defined as the difference between the actual tip position and the target tip position. Note that the total error is the vector sum of the controller and localization errors as shown in Fig. 2.4.

2.4 Experiments and Results

We evaluate the performance of our proposed control strategy both in simulation and on a physical growing robot system. We evaluate the control error, as well as the localization error, in order to characterize the overall performance of the entire closed-loop system.

2.4.1 Simulation

In order to study the controller performance without the effects of localization inaccuracy, robot dynamics, and environmental constraints, we created a simulated, path-following task in MATLAB. We choose to run this simplified simulation to verify that the controller would work in an ideal setting. Simulating the more complex behaviors associated with these robots was deemed beyond the scope of this work, given that dynamic simulations of even simpler continuum robots is still active research [69] [70]. We leave the study of controller performance in the presence of noise, dynamics and constraints for the physical experiments in the subsequent sections.

The simulated robot obeys a kinematic model developed in [56] that maps actuator inputs to unique robot configurations based on the constant-curvature assumption and a linear model of the relationship between actuator pressure and contraction. The model parameters were chosen such that the behavior of the simulated robot approximated that of our physical robot.

As seen in Fig. 2.5, the controller successfully guides the tip of the robot along a specified spiral trajectory using the tip position and 2-degree-of-freedom orientation. Because the controller uses the true pose of the robot tip in this simulation, the localization error is zero

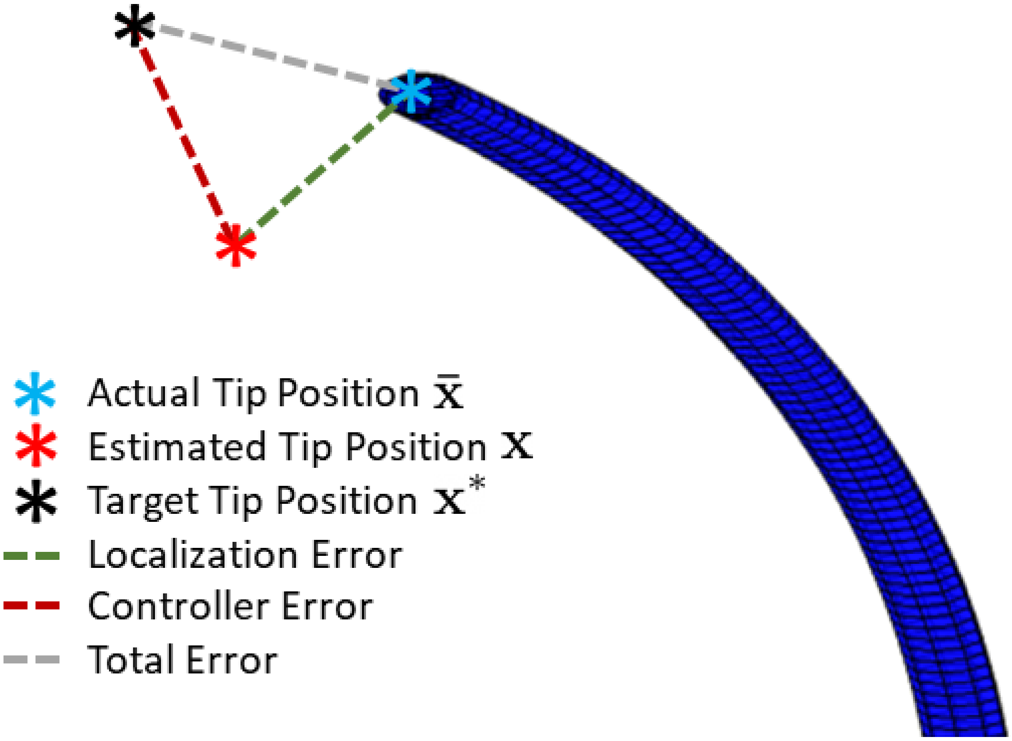

Actual Tip Position $\bar{\mathbf{x}}$
Estimated Tip Position $\mathbf{x}$
Target Tip Position $\mathbf{x}^*$
Localization Error
Controller Error
Total Error

Figure 2.4. Three error types are considered for evaluation of the closed-loop system: (1) localization error, defined as the difference between the estimated tip position and the actual tip position; (2) controller error, defined as the difference between the target tip position and the estimated tip position; and (3) total error, defined as the difference between the actual tip position and the target tip position.

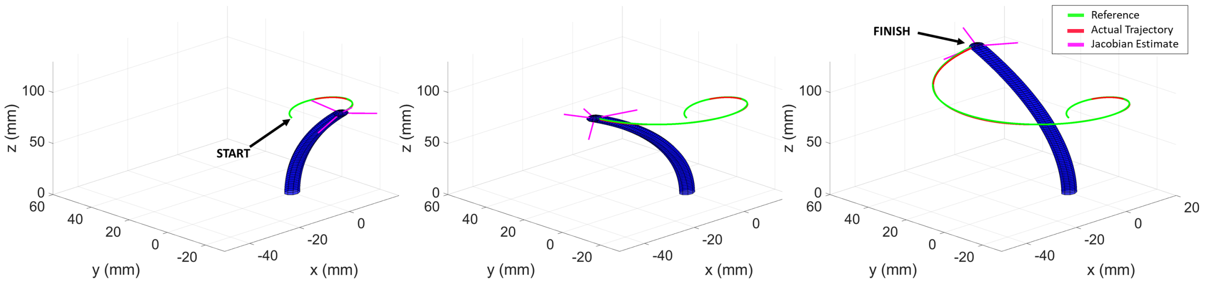


Figure 2.5. Time steps of a path following simulation, including the reference (green) and actual (red) trajectories, along with the Jacobian estimate (magenta).

and the controller error is the same as the total error, the mean of which is 0.45 mm. Simulation also helps illustrate the roles of the orientation and magnitude adjustment components of the proposed controller. The magenta vectors, shown in Fig. 2.5, point in the direction of the columns of the Jacobian estimate at the given timestep. It is clear that the controller's orientation

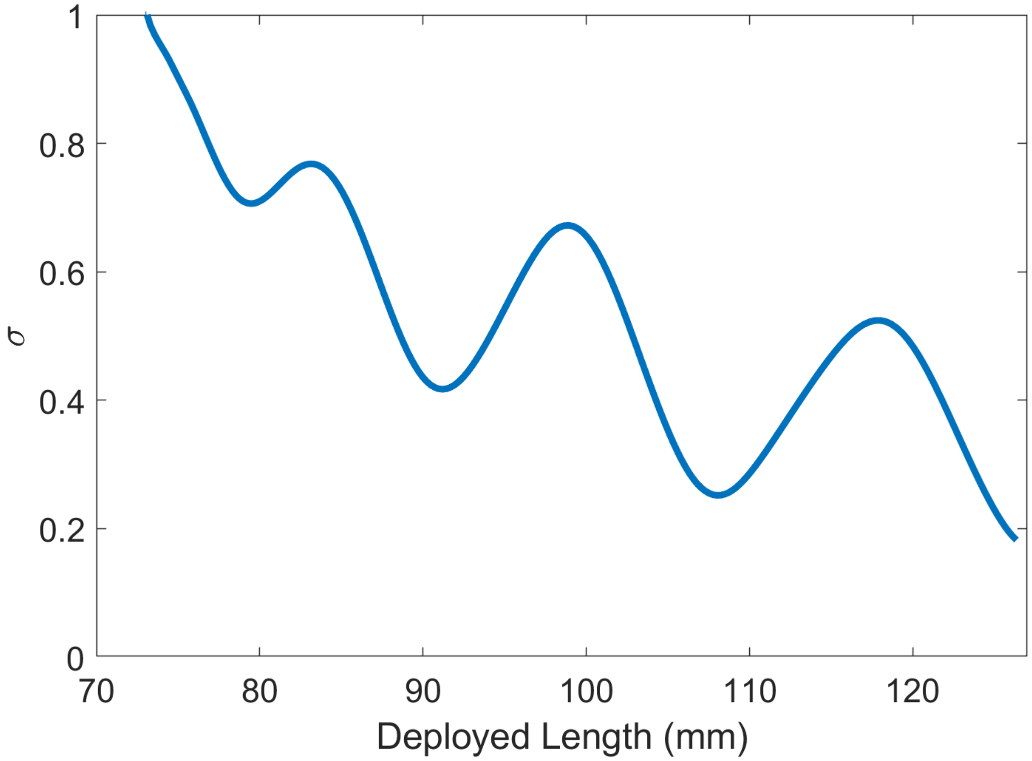


Figure 2.6. Magnitude adjustment constant versus the deployed length of the robot during the path following simulation.

adjustments successfully align the column of the Jacobian pointing in the direction of robot growth with the tangent of the robot's backbone. The magnitude adjustment constant, σ, is plotted against the deployed length of the simulated robot in Fig. 2.6. The overall downward trend is expected as the robot grows, since the same magnitude of actuation has larger effects on the change in end effector position for longer robot lengths.

2.4.2 Point Reaching in Free Space

Next, we implemented the control strategy on our physical growing robot system to perform a point reaching task. We selected five points within the robot's workspace and connected them via straight lines to form a trajectory for the robot to follow (see Fig. 2.7). We initialize our estimate of the robot Jacobian one column at a time, as described in [19], by incrementing each of the actuators a small amount, Δq_i, and recording the resulting change in position, $\Delta \mathbf{x}_i$, determined by the magnetic localization. The robot Jacobian column for actuator i is then given by, $J_i = \Delta \mathbf{x}_i / \Delta q_i$. Note that initializing the Jacobian in this way does not require a model of the robot's kinematics—the linear map from the actuator velocities to tip velocities is approximated from data.

We evaluate the effectiveness of the controller by comparing the robot's actual, estimated, and target tip positions at each of the points as described above in Section 2.3.2. In order to demonstrate the repeatability of the results, this experiment was run five times. The mean and standard deviation of the absolute controller error, localization error, and total error are displayed in Table 2.1. The consistency of the mean total error across all five of the test points and the low standard deviation indicate the repeatability of the trials. The mean total error across all five points is 3.22±1.31 mm. These results could be further improved by reducing the localization error, which is significantly higher than the controller error, through the addition of more sensors, the use of a stronger magnet, or investigating ways to optimize the placement of the sensors so as to guarantee a certain localization accuracy over the robot's workspace. These results demonstrate that our controller is able to accurately and reliably position the tip of the growing robot on a physical system using measurements of the tip position and orientation.

Table 2.1. Mean and standard deviations of the controller error, localization error, and total error for each of the five points in the free-space, point-reaching task. On average, the robot reaches a deployed length of 308 mm.

Pt #	Controller Error	Localization Error	Total Error
1	0.50 ± 0.23	2.69 ± 0.90	2.52 ± 0.84
2	1.60 ± 0.60	5.07 ± 1.09	3.87 ± 0.67
3	1.29 ± 0.62	2.02 ± 0.96	2.05 ± 1.54
4	1.15 ± 0.64	3.66 ± 0.82	4.13 ± 0.91
5	1.94 ± 1.21	3.08 ± 0.96	3.56 ± 1.40

2.4.3 Point Reaching in a Constrained Environment

Finally, we investigate the closed-loop system performance in the presence of environmental constraints. We designed a branching acrylic structure meant to be a proxy for an industrial or medical task, such as pipe inspection or ablation, in which a growing robot would be required to navigate to a target site and reach multiple points once at that site. Previous research has shown that growing robots are capable of traversing highly-curved, constrained paths with deterministic outcomes through growth alone [53] [54]. Accordingly, we choose to focus instead on the branch selection task since it necessitates active steering control of the tip

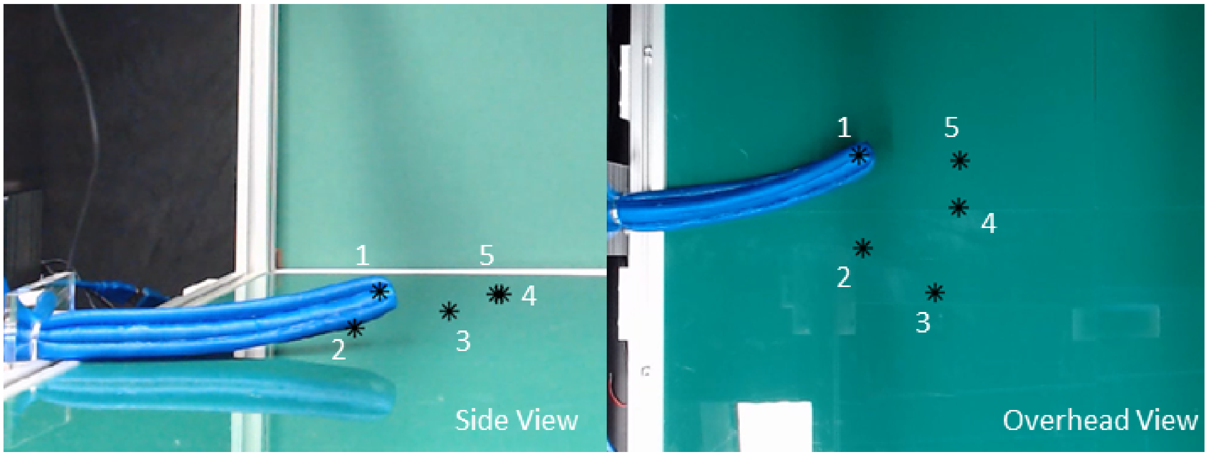


Figure 2.7. Side and overhead views of the growing robot with the five target points from the free-space, point-reaching task overlaid.

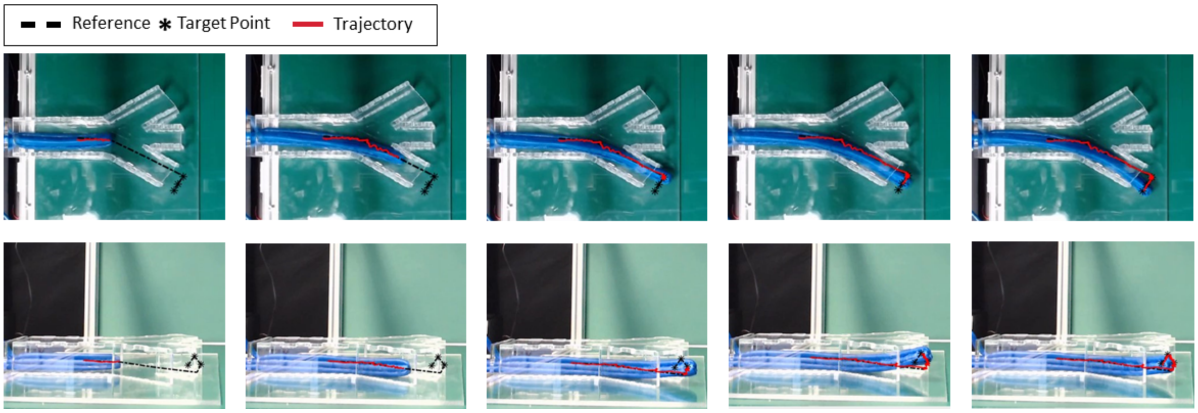


Figure 2.8. Still-frame images from the overhead and side cameras during the point-reaching task in the constrained environment. The proposed control approach is used to steer the growing robot along the reference trajectory (overlaid dashed line). The actual trajectory is overlaid in red, along with the three target points located at the end of the branch.

of the robot.

For each branch, we generated a path for the robot to follow by running the robot in open-loop through that branch and then approximating the recorded trajectory by straight-line segments. We chose three points at the end of the path within the robot's workspace that would require the robot to make contact with each of the branch outlet walls in order to reach all three (see Fig. 2.8). Requiring the robot to make and break contact with the constraints posed by the environment tests the controller's ability to compensate for unknown forces at unknown positions along the robot length, as would be expected in a real-world scenario.

We initialized the robot Jacobian as before and ran the controlled robot through this constrained, point-reaching task five times per branch. As before, the controller only uses online pose estimates and does not have a map of the obstacles in its environment. We evaluate the errors at the three target points for each branch. The results are shown in Table 2.2 and demonstrate that the closed-loop system is still accurate and repeatable in the constrained setting. The controller error is slightly higher than, though still similar to, the error in the unconstrained case. This is likely due to the presence of the contacts from the environment which introduce effects that are hard to compensate for, such as friction. Although the environment is static, as

Table 2.2. Mean and standard deviations of the controller error, localization error, and total error for target points reached via each branch of the constrained environment. On average, the robot reaches a deployed length of 424 mm.

Branch	Controller Error	Localization Error	Total Error
Upper	2.95 ± 2.12	3.66 ± 1.59	3.84 ± 1.11
Middle	2.24 ± 1.61	4.88 ± 1.37	5.79 ± 1.74
Lower	1.27 ± 0.88	3.59 ± 0.65	4.05 ± 0.93

the robot grows, the location and type of contact changes. As seen in Fig. 2.8, there are instances of multiple unknown, time-varying distributed and point forces applied to different locations along the length of the robot backbone. These constraint forces are truly three dimensional because the robot must make contact with each of the three walls of the environment in order to reach the target points. The mean total error of the closed-loop system over all runs and all branches is 4.56±1.56 mm. As in Section 2.4.3, this could also be reduced by improving localization error, which is greater than the controller error, particularly for the middle branch, which is furthest away from any of the magnetic sensors. It is worth noting that although the constrained environment does limit the maximum possible controller error, the width of the acrylic channel (35 mm) provides sufficient space for errant robot motion and allows the controller performance to be judged effectively.

2.5 Discussion

We have shown that a series of updates, estimated from continuum robot pose measurements, can be successfully applied to correct an estimate of the robot's Jacobian without introducing artificial singularities. We have also shown that this Jacobian estimate can then be used to steer a robot to a desired target, even in the presence of environmental constraints. By combining this control approach with a permanent magnet-based approach to localization, we have enabled closed-loop control for growing robots that does not require line of sight. We characterized the closed-loop system error in both the free space and constrained cases, demonstrating the accuracy and repeatability with which the controller can position the robot tip on a physical system. Notably the constrained environment in which we tested the robot was static. One of the main limitations of the proposed approach is that it does not distinguish robot motion from environment motion. As such, it is not currently suited for use in cases where rapid changes in the environment have large effects on the robot dynamics. Additionally, as with other model-free Jacobian estimation algorithms, the quality of the Jacobian estimate depends on the quality of the sensor information available—if the measurements of the robot pose are corrupted by noise, this will lead to a poor estimate of the Jacobian, which in turn will lead to poor controller performance. Noisy data must be filtered to produce a clean signal, but at the expense of introducing a delay in the Jacobian update, which limits the speed at which the robot may be accurately steered along a trajectory.

Based on these results, there are several directions for future work. First, as the rapid pace of research on growing robots continues and new methods of sensing and actuation are developed, methods for improved control algorithms can be created by incorporating this new hardware and associated sensor information. For example, one drawback of the permanent-magnet based localization used here, is that it requires magnetic sensors to be placed throughout the robot workspace before the robot is deployed, limiting the number of applications where it can be used. By investigating other localization technologies that could be made to be compatible with these robots, we could overcome this limitation. Second, given the usefulness of the Jacobian estimation-based controller and the many safety-critical scenarios in which it is intended to be applied, theoretical guarantees can be developed to rigorously define the operating conditions under which it can confidently be deployed. In the future, it will be important to analyze the stability of the controller in the presence of real-world phenomena such as sensor and actuator noise, robot dynamics, and environmental constraints.

2.6 Acknowledgements

This chapter is, in part, a reprint of material from C. Watson, R. Obregon, T.K. Morimoto, ”*Closed-Loop Position Control for Growing Robots Via Online Jacobian Corrections*.” The dissertation author is the primary author of this work.

Chapter 3

Learning Non-Parametric Models in Real Time via Online Generalized Product of Experts

3.1 Introduction

Many real-world robotics problems involve phenomena such as friction and uncertainty that are difficult to model from first principles. These challenges have motivated the use of learning approaches that build models directly from data without requiring expert knowledge of the underlying system. While these learning-based approaches have achieved impressive results, many of the algorithms can only be performed offline, limiting their practical use.

Online learning algorithms seek to address this limitation by inferring models from data and adapting these models continually over time as new data arrives. In particular, nonparameteric online learning methods based on the popular Gaussian process (GP) statistical model are promising due to their many desirable properties, such as adaptable modeling complexity and prediction uncertainty quantification [71].

3.1.1 Related Works

Many approaches have been proposed to minimize the high computational cost of GP models, which scales cubically with the size of the dataset, N, during training [72]. Approximation strategies for GPs can typically be classified as either global or local methods. Global approximation strategies involve choosing $M \ll N$ points, called inducing or psuedo-inputs, that, when used for training, produce an approximate GP with minimal KL-divergence from a GP trained with all N samples. The inducing input locations may be sampled from the available data (based on, e.g., information gain [73], KL divergence [74], or Nyström sampling [75]), or considered as variables to be optimized for jointly with the kernel hyperparameters during training [76]. However, for online learning problems where the size of the dataset grows large over time, the quality of the GP approximation using a fixed number of inducing points may degrade [77] or be confined to a local region of the input space [78].

Local GP approximations improve scalability by first dividing the training dataset into smaller subsets (e.g. by k-means [79], randomly [80], or by feature space distance from existing local models [81]) and then training a local GP model on each subset of data. Prediction on new inputs can be done with the local model nearest the input (i.e. naive local experts (NLE) [72]) or by taking a weighted combination of predictions from all local models (i.e. mixture of experts (MoE) [82] or product of experts (PoE) [83]). Adapting this local GP approximation strategy to the online setting is difficult because most of the effective data partitioning and prediction combination methods that have been used to date, require some form of offline computation.

Recent works have proposed to combine both local and global GP approximations to leverage the benefits of each approach [84] [85]. Training data is divided between multiple, local models, where each local model is itself a sparse GP approximation. Although these approaches substantially reduce training times and effectively scale the complexity of the models to the observed data, there remain limitations in their prediction strategies. For instance, [85] uses an NLE strategy, which is known to give artificially discontinuous predictions on the boundaries of the subregions between local models [72]. The prediction strategy in [84], on the other hand, uses an MoE approach to prediction, but does not learn the weighting function jointly with the local model parameters, which typically leads to suboptimal predictions [80].

To address remaining limitations in using GPs for online learning, we propose to adapt the generalized product of experts (GPoE) framework for multiple model prediction [80] to the online setting. GPoE is a method for combining the predictions from multiple local models in a way that does not require joint training, allowing each local model to be learned independently. Despite this independent training, the aggregate GPoE prediction has been shown to retain important properties of multiple model prediction, such as rejection of poor predictions from local models, smooth interpolation between local models, and the ability to precisely combine many uncertain local model predictions [80].

3.1.2 Contributions

The contributions of this paper are as follows. First, we present a new algorithm, which we call online generalized product of experts (OGPoE), that extends GPoE to the online learning setting. This algorithm offers a combination of benefits unavailable from other online learning algorithms, including the simultaneous online optimization of local model parameters, a clear and interpretable data partitioning strategy, and principled fusion of multiple model predictions without the need for joint training. We evaluate our proposed algorithm in simulation and show that it compares favorably to other popular online and offline modeling approaches on a well-cited benchmark dataset. Finally, we demonstrate how OGPoE can be used for online learning tasks that involve physical hardware and dynamic environments. Specifically, we demonstrate online learning of the kinematics of a concentric tube robot — both in free space and in the presence of dynamic environmental constraints — which is a first for this type of robot. These experiments not only demonstrate the modeling accuracy that OGPoE can achieve, but also its versatility in the types of problems it applies to, making it a powerful tool for online learning.

3.2 Background

In this section, we review GP fundamentals, along with key works that are leveraged and adapted for OGPoE.

3.2.1 Streaming Sparse Gaussian Process Approximations

A standard GP regression model assumes that input-output data pairs, $\{\mathbf{x}, y\}$, can be described by the expression $y = f(\mathbf{x}) + \varepsilon$, where f is an unknown function corrupted by Gaussian observation noise $\varepsilon \sim \mathcal{N}(0, \sigma_y^2)$ [77]. A GP prior over the unknown function, $f \sim \mathcal{GP}(m(\cdot), k(\cdot, \cdot|\theta))$, is specified by a mean function, $m(\cdot)$, and a covariance function, $k(\cdot, \cdot)$, that depends on hyperparameters θ. As is typically done, we adopt a zero-mean function for notational convenience in the remainder of this section. The choice of covariance (or kernel) function encodes prior assumptions about the spatial correlations between the inputs to the unknown function, f, and may be informed by prior knowledge. We employ the popular and interpretable squared exponential kernel with automatic relevance determination,

$$k(\mathbf{x},\mathbf{x}') = \sigma_f^2 \exp\left(-\frac{1}{2}(\mathbf{x}-\mathbf{x}')^\top \mathbf{W}^{-1}(\mathbf{x}-\mathbf{x}')\right), \tag{3.1}$$

which depends upon the hyperparameters that capture the signal variance (σ_f^2) and the length-scale along each dimension of the input ($\mathbf{W} = \mathrm{diag}(l_1^2,...,l_d^2)$).

Given a collection of input-output data pairs, the assumption of Gaussian observation noise, and the GP prior over f, it is possible to compute the mean and variance of the posterior process in closed form at a new input, $\mathbf{x}_*$, as

$$\begin{multline} m_*(\mathbf{x}_*) = k(\mathbf{x}_*,\mathbf{X})(\sigma_y^2\mathbf{I}+k(\mathbf{X},\mathbf{X}))^{-1}\mathbf{y} \\ V_*(\mathbf{x}_*) = k(\mathbf{x}_*,\mathbf{x}_*) - k(\mathbf{x}_*,\mathbf{X})(\sigma_y^2\mathbf{I}+k(\mathbf{X},\mathbf{X}))^{-1}k(\mathbf{X},\mathbf{x}_*) \quad (3.2) \end{multline}$$

The kernel matrix, $k(\mathbf{X},\mathbf{X})$, consists of kernel function evaluations between all N datapoints, and must be inverted for inference. As N approaches $\sim 10^4$, this computation becomes intractable, motivating methods for choosing $M \ll N$ inducing inputs to approximate the posterior and retain tractability [76]. The vast majority of these approximation methods, however, can only be applied when all training data is available at once, which is not the case in the online setting where data is revealed incrementally over time.

To address this problem, the authors in [77] pose an optimization problem that *recursively* updates the locations of M inducing points, $\mathbf{Z} = \{\mathbf{z}_1,...,\mathbf{z}_M\}$, as well as the kernel hyperparameters, $\theta = \{\sigma_y^2, \sigma_f^2, \mathbf{W}\}$, of a sparse GP model as data is processed online. Specifically, they show that given some current posterior GP approximation, $\mathscr{GP}_{\text{old}}$, with hyperparameters, θ_{old}, and inducing input locations, $\mathbf{Z}_a$, that is assigned new data, $\mathscr{D}_{\text{new}} = \{\mathbf{x}_i, y_i\}_{i=1}^{Nnew}$, then a quantity called the online variational free energy ($\mathscr{F}_{\text{OVFE}}$) can be maximized to find values for new hyperparameters and inducing input locations, $\{\theta_{\text{new}}, \mathbf{Z}_b\}$. The new hyperparameters and inducing input locations are then used to form a new posterior approximation, $\mathscr{GP}_{\text{new}}$, which, in turn, can be used for prediction [77].

3.2.2 Generalized Product of Experts

The PoE method for ensemble prediction [83] attempts to model a target distribution as the product of multiple densities, each of which is given by a single expert. This method is sensitive to errors in the predicted variance of the local experts, in that erroneously low-variance predictions from some local models will bias the aggregate mean and lead to overconfident predictions [80]. To suppress the influence of poor experts on the overall model, GPoE [80] introduces a weight to each expert's prediction,

$$p(y_*|\mathscr{D},\mathbf{x}_*) = \frac{1}{Z}\prod_{i=1}^{M} p_i^{\beta_i(\mathbf{x}_*)}(y_*|\mathscr{D}_i,\mathbf{x}_*), \tag{3.3}$$

where Z is a normalizer, $\mathscr{D}_i$ is the training data seen by the ith model, $\mathbf{x}_*$ is a predictive input, and y_* is an output. The input-dependent weight function, $\beta_i(\mathbf{x}_*) = \frac{1}{2}(\log\sigma^2_{i,**} - \log V_i(\mathbf{x}_*))$ is the differential entropy between the expert's prior and posterior. When the entropy change for an expert is zero at $\mathbf{x}_*$, then that expert's prediction does not contribute to the combined prediction because it does not have any training information relevant to this new input. Using this weight and Eq. (3.3), the aggregate mean (m_{GPoE}) and variance (V_{GPoE}) of the ensemble of experts can be computed in closed form [80]. This simple strategy for weighting predictions still requires no joint training and alleviates the PoE problem of being unable to reject poor experts. Despite its simplicity and minimal computational requirements, GPoE achieves excellent results on multiple benchmarks [80].

3.2.3 DP-Means

DP-means [86][87] is a nonparametric clustering technique that allows data to be partitioned for local model training *without* specifying the number of clusters *a priori*. We consider a set of streaming data, $\{\mathbf{x}_1,...,\mathbf{x}_T\}$, where $\mathbf{x}_t \in \mathbb{R}^d$ is the data observed at time t.

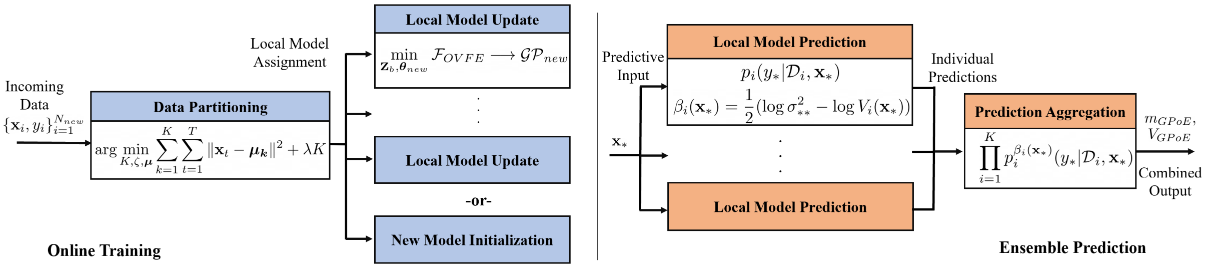

Incoming
Data
$\{\mathbf{x}_i, y_i\}_{i=1}^{N_{new}}$
Data Partitioning
$\arg\min_{K,\zeta,\boldsymbol{\mu}} \sum_{k=1}^{K}\sum_{t=1}^{T} \|\mathbf{x}_t - \boldsymbol{\mu}_{\boldsymbol{k}}\|^2 + \lambda K$
Local Model
Assignment
Local Model Update
$\min_{\mathbf{Z}_b,\boldsymbol{\theta}_{new}} \mathcal{F}_{OVFE} \longrightarrow \mathcal{GP}_{new}$
Local Model Update
-or-
New Model Initialization
Online Training
Predictive
Input
$\mathbf{x}_*$
Local Model Prediction
$p_i(y_*|\mathcal{D}_i, \mathbf{x}_*)$
$\beta_i(\mathbf{x}_*) = \frac{1}{2}(\log \sigma_{**}^2 - \log V_i(\mathbf{x}_*))$
Local Model Prediction
Individual
Predictions
Prediction Aggregation
$\prod_{i=1}^{K} p_i^{\beta_i(\mathbf{x}_*)}(y_*|\mathcal{D}_i, \mathbf{x}_*)$
m_{GPoE},
V_{GPoE}
Combined
Output
Ensemble Prediction

Figure 3.1. Flowcharts depicting online learning with incoming data and the GPoE framework for ensemble prediction. Online learning assumes that data arrives sequentially in small batches of N_{new} samples that are used either to update an existing local model, or to generate a new local model. Ensemble prediction returns an aggregate predictive mean and variance, m_{GPoE}, V_{GPoE}, for an input $\mathbf{x}_*$.

Each observation is assumed to belong to one of K clusters and is labeled $\{\zeta_1,...,\zeta_t\}$, with $\zeta_t \in \{1,...,K\}$. A prior over the number of clusters, K, and the cluster assignments, $\zeta_{1:t}$, specified by a Chinese Restaurant Process (CRP) [88] with hyperparameter α, is assumed. The first observation, $\mathbf{x}_1$, is assigned to the first cluster, $\zeta_1 = 1$, and the tth observation is assigned either to existing cluster k, with probability proportional to the number of data already in that cluster, or to a new cluster, with probability proportional to α. If the data in each cluster is assumed to be Gaussian with cluster-specific means, $\boldsymbol{\mu}_k$, and a shared variance, $\sigma^2\mathbf{I}_d$, then the likelihood of the data, $\mathbf{x}_{1:t}$, given clustering $\zeta_{1:t}$ and means $\boldsymbol{\mu}_{1:k}$, is

$$p(\mathbf{x}|\zeta,\boldsymbol{\mu}) = \prod_{k=1}^{K}\prod_{t=1}^{T}\mathcal{N}(\mathbf{x}_t|\boldsymbol{\mu}_k,\sigma^2\mathbf{I}_d). \tag{3.4}$$

Further assuming a zero-mean Gaussian prior on the cluster means themselves, $\boldsymbol{\mu}_k \sim \mathcal{N}(0,\rho^2\mathbf{I}_d)$, Bayes theorem gives the posterior distribution over the clustering,

$$p(\zeta,\boldsymbol{\mu}|\mathbf{x}) \propto p(\mathbf{x}|\zeta,\boldsymbol{\mu})p(\zeta)p(\boldsymbol{\mu}). \tag{3.5}$$

The *maximum a posteriori* (MAP) estimate for the clustering and the cluster means is found by maximizing this posterior distribution or, equivalently, minimizing the negative log joint likelihood

$$\max_{K,\zeta,\boldsymbol{\mu}} p(\zeta,\boldsymbol{\mu}|\mathbf{x}) \propto \min_{K,\zeta,\boldsymbol{\mu}} -\log p(\mathbf{x},\zeta,\boldsymbol{\mu}). \tag{3.6}$$

Typically, computing the MAP estimate is quite difficult, but by choosing the hyperparameter $\alpha = \exp(-\lambda/2\sigma^2)$ and taking the small variance asymptotic limit as $\sigma^2 \to 0$, the optimization reduces to

$$\min_{K,\zeta,\boldsymbol{\mu}} \sum_{k=1}^{K} \sum_{t=1}^{T} \|\mathbf{x}_t - \boldsymbol{\mu}_k\|^2 + \lambda K. \tag{3.7}$$

Note this objective function takes the same form as that of the *k*-means algorithm plus a penalty term for adding new clusters. It is solved straightforwardly by the DP-means algorithm, which iteratively assigns data to the cluster with the nearest mean unless the nearest cluster mean exceeds a threshold distance, controlled by the user-set value λ, upon which a new cluster is generated. Although more complicated extensions to the DP-Means algorithm exist (e.g. [86] [89]), DP-Means requires few user-set parameters and often leads to satisfactory results.

3.3 Online Generalized Product of Experts

In this section, we present Online Generalized Product of Experts (OGPoE), which consists of a partitioning stage, a training stage, and a prediction stage. We present the algorithm design and implementation, and we discuss how the specific combination of algorithms gives OGPoE several advantages compared to the state-of-the-art in online learning.

3.3.1 Algorithm Design

To overcome current limitations in using GPs for online learning, each component of the OGPoE algorithm must be carefully selected. First, the training component should provide a principled method for updating the model inducing point locations and hyperparameters. Previous works either update these values separately [90], require that the hyperparameters be known *a priori* [81], or implicitly assume that the training data is independently, identically distributed (iid) [91]. We select to use streaming sparse GP approximations in order to simultaneously update both the inducing point locations and hyperparameters even when the training data is generated from a time series and is not iid.

Second, in order to grow the model complexity as the amount of training data increases, the partitioning strategy must be simple, easily tune-able, and interpretable, in order to divide data amongst an ensemble of local models. Previous works have used somewhat opaque online partitioning strategies [85][81][84] that make it difficult to predict when a new model will be formed, and consequently more challenging to tune. We select to use the DP-means algorithm, which can be analysed through the lens of small variance asmpytotics [87] to better understand its inherent assumptions and convergence. This simple strategy is regulated by a single user-set parameter that is straightforward to tune.

Lastly, the prediction strategy must be able to fuse predictions from multiple local models, online, without imposing restrictive assumptions on how the models are trained. Many prediction strategies, however, assume all local models share the same hyperparameters ([81] [92]). We therefore select to use the GPoE framework, which allows for each of the individual experts to be trained independently and the aggregate prediction to better capture non-stationary features in the dataset [72]. Further, GPoE does not require the data used for training each expert to be local, and produces good aggregate predictions even when the individual experts are trained on random subsets of data that may significantly overlap [80], as could be the case when using a sparse GP approximation. Together, these features allow OGPoE to build models online that describe data exhibiting complex phenomena such as non-stationarity and heteroscedastic noise.

3.3.2 Implementation

Implementation of OGPoE consists of an initialization phase, followed by online learning and prediction (Fig. 3.1).

Initialization: We assume access to $N_{\text{init}} > M$ samples which are used to initialize our model. When learning a kinematics or dynamics model, for example, these samples might be acquired by sending random, low-magnitude signals to the robot joints. We assign all initial data to a single cluster, which is represented by its mean, $\boldsymbol{\mu}_1 = \frac{1}{N_{\text{init}}} \sum_{i=1}^{N_{\text{init}}} \mathbf{x}_i$. A sparse GP model is then instantiated as in [76] by solving an optimzation problem to determine initial inducing input and hyperparameter values.

Online Training: Data arrives sequentially in small batches of N_{new} samples for processing. Each data point is assigned to either an existing cluster or to a new cluster by solving the optimization problem in Eq. (3.7). Once all data in a batch has been assigned, existing cluster means are updated with the data assigned to that cluster. The assigned data is then used to update the sparse GP approximation associated with each cluster by optimizing the online variational free energy objective as described in Section 3.2.1. If a new cluster is generated, the sparse GP approximation of the nearest existing cluster is used for this optimization. This process can be repeated as long as there are batches of data to process.

For all model training, we empirically find it helpful to add an inequality constraint to the online VFE optimization problem that constrains each of the lengthscales of the squared exponential kernel to not be greater than twice the legnthscale parameter used in the clustering algorithm, $l_i \leq 2\lambda \ \forall i \in \{1, ..d\}$. Intuitively, the kernel function lengthscales, which dictate the correlations between datapoints, should not exceed the diameter of the cluster from which they originate. Note that this constraint does not require the individual kernel lengthscales to be equal to one another, but merely enforces an upper bound.

Prediction: In order to combine predictions from multiple local models on a new input, we use the GPoE framework described in Section 3.2.2. One disadvantage is that for test points far from any training data, the difference between the prior and posterior variance for all experts is nearly zero [79] [92], causing all of the weights to go to zero and the predictive variance of the aggregate model to explode (i.e. as $\beta_i \to 0$, $V_{\text{GPoE}} \to \infty$). Although this is only a minor concern for most online learning problems since predictions are made close to the training data which is revealed incrementally over time, we choose to mitigate this issue by simply normalizing the weights such that $\sum_{i=1}^{K} \beta_i = 1$ for prediction.

User Set Parameters: OGPoE requires little tuning to apply to new problems. We provide the following suggestions for the only two parameters, M and λ, that must be set by the user. It is worth noting that these are distinct from the kernel hyperparameters, θ, which are determined automatically from the data by optimization. M is the number of inducing inputs per sparse GP model that will be used for training. The question of how to choose M, such that $M \ll N$ points are guaranteed to approximate a GP model with tight KL divergence bounds is still an area of active research [93]. Practically, we find that setting the value for M presents a tradeoff between increased computation time for training and better modeling accuracy. Therefore, we advise choosing M to be as large as possible, while still maintaining a model update rate that is suitable for the particular application. The other parameter that must be set by the user is the cluster lengthscale, λ, used in the DP-means algorithm for data partitioning. Because of the similarity between DP-means and k-means, if a subset of data is available for offline computation, we find that clustering this data with k-means and then setting λ to be the average radius of the resultant clusters, is an effective initial guess for this parameter that may be further tuned as needed. Finally, although the batch size of training data is typically not controlled by the user since data is streaming at a fixed rate and processed as fast as possible, N_{new} can be set for simulated experiments. In this case, larger values of N_{new} increase the training time per batch of data, but can lead to better fits ([77]).

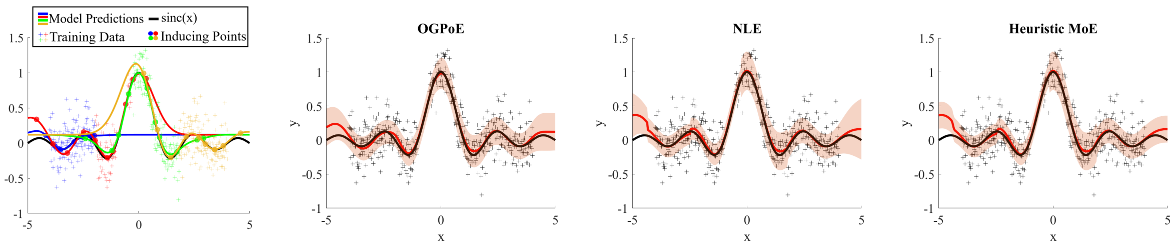
3.4 Simulated Evaluation

We assess the performance of OGPoE through numerical experimentation. We present a simple learning problem to demonstrate the effectiveness of OGPoE for multiple model prediction, and we compare its performance to other popular modeling approaches on the SARCOS arm dataset.

3.4.1 Toy Example

We consider the problem of learning an approximation to the 1-D function $y = \text{sinc}(x) + \varepsilon$, with $\varepsilon \sim \mathcal{N}(0, 0.04)$. We generate 500 samples from this function for training, which we process sequentially from $x = -4$ to $x = 4$ (excluding $x \in [-0.5, 0]$) in batches of $N_{\text{new}} = 20$ samples, in order to simulate online training. For testing, we compare against the noise-free function $\bar{y} = \text{sinc}(x)$ for $x \in [-5, 5]$. With $\lambda = 0.5$ and $M = 10$, online training produces 4 local models with inducing point locations and individual predictions shown in Fig. 3.2a. No one local model learns the entire latent function — with an RMSE of 0.091 for the best local model — necessitating an ensemble prediction strategy. There is overlap between the models, which can be attributed to the fact that each new local model is initialized using the nearest, existing local model, and the optimization used to update the locations of the local model inducing points can find values outside the region of the training data [77][84]. Despite the overlap, Fig. 3.2b shows that OGPoE can effectively combine local model predictions and capture the underlying function (0.059 RMSE), because it does not assume that the individual models cover independent regions of the input space. Instead, OGPoE effectively blends contributions from proximal models in an aggregate prediction.

Other ensemble prediction strategies are shown in Fig. 3.2c and Fig. 3.2d. NLE, for



(a) (b) (c) (d)

Figure 3.2. (a) Sparse, local GP models are learned using data, '+', drawn from a noisy, unknown function as detailed in Section 3.4.1. The predictions from these local models are combined using (b) OGPoE, (c) NLE, and (d) a heuristic MoE approach. The predictive mean of each aggregation method is shown in red along with the one standard deviation confidence bounds in orange.

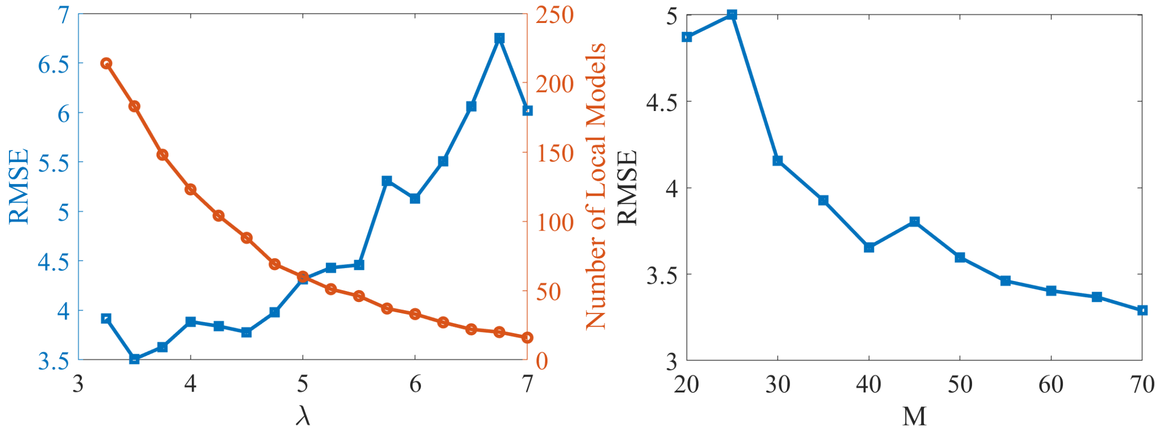
example, uses the predictive mean and covariance of the local model nearest the predictive input (Fig. 3.2c). Consequently, its prediction contains artificial discontinuities where there is a transition between the local models used, despite the smoothness of the underlying function (0.085 RMSE). Heuristic MoE (Fig. 3.2d) trains an additional meta GP model used to weight the predictive mean from each local model [81] [84]. Because this method combines the predictions of multiple models, it avoids the discontinuities associated with the NLE method but at the cost of additional training time. Additionally, because the local model weights are exponential in the distance between the model center and the predictive input, most of the weight for prediction is usually concentrated on a single model, producing a predictive output that is not substantially better than prediction with NLE (0.084 RMSE).

3.4.2 SARCOS Arm Benchmark

In this section, we study the effects of varying OGPoE's user-set parameters, λ and M, and compare our proposed algorithm to other modeling approaches, using the SARCOS arm dataset[1]. This publicly available dataset is frequently used to benchmark the performance of offline [71] [79] [81] and online [94] [84] [85] modeling approaches, making it a suitable standard to compare against without having to reproduce the results of many previous works. The objective is to learn an inverse dynamics mapping from a 21 dimensional input joint space to the first of 7 output joint torques, using a dataset that consists of 48,933 samples (44,484 for training and 4,449 for validation)[71] [79] [85].

First we assess the effect of the clustering lengthscale parameter, λ. We set $M = 50$ and process data sequentially in batches of $N_{new} = 100$ samples for training, as is done in [85], and then evaluate the prediction of the learned model on the validation dataset. We repeat this simulation for $\lambda \in [3.25, 7]$ (Fig. 3.3a). As expected, the performance degrades as lambda is increased since the number of local models generated, and consequently the total number of inducing points used to capture the data, decrease. Next, to assess the effect of M on learning

[1] Dataset available at: http://www.gaussianprocess.org/gpml/data/



(a) **(b)**

Figure 3.3. Plots depicting training results on the SARCOS arm benchmark for different values of the user-set parameters (a) λ and (b) M.

performance, we fix λ to the best value from Fig. 3.3a and repeat the experiment, this time varying M from 20 to 70. The results, shown in Fig. 3.3b, depict how performance improves with increasing M, although with diminishing returns as M becomes large.

Finally, we report the results of other popular modeling approaches on the SARCOS dataset in Fig. 3.4. The value of M used for OGPoE was set to 50 in order directly compare against the methods studied in [85], while λ was set to 3.5, which was the best result from the experiment shown in Fig. 3.3a. Looking at the performance of LGPR [81], which requires offline optimization of the hyperparameters, we would expect it to significantly outperform OGPoE, where the hyperparameters are optimized online. However, we note that we achieve nearly comparable results, with a RMSE of 3.507 compared to 3.195 reported in [85].

The online methods we compare to are the well-known LWPR [94], as well as SOLAR-GP [84] and WGPR [85], which are the most similar algorithms to OGPoE. Our results show that we outperform these methods on this benchmark, even for a wide range of values of λ (Fig. 3.3a). Because SOLAR and WGPR leverage the same sparse GP approximation strategy as OGPoE, the difference in performance can likely be attributed to differences in how data is partitioned between local models and how combined predictions are generated. OGPoE's data partitioning scheme differs in that data assignment is independent of the parameters used

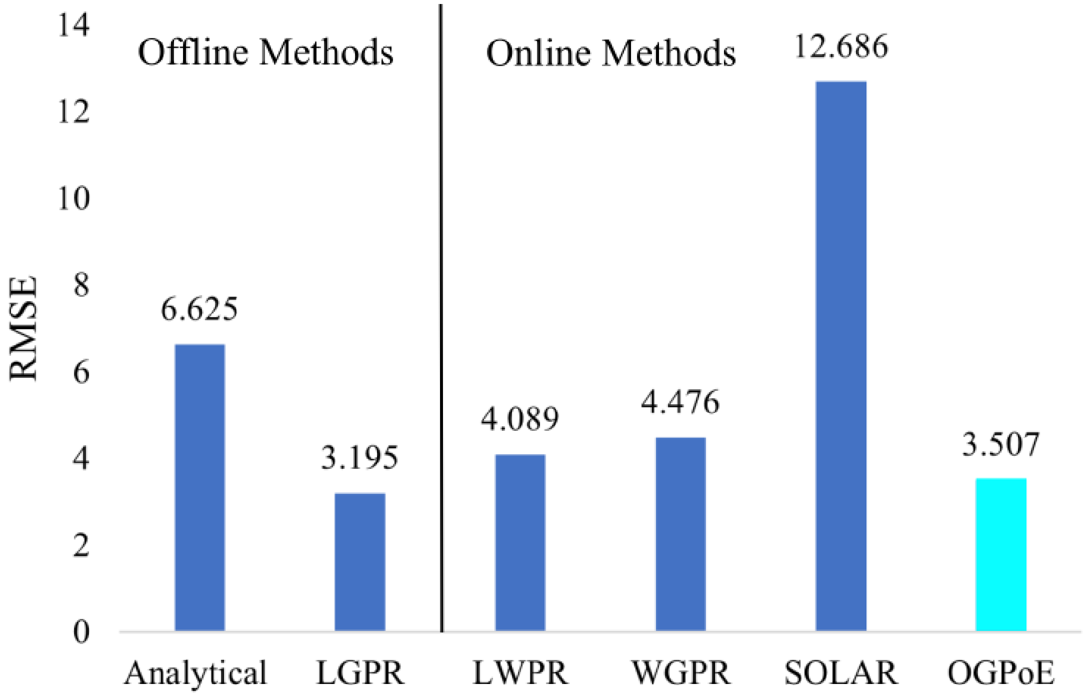


Figure 3.4. SARCOS benchmark results demonstrating OGPoE achieves excellent performance relative to the state of the art in online learning methods. We report the results of SOLAR, WGPR, and LGPR from [85] and the analytical model and LWPR from [94].

to define the local models, which change over time. The prediction framework employed by OGPoE more effectively combines information from multiple local models, leading to improved aggregate prediction as discussed in Section 3.4.1.

3.5 Hardware Evaluation

In this section, we use OGPoE to learn the forward kinematics of a concentric tube robot (CTR) from streaming data. CTRs are a class of continuum robot made up of precurved, flexible, telescoping tubes that when translated and rotated relative to one another, interact in bending and torsion to reach different equilibrium configurations [95] [14]. Mechanical models based on the theory of Cosserat Rods [96] have been developed to describe their forward kinematics, where the tip position is computed as a function of the rotational and translational positions of each of the tubes [18]. Typically, these models assume that all tubes share a common centerline and ignore the effects of friction and clearance between the tubes. While these assumptions lead to a

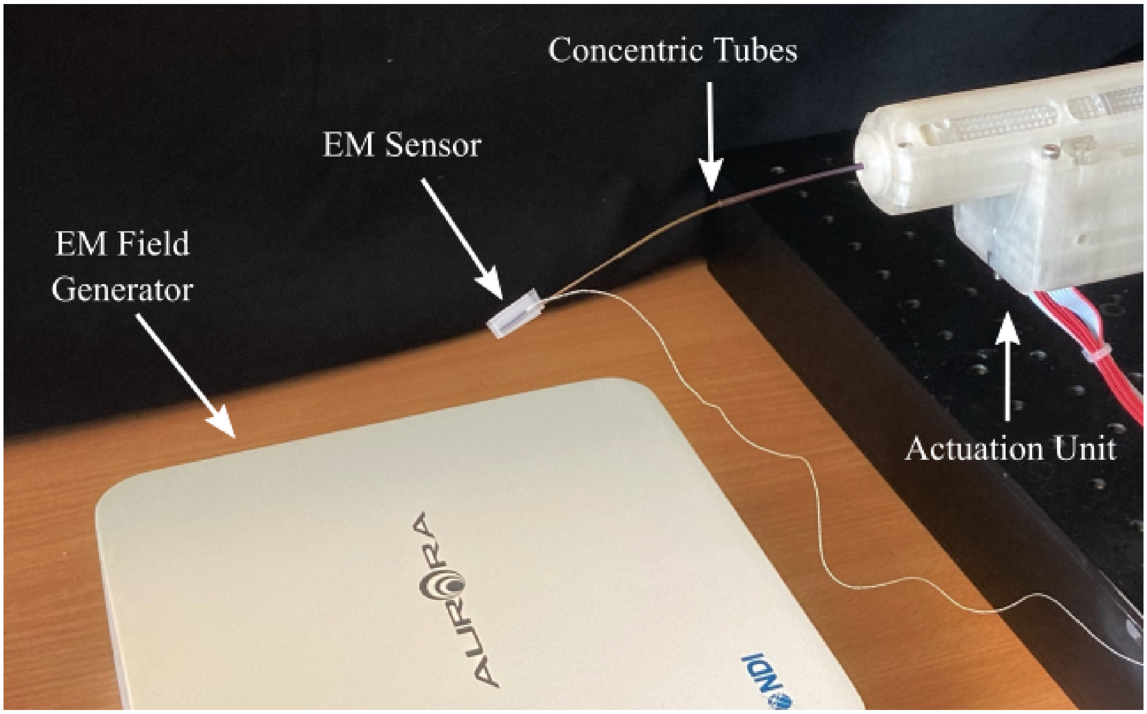


Figure 3.5. The experimental setup consists of a 3-tube CTR and associated actuation unit from [97]. An EM sensor is attached to the CTR tip in order to track its position.

tractable set of equations for the kinematics, they also contribute to modeling inaccuracies. We approach learning kinematics for these robots in two ways: one in which the forward kinematics must be learned completely from scratch, and another in which we augment the learned model with a mean function derived from first principles.

3.5.1 Experimental Setup

The 3-tube CTR and associated actuation unit from [97] were used in this experiment (Fig. 3.5). An Aurora electromagnetic (EM) tracking system (NDI) is used to measure the 3-D position of an EM sensor (0.92 mm outer diameter) attached to the tip of the robot. Joint and EM sensor positions are sent to the OGPoE algorithm to be used for training via the Robot Operating System (ROS) [98], which is used to enable parallel processes for prediction and training [84]. An open-loop joint space trajectory is also published over ROS to set the joint positions and to predict the robot tip position from the learned model.

For our hybrid learning approach, which involves augmenting the learned prediction with an analytical model, we follow the model derivation of [18]. This approach amounts to solving a set of ordinary differential equations subject to boundary conditions (BCs) imposed by the joint values of the tubes. For our CTR, these equations have a unique solution for every set of BCs dictated by the robot's joint configuration, defining a unique mapping that we use as our analytical kinematic model. Because this mapping is deterministic, we may incorporate it into our learning problem by adopting a zero-mean GP prior over the *difference* between the tip position predicted by this model and the true tip position observations measured by the EM sensor for learning. For prediction, we combine the output of the analytical model with this learned error function. To save computation time, we follow the example of [94] and [84] by using the same learned model to predict each component of the tip position.

3.5.2 Online Learning of Robot Kinematics in Free Space

In this experiment, the CTR is commanded to follow an open-loop joint-space trajectory consisting of 600 points that was designed offline to cover a significant region of the robot workspace. At each timestep, we predict the robot tip position for the next joint configuration using one of three models — the Cosserat-rod-based kinematics model, the OGPoE model,

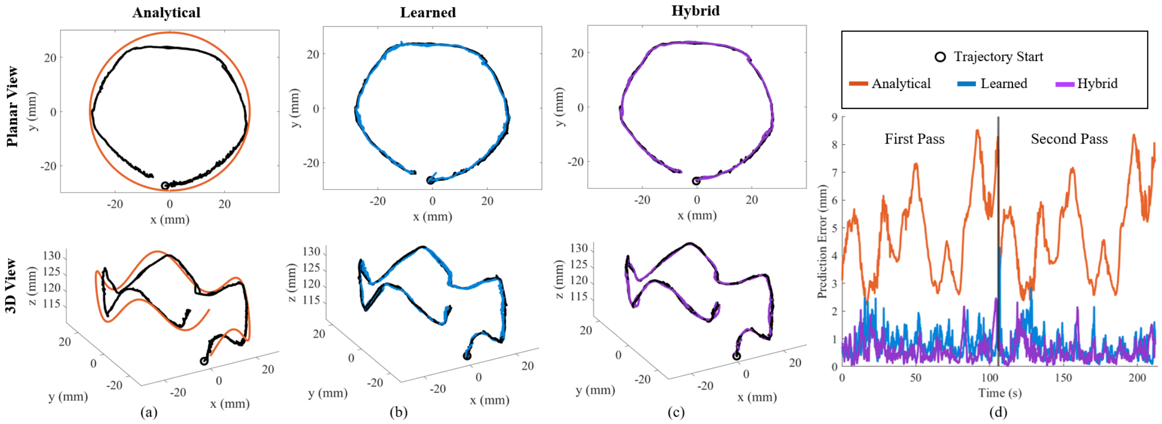


Figure 3.6. Results of the online kinematics learning task using (a) an analytical model, (b) a model learned with OGPoE, and (c) a hybrid of the two approaches, where OGPoE is used to learn the error in the analytical model. The plots on the left show the predicted and measured tip positions from two different angles for each of the models used, and the plot on the right shows the prediction error over time for all three models.

or the hybrid model. To initialize the learned models, we sample random joint configurations in close joint space proximity to the robot's home configuration, which results in small tip motions. Specifically, we draw 50 samples for each joint from $\mathscr{U}(q_0 - 2.5\text{ mm}, q_0 + 2.5\text{ mm})$ for translational joints and from $\mathscr{U}(q_0 - 5^\circ, q_0 + 5^\circ)$ for rotational joints, where q_0 is the home position of the joint and $\mathscr{U}(a, b)$ is a uniform random distribution with lower limit a and upper limit b. These initial models are updated online according to the OGPoE framework as data arrives over the course of the trajectory and used to predict on new inputs just outside the distribution of training data.When the robot reaches the end of the trajectory, it returns to the starting position and then follows the same trajectory again to assess the behavior of the learned models when re-entering previously visited regions of the state space. For both the hybrid and purely learned OGPoE models, we set $\lambda = 3.75$ and $M = 25$, which we found to be acceptable after minimal tuning.

The prediction results in Fig. 3.6 show that there is significant error in the analytical model predictions (4.8 mm ± 1.52 mm) primarily in predicting the x and y components of the tip position. This error is expected for these models [18][99], and a large part is likely due to ignoring clearance between the tubes of the robot, which causes the true shape to be straighter than the Cosserat rod model predicts [100]. Consequently, the path traced by the robot tip has a smaller radius than the one predicted by the model (Fig. 3.6a). Both of the learned models, on the other hand, achieve much lower mean absolute error for predicting the tip position — 0.79 mm ± 0.48 mm and 0.58 mm ± 0.4 mm for the purely learned and hybrid models, respectively — demonstrating the feasibility of using OGPoE in an online learning scenario. The prediction error of the learned models decreases slightly between the two passes through the trajectory, while the number of local models stays constant, indicating that these learned models retain information about regions of the state space they have already visited. While both approaches achieve high accuracy predictions, the hybrid approach exhibits a slightly smaller average error due to incorporation of a good, physics-based, analytical mean function.

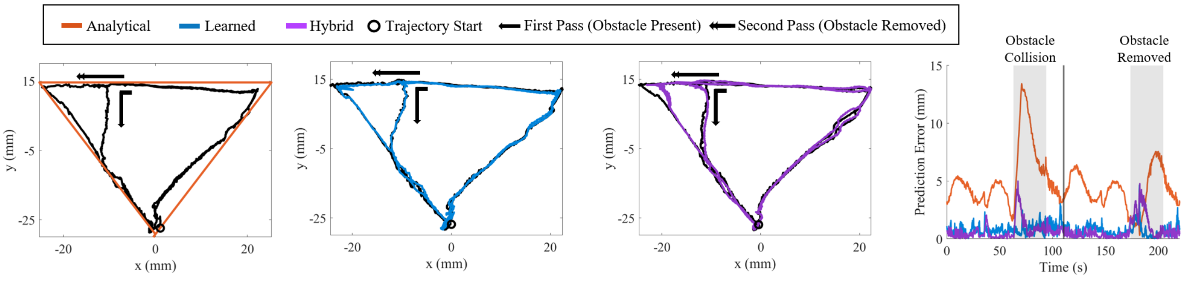
The success of OGPoE can be attributed to the fact that it updates a model online with data from the trajectory that is being followed, allowing for good predictions on new, nearby inputs. On average, the model update rate is 0.3 s and the prediction rate is 106 Hz, which is sufficient for surgical procedures where the robot speed will not exceed a few cm/s [101]. The benchtop setup can also be substantially improved with specialized hardware and software.

3.5.3 Online Learning of Robot Kinematics with Dynamic Loading Conditions

Next we demonstrate the ability of OGPoE to learn the kinematics of a CTR when the robot is subjected to an unknown and changing environmental constraint. This ability is particularly important for CTRs, since many of their intended applications involve navigation through dynamic environments, yet there is a notable lack of proposed solutions in the literature due to the difficulty of the problem.

We initialize the robot as in Section 3.5.2 and command it to follow a new open-loop trajectory, twice. This time however, an un-modeled obstacle is placed in the path of the robot, such that the robot tip is significantly deflected when it comes into contact with the obstacle. Before the robot traverses the trajectory a second time, the obstacle is removed, resulting in different tip position measurements that conflict with the data gathered on the first pass. Figure 3.7 shows the robot tip position predicted by the analytical, learned, and hybrid models, along with the associated prediction errors.

As seen in Fig. 3.7(a), the presence of the obstacle causes large errors in the analytical model prediction (4.73 ± 2.23 mm average, 13.42 mm maximum), whereas the OGPoE-based methods achieve much more accurate results (learning: 0.75 ± 0.49 mm average, 4.67 mm maximum; hybrid: 0.72 ± 0.84 mm average, 5.00 mm maximum). It is also interesting to note that the error in the hybrid modeling approach has a small, transient increase when the robot encounters the obstacle (presumably due to the inaccuracy of the analytical model) and again when the robot traverses the region of the state space where the obstacle has been removed



(a) (b) (c) (d)

Figure 3.7. Predicted tip position of the CTR following a planar, triangle-shaped path using an (a) analytical (b) learned and (c) hybrid model as well as (d) the prediction errors over time. On the first pass through the trajectory, an obstacle blocks the robots path, causing a large deviation from the analytical model. On the second pass through, the obstacle is removed.

(where the previously learned deviation from the analytical model needs to be un-learned). Nonetheless, the low average prediction error of both OGPoE-based methods demonstrates the applicability of this method for learning CTR kinematics under these challenging conditions.

3.6 Discussion

In conclusion, we presented OGPoE and illustrated its effectiveness starting with a simple 1-D example. We verified that OGPoE achieves excellent results on the SARCOS arm benchmark relative to the state of the art, making it a potentially useful tool for complex online learning problems. Finally, we demonstrated the ability of OGPoE to learn the kinematics for a continuum robot in an online setting, both with and without prior model information and even in the presence of changing, unknown loading conditions.

There are several directions for future work that can further expand the applicability of OGPoE. For example, the development of a method to bound the prediction error of the aggregate model learned by OGPoE can enable its use in online learning applications that require guaranteed safety, as have been recently studied using standard GPs with known hyperparameters [102]. Additionally, dimensionality reduction techniques could potentially be used with OGPoE to do effective online learning for problems with very high dimensional inputs. Finally, OGPoE could be applied to many other robotics problems, including learning and adapting a model for use with inverse kinematics methods to enable real-time control in unknown environments.

3.7 Acknowledgements

This chapter is, in part, a reprint of material from C. Watson, T.K. Morimoto, ”*Learning Non-Parametric Models in Real Time via Online Generalized Product of Experts*.” IEEE Robotics and Automation Letters, vol. 7, no. 4, pp. 9326-9333, 2022. The dissertation author is the primary author of this paper.

Chapter 4

Low Frequency Disturbance Rejection for Cable-Driven Continuum Robots without Integrated Sensing

4.1 Introduction

In this chapter, we develop a new dynamics model for a cable-driven continuum robot. This new model unlocks tools from control theory developed decades earlier for serial arm robots with elastic joints, allowing us to solve a disturbance rejection problem using only partial state feedback.

4.1.1 Related Works

Theoretically, the backbone of a continuum robot can take the shape of an arbitrary, oriented curve in space. Many researchers have devoted significant effort to developing models that are able to capture this full complexity, the most successful of which employ the theory of Cosserat rods [103]. The trade-off posed by the expressivity of these methods is much higher computational cost, which makes real-time control challenging and is considerably difficult to extend to predict dynamic robot behavior [17].

For many applications of interest, however, kinematic models based on the assumption that a continuum robot bends in a shape defined by piece-wise constant curvature arcs have been shown to be reasonably accurate [104][105]. The computational complexity of these models is trivial compared to that of Cosserat rod-based approaches, and they can be extended to capture how external forces from actuators or the environment affect the robot shape [31] [106]. The key insight of these works is to develop a quasi-static model for a system that is mechanically analogous to the continuum robot. These works consider the actuating cables to be springs with linear stiffness and the robot itself to be a torsional spring connected to a linear spring (see Fig 4.1). Displacement of the linear spring represents compression or elongation of the robot backbone while displacement of the torsional spring represents a change in the robot bend angle or curvature.

In addition to being an intuitive framework for understanding continuum robot motion, these so-called kineto-static models capture an important phenomenon in that the robot cable

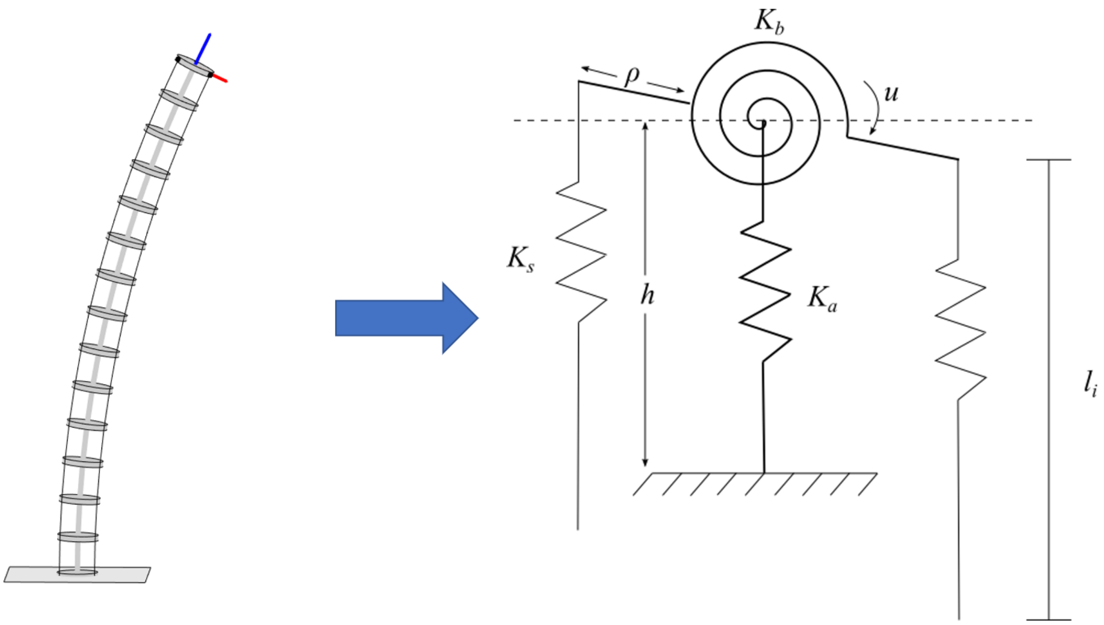


Figure 4.1. The continuum robot pictured on the left can be approximated by the analogous mechanical system on the right. An axial spring represents resistance to changes in the robot length from some equilibrium value. Torsional springs represent resistance to robot bending that is proportional to its curvature. Although we only depict this bending resistance in one direction here, it is possible to augment this model with another torsional spring to capture bending in another direction. The actuating cables are also treated as stretchable spring elements.

lengths, which define its shape are geometric quantities that do not map uniquely to actuator positions. The non-uniqueness of the mapping between cable lengths and actuator positions is due to the fact that the cables themselves can stretch under to applied forces. This means that although there is a unique robot configuration for every set of cable lengths, there is not a unique robot configuration for every set of actuator positions. The kineto-static model resolves this issue by modeling both the robot *and* its cables as linear spring elements under uniform tension or compression.

4.1.2 Contributions

In this work, we extend the kineto-static models described above to capture the dynamics of cable-driven continuum robots. We are motivated by the fact that dynamic models are fundamental in control theory and robotics, which would allow us to leverage algorithms developed on other, similar systems for use with these continuum robots. Specifically, we are interested in the problem of disturbance rejection when only partial state measurements are available. This is an important problem that must be addressed if continuum robots are to be used in tasks like ablation or knot-tightening [45]. To address this challenge, this work presents the following contributions:

- A new, constant curvature dynamic model. This model is the simplest to date for cable-driven continuum robots. It is intuitive to use and captures important important aspects of a real world dynamic system in its prediction, including oscillation and delay.
- An observer-based disturbance estimator. Our approach offers a number of important advantages over previous works including provable convergence under reasonable assumptions, use of only actuator position and velocity measurements, as well as analytical insight into the estimation problem that would otherwise be obscured with a purely learning-based method.
- A control strategy that leverages the above estimator to reject low-frequency disturbances acting on the robot body.

We detail the development of the model in the next section, followed by a description of how it can be used to design an observer-based control scheme for disturbance rejection. Finally we validate these results on a hardware and conclude with some potential directions for future work.

4.2 Modeling

In this section, we detail our modeling approach and discuss its unique aspects as compared to other approaches in the literature. First, we detail the kinematics of our robot based on a constant curvature assumption. Then we discuss how this assumption can be incorporated into a dynamics model for an equivalent mechanical system that approximates the true robot motion.

4.2.1 Kinematics

There are multiple ways to parameterize a constant curvature arc [16][107], but we find that using the components of the robot's curvature vector is the most convenient due to the linearity of the equations and its singularity-free representation of the robot's straight configuration [106][107]. In this work, we will focus on only a single constant curvature segment, but note that extensions to multiple segments is possible. Specifically, we consider a configuration variable, $\mathbf{x}$, as in [107] which defines the shape of the robot,

$$\mathbf{x} = [u,\ v,\ h]^T. \tag{4.1}$$

The two components of the robot curvature are u and v, while h captures the robot length. Considering the length of the robot as a configuration variable is only necessary if it changes significantly over time due to compressibility of the robot backbone, otherwise it may be considered constant and omitted.

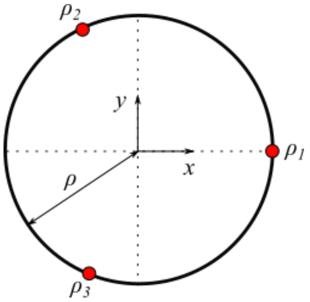


Figure 4.2. Three cables spaced evenly around the circumference of a continuum robot cross section. The cross section has radius ρ and the cables are attached at locations ρ_1, ρ_2, and ρ_3.

From this representation of the robot shape, it is also straightforward to determine the tip position and orientation of the robot as a homogeneous transformation matrix

$$g = e^{(\hat{\boldsymbol{\omega}} h)}, \tag{4.2}$$

where

$$\hat{\boldsymbol{\omega}} = [0,\ 0,\ 1,\ u,\ v,\ 0]^T. \tag{4.3}$$

The kinematics relating the cable lengths of the robot to its configuration is similarly convenient to express using this parameterization. For a three cable robot with cable lengths $\mathbf{l} = [l_1, l_2, l_3]^T$ routed evenly around a cross section of the robot (see Fig 4.2), the kinematics can be written as in [107]:

$$\begin{aligned} l_1 &= h - \rho v \\ l_2 &= h + \frac{1}{2}\rho(v + \sqrt{3}u) \\ l_3 &= h + \frac{1}{2}\rho(v - \sqrt{3}u) \end{aligned} \tag{4.4}$$

$$\begin{aligned} h &= \frac{1}{3}(l_1 + l_2 + l_3) \\ u &= \frac{l_2 - l_3}{\sqrt{3}\rho} \\ v &= \frac{h - l_1}{\rho}, \end{aligned} \tag{4.5}$$

where ρ is the robot radius. The differential kinematics, relating small changes in cable lengths to small changes in robot configuration can be determined straightforwardly by taking derivatives,

$$\dot{\mathbf{l}} = \mathbf{J}\dot{\mathbf{x}} = \frac{\partial \mathbf{l}}{\partial \mathbf{x}}\dot{\mathbf{x}} \tag{4.6}$$

The kinematic equations for robots with additional cables can be found in [16][107]. Note that more complex cable routing patterns, such as spiral ones described in [103], are not well modeled by the constant curvature assumption and will not be considered here.

4.2.2 Dynamics

Our approach to dynamic modeling assumes that there are linear and torsional springs representing the robot body (Fig 4.1) which have a mass and rotational inertia such that we can write the kinetic energy of the robot as

$$KE_{robot} = \frac{1}{2}\dot{\mathbf{x}}^{\mathbf{T}}\mathbf{M}\dot{\mathbf{x}}. \tag{4.7}$$

We consider each of the springs to be effectively decoupled from one another such that the mass matrix, $\mathbf{M}$, is diagonal. Although this is an approximation, we find empirically that it yields good results (see Section 4.4.3). Ignoring gravity, the potential energy of the robot depends on the linear and torsional spring displacements

$$PE_{robot} = \frac{1}{2}(\mathbf{x} - \mathbf{x_{eq}})^{\mathbf{T}}\mathbf{K_b}(\mathbf{x} - \mathbf{x_{eq}}), \tag{4.8}$$

where $\mathbf{x_{eq}}$ is the unstressed configuration of the robot body and $\mathbf{K_b}$ is a diagonal stiffness matrix of spring constants for each spring element of the robot body. We consider the cables of the robot to be mass-less, but that the motors actuating the cables have some rotational inertia, $\mathbf{\Phi}$, which gives a potential energy for the actuation system of

$$KE_{actuation} = \frac{1}{2}\dot{\mathbf{q}}^{T}\mathbf{\Phi}\dot{\mathbf{q}}. \tag{4.9}$$

The potential energy of the actuation system depends on the cable stretch, which is a function of both the actuator positions and the cable lengths, which are themselves a function of the robot configuration (Eq. 4.4). This gives a potential energy of

$$PE_{actuation} = \frac{1}{2}(\mathbf{q} - \mathbf{l}(\mathbf{x}))^{\mathbf{T}}\mathbf{K_s}(\mathbf{q} - \mathbf{l}(\mathbf{x})), \tag{4.10}$$

where $\mathbf{K_s}$ is a diagonal matrix of cable stiffnesses. Note that these actuator positions are typically measured by an angular encoder connected to the pulley around which the cables are wound. Here we convert these measurements using $\mathbf{q} = r\boldsymbol{\theta}$, where r is the pulley radius and $\boldsymbol{\theta}$ are the encoder measurements in radians, and instead consider the actuator positions $\mathbf{q}$ to have units of length. Similarly we consider commanded input tensions to the cables, $\mathbf{u}$, rather than motor torques, $\boldsymbol{\tau} = r\mathbf{u}$ for ease of notation.

Using these quantities, the difference between the kinetic and potential energies is

$$\mathscr{L} = KE_{robot} + KE_{actuation} - PE_{robot} - PE_{actuation}. \tag{4.11}$$

Using this quantity, we can write the equations of motion for our system as

$$\begin{aligned} \frac{d}{dt}\frac{\partial \mathscr{L}}{\partial \dot{x}_i} - \frac{\partial \mathscr{L}}{\partial x_i} &= -\frac{\partial \mathscr{F}}{\partial \dot{x}_i} \\ \frac{d}{dt}\frac{\partial \mathscr{L}}{\partial \dot{q}_j} - \frac{\partial \mathscr{L}}{\partial q_j} &= Q_j, \end{aligned} \tag{4.12}$$

where i denotes the index of the configuration variable and j denotes the index of the motor. The term $\frac{\partial \mathscr{F}}{\partial \dot{x}_i}$ is included to capture damping that opposes motion of the robot body. It is important to include such a term because real world continuum robots are not made of perfectly elastic materials. We model this damping as proportional to the velocity of the robot, $\frac{\partial \mathscr{F}}{\partial \dot{x}_i} = d_i x_i$, which can be thought of as including linear and torsional dashpots to our analagous mechanical system. The term Q_i captures inputs to the system from the motors. Specifically, $Q_j = u_j - f_{m,j}$, where u_j is the tension command sent to the j-th motor and $f_{m,j}$ is the motor friction opposing that input. Note that there are many possible ways to model this friction term, which is a choice left to the user as a design decision.

The above derivatives yield the following system of equations:

$$\begin{aligned}\mathbf{M}\ddot{\mathbf{x}} + \mathbf{D}\dot{\mathbf{x}} + \mathbf{K_b}(\mathbf{x} - \mathbf{x_{eq}}) &= \mathbf{J^T}\mathbf{K_s}(\mathbf{q} - \mathbf{l}(\mathbf{x})) \\ \mathbf{\Phi}\ddot{\mathbf{q}} + \mathbf{K_s}(\mathbf{q} - \mathbf{l}(\mathbf{x})) &= \mathbf{u} - \mathbf{f_m}.\end{aligned} \tag{4.13}$$

Note that these equations closely resemble those for other classical robotic systems such as serial robot arms with elastic joints [108] and have exactly the same form as the equations for cable-driven parallel mechanisms [109]. We believe that this similarity is an important feature of our model that will allow for the application of mature control and estimation techniques that have been developed for these more classical systems to be applied to continuum robots.

4.3 Observer-Based Control

This section explains how the dynamic model derived above can be used to design an observer that estimates the robot state using only partial information –specifically measurements of the actuator positions and velocity. We also explain the limitations of the observer, as well as practical challenges with using it directly in a control loop. We address the latter issue by developing a reference input-based control strategy that can take advantage of the observer to compensate low-frequency disturbances effecting the robot.

4.3.1 High Gain Observer Design

Our observer design is motivated by the fact that our system dynamics are similar to that of a serial robotic arm with elastic joints [108]. The problem of controlling these robots using only partial feedback was first explored in [110] and has seen many variations since (e.g. [111] [112]). We adopt a very similar approach to [110] in this work.

We assume that we have access to measurements of the actuator positions and velocities

$\mathbf{y} = [\mathbf{y_1}, \mathbf{y_2}]^{\mathbf{T}} = [\mathbf{q}, \dot{\mathbf{q}}]^{\mathbf{T}}$. Consider a change of variables

$$\mathbf{z} = \begin{bmatrix} \mathbf{K_s^{-1}}\boldsymbol{\Phi}\dot{\mathbf{q}}, & \mathbf{l}, & \dot{\mathbf{l}} \end{bmatrix}^T, \tag{4.14}$$

which leads to a reduced-order system

$$\begin{aligned}
\dot{\mathbf{z}}_1 &= \mathbf{z_2} - \mathbf{y_1} + \mathbf{K_s^{-1}}(\mathbf{u} - \mathbf{f_m}(\mathbf{y_1}, \mathbf{y_2}, \mathbf{u})) \\
\dot{\mathbf{z}}_2 &= \mathbf{z_3} \\
\dot{\mathbf{z}}_3 &= \boldsymbol{\psi}(\mathbf{z_2}, \mathbf{z_3}, \mathbf{y_1}) \\
\mathbf{y}_2 &= \boldsymbol{\Phi}^{-1}\mathbf{K_s}\mathbf{z_1},
\end{aligned} \tag{4.15}$$

where

$$\boldsymbol{\psi}(\mathbf{z_2}, \mathbf{z_3}, \mathbf{y_1}) = \mathbf{J}\mathbf{M}^{-\mathbf{1}}(\mathbf{J^T}\mathbf{K_s}(\mathbf{y_1} - \mathbf{z_2}) - \mathbf{D}\mathbf{J}^{\#}\mathbf{z_3} - \mathbf{K_b}\mathbf{J}^{\#}\mathbf{z_2}). \tag{4.16}$$

Note that we make use of the relationship $\ddot{\mathbf{l}} = \mathbf{J}\ddot{\mathbf{x}} + \dot{\mathbf{J}}\dot{\mathbf{x}}$ in order to get equations for the cable accelerations, but for our particular parameterization of the robot curvature, the matrix $\mathbf{J}$ is constant. We can rewrite these equations as

$$\begin{aligned}
\dot{\mathbf{z}} &= \mathbf{A}\mathbf{z} + \mathbf{g}(\mathbf{z}, \mathbf{y}, \mathbf{u}) \\
\mathbf{y}_2 &= \boldsymbol{\Phi}^{-1}\mathbf{K_s}\mathbf{C}\mathbf{z}
\end{aligned} \tag{4.17}$$

where

$$\begin{aligned}
\mathbf{A} &= \begin{bmatrix} \mathbf{0} & \mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{I} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix} \\
\mathbf{C} &= \begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} \end{bmatrix}
\end{aligned} \tag{4.18}$$

and

$$\mathbf{g}(\mathbf{z},\mathbf{y},\mathbf{u}) = \begin{bmatrix} \mathbf{K_s^{-1}}(\mathbf{u}-\mathbf{f_m})-\mathbf{y_1} \\ \mathbf{0} \\ \boldsymbol{\psi}(\mathbf{z_2},\mathbf{z_3},\mathbf{y_1}) \end{bmatrix}. \tag{4.19}$$

The matrix pair $(\mathbf{A},\mathbf{C})$ is in block observer canonical form, meaning it is possible for us to choose a matrix $\mathbf{L}$ such that the eigenvalues of $\mathbf{A}-\mathbf{LC}$ are in the left half complex plane. In order to construct our observer, we must also assume that there exist two, positive, known constants M and N that bound the cable velocities and the elastic displacements

$$\begin{aligned} ||\dot{\mathbf{l}}|| = ||\mathbf{z_3}|| &< M \\ ||\mathbf{q}-\mathbf{l}|| = ||\mathbf{y_1}-\mathbf{z_2}|| &< N. \end{aligned} \tag{4.20}$$

These bounds do not need to be tight and are reasonable to assume in practice given the limitations of real-world hardware. We can then define the following high gain observer for the system:

$$\begin{aligned} \dot{\hat{\mathbf{z}}} &= (\mathbf{A}-\Gamma\mathbf{LC})\hat{\mathbf{z}}+\mathbf{g}(\bar{\mathbf{z}},\mathbf{y},\mathbf{u})+\Gamma\bar{\mathbf{L}}\mathbf{y_2} \\ \bar{\mathbf{z}}_3 &= M\frac{\hat{\mathbf{z}}_3}{||\hat{\mathbf{z}}_3||}\text{sat}\left(\frac{||\hat{\mathbf{z}}_3||}{M}\right) \\ \bar{\mathbf{z}}_2 &= \mathbf{y_1}-N\frac{\mathbf{y_1}-\hat{\mathbf{z}}_2}{||\mathbf{y_1}-\hat{\mathbf{z}}_2||}\text{sat}\left(\frac{||\mathbf{y_1}-\hat{\mathbf{z}}_2||}{N}\right) \end{aligned} \tag{4.21}$$

where the high gain matrix, Γ, is defined as

$$\Gamma = \begin{bmatrix} \gamma\mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \gamma^2\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \gamma^3\mathbf{I} \end{bmatrix}, \tag{4.22}$$

for some scalar $\gamma \geq 1$, and the function $sat(\cdot)$ is defined as

$$\text{sat}(x) = \begin{cases} x & \text{if } |x| \leq 1 \\ 1 & \text{if } x > 1 \end{cases}, \tag{4.23}$$

and $\bar{\mathbf{L}} = \mathbf{L}\mathbf{K_s^{-1}}\boldsymbol{\Phi}$. The saturation function serves the purpose of enforcing a Lipschitz bound on the nonlinearities [110] which can help to mitigate the effects of peaking in the estimator. The estimation error for the observer, $\mathbf{e} = \mathbf{z} - \hat{\mathbf{z}}$, evolves according to

$$\dot{\mathbf{e}} = (\mathbf{A} - \boldsymbol{\Gamma}\mathbf{L}\mathbf{C})\mathbf{e} + \mathbf{g}(\mathbf{z}, \mathbf{y}, \mathbf{u}) - \mathbf{g}(\bar{\mathbf{z}}, \mathbf{y}, \mathbf{u}) \tag{4.24}$$

and allows us to state the following theorem about the stability of the observer:

Theorem 1: There exists a constant γ_0 such that the error, $\mathbf{e}(t)$, is exponentially stable for any choice of $\gamma > \gamma_0$ and bounded input $\mathbf{u}$.

The proof follows the one detailed in [110], which relies on the fact that the nonlinear term in Eq. 4.24, $\boldsymbol{\delta}$, has the form

$$\boldsymbol{\delta} = \mathbf{g}(\mathbf{z}, \mathbf{y}, \mathbf{u}) - \mathbf{g}(\bar{\mathbf{z}}, \mathbf{y}, \mathbf{u}) = \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \boldsymbol{\psi}(\mathbf{z_2}, \mathbf{z_3}, \mathbf{y_1}) - \boldsymbol{\psi}(\bar{\mathbf{z}}_2, \bar{\mathbf{z}}_3, \mathbf{y_1}) \end{bmatrix}. \tag{4.25}$$

Because of the saturated estimates used in the observer and the assumptions that the cable velocities and elastic displacements are bounded, it is possible to show that the error in this term is also bounded and that Eq. 4.24 converges to the origin. Although the observer proposed here estimates the cable lengths, it is straightforward to use these to determine the configuration of the robot by solving the forward kinematics as in Eq. 4.5. Ultimately, this convergence result is significant for continuum robots because most of the estimation and control methods described for these robots in the literature do not provide guarantees regarding their performance. As researchers look to extend the space of applications to which these robots are applied however, these kinds of guarantees will be become increasingly important.

4.3.2 Observer Analysis

Here we discuss the theoretical limitations of the observer and their corresponding implications for problems of interest for continuum robots.

The error dynamics of the estimator in Eq. 4.24 can be thought of as a linear system perturbed by a nonlinear term, where the nonlinear perturbation depends itself on the estimation error. When this nonlinear perturbation, $\boldsymbol{\delta}(\mathbf{e},t)$, takes the form shown in Eq. 4.25, it vanishes at the origin, i.e. $\boldsymbol{\delta}(\mathbf{0},t)=\mathbf{0}$ [113]. Implicitly, this vanishing condition assumes that we have a perfect model of the friction acting on the cable such that

$$\mathbf{K_s^{-1}}(\mathbf{u}-\mathbf{f_m})-\mathbf{y_1}-\mathbf{K_s^{-1}}(\mathbf{u}-\hat{\mathbf{f}}_\mathbf{m})+\mathbf{y_1}=\mathbf{K_s^{-1}}(\mathbf{f_m}-\hat{\mathbf{f}}_\mathbf{m})=\mathbf{0}, \tag{4.26}$$

where $\mathbf{f_m}$ is the true friction acting on the cable and $\hat{\mathbf{f}}_\mathbf{m}(\mathbf{y_1},\mathbf{y_2},\mathbf{u})$ is a model of that friction that can depend on available sensor measurements and control inputs. If, however, this is not the case and $(\mathbf{f_m}-\hat{\mathbf{f}}_\mathbf{m}(\mathbf{y_1},\mathbf{y_2},\mathbf{u}))\neq 0$, the nonlinear perturbation term becomes

$$\boldsymbol{\delta}=\begin{bmatrix}\mathbf{K_s^{-1}}(\mathbf{f_m}-\hat{\mathbf{f}}_\mathbf{m}(\mathbf{y_1},\mathbf{y_2},\mathbf{u}))\\ \mathbf{0}\\ \boldsymbol{\psi}(\mathbf{z_2},\mathbf{z_3},\mathbf{y_1})-\boldsymbol{\psi}(\bar{\mathbf{z}}_\mathbf{2},\bar{\mathbf{z}}_\mathbf{3},\mathbf{y_1})\end{bmatrix}, \tag{4.27}$$

which is now potentially non-vanishing at the origin. As a result, the origin would no longer be an equilibrium point of the system [113], and the observer would no longer accurately predict the state of the robot.

There are some immediate consequences of this observation for continuum robots, particularly those with long passive segments through which the cables are routed before they reach the bending section to be actuated. This design is of practical importance because many continuum robot applications in minimally invasive surgery require that the robot be able to traverse long, winding paths through lumens in the body before accessing the surgical site [106] [22]. This winding of the robot body induces a capstan friction on the cables that depends on the total winding angle of the passive segment, $F = e^{\varphi\mu}$ [114], which is an unmeasured quantity. While it is a straightforward problem to develop an accurate friction model $\mathbf{f_m}(\mathbf{y_1}, \mathbf{y_2}, \mathbf{u})$ for the robot when it is in a single configuration, for robots with long passive segments, this model will become increasingly less accurate as the wind angle of the passive segment, φ, changes from its original value for which this friction model was derived.

Our framework suggests that there are two ways to mitigate the error in the state estimate for this special case of continuum robot when only partial state measurements are available. The first would be to adapt the friction model online based on an indirect estimate of the wind angle to better agree with the true friction experienced by the actuating cables. While this likely poses challenges of its own regarding how to distinguish external forces applied to the bending section of the robot from friction forces, it is worth further investigation. The other approach is motivated by the fact that the friction modeling error term is multiplied by the inverse of the cable stiffness matrix, Eq. 4.27. This would suggest that using stiffer cables in the robot would help to mitigate the effect of friction modeling errors on the state estimate.

4.3.3 Reference Input Control via Integral Action for Disturbance Rejection

In this section we describe how the observer detailed above can be used for control of a continuum robot. In theory, this is a straightforward task, as the estimates from the observer could be used directly in the feedback path of a control loop to regulate the configuration of the robot, as is proposed in [110]. In practice however, the velocity signal used in the output injection term (Eq. 4.21) is often corrupted by large amounts of noise which would

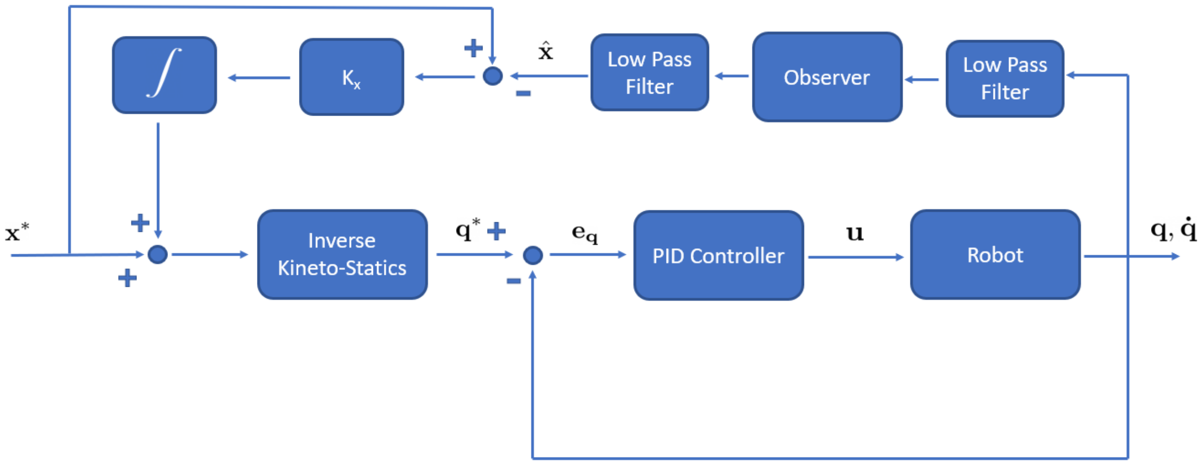


Figure 4.3. This block diagram represents the reference input control scheme that is used for control of the continuum robot and disturbance rejection. The observer is used to add integral action to a nominal, quasi-static control loop by updating the reference input such that it will regulate the robot to a desired configuration despite a disturbance applied to the robot.

be subsequently fed into the estimate of the robot configuration. If the system actuators have sufficiently high bandwidth, then the controller will respond to this noise, leading to poor, oscillatory performance. Simply low pass filtering the estimate is not guaranteed to solve the issue, as the phase lag introduced by the filtering step can be potentially destabilizing [115]. In order to avoid these problems, we instead choose to augment the standard control loop used for continuum robots in the quasi-static case with integral control action that modifies the reference input to this control loop [116] using the observer's estimate of the state (see Fig 4.3).

The standard control loop for a cable driven continuum robot computes a desired set of actuator positions to achieve a target configuration, $\mathbf{x}^*$ assuming negligible robot dynamics by first computing the necessary cable tensions as the solution to

$$\mathbf{K_b x}^* = \mathbf{J^T u}. \tag{4.28}$$

In cases where this solution is not unique, it is important to choose a solution such that $\mathbf{u} \leq \mathbf{0}$ so that all of the cables are "pulling" on the robot rather than "pushing". There are many suggestions for this in the literature [31][106][117]. We find that adding a small pre-tension to all cables is sufficient for our experiments. From there, it is possible to determine the actuator positions that will achieve these tensions, using

$$\mathbf{q}^{*} = \mathbf{K_s^{-1}}\mathbf{u} + \mathbf{J}\mathbf{x}^{*}, \tag{4.29}$$

and to reduce the error between this target and the measured actuator positions, $\mathbf{e_q} = \mathbf{q}^{*} - \mathbf{q}$, through standard PID control. If we consider a constant disturbance, $\mathbf{d}$, acting on the system, however, the force balance in Eq. 4.28 would no longer be valid, and instead becomes

$$\mathbf{K_b}\mathbf{x}^{*} + \mathbf{d} = \mathbf{J^T}\mathbf{u}. \tag{4.30}$$

To correct this, we modify the original configuration target $\mathbf{x}^{*}$ as determined above, by adding the integral of the error between this target and a low-pass filtered estimate of the robot configuration from the observer

$$\mathbf{x}^{**} = \mathbf{x}^{*} + K_x \int \mathbf{x}^{*} - \hat{\mathbf{x}}, \tag{4.31}$$

where K_x is the integral gain and $\hat{\mathbf{x}}$ is the filtered observer estimate. This reference input control scheme allows us to low pass filter the observer estimate to prevent noise from the velocity signal being injected into the control loop in a way that does not destabilize the system. The trade-off associated with this approach is reduced dynamic performance of the controller, which would not be able to reject disturbances that vary significantly in time.

4.4 Experiments

In this section we evaluate our proposed modeling, estimation, and control approaches on a hardware system.

4.4.1 Setup

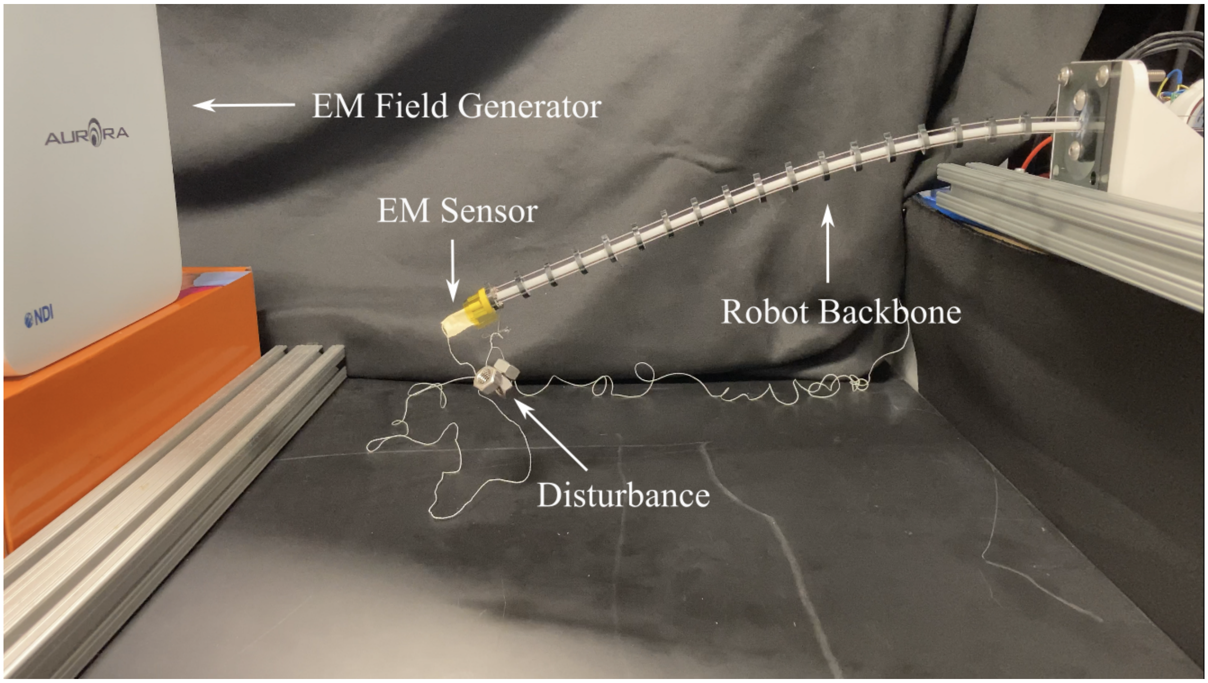


Figure 4.4. Hardware used in the experimental evaluation of our methods including an electromagnetic tracker used to generate reference measurements and a disturbance that is applied to the robot.

The test setup used in our experiments is shown in Fig. 4.4 and Fig. 4.5. It consists of a 295 mm long continuum robot actuated by three 18-8 stainless steel cables that are 0.23 mm in diameter (McMaster). The cables are spaced evenly relative to one another around the circumference of the robot cross section a distance 4.2 mm from the robot centerline. The cables are each wound around a 11 mm diameter pulley attached to a direct drive motor (Turnigy SK3) which is controlled by a motor controller (ODSEC). The motor positions are measured using an absolute encoder (AMS6500), which circumvents the need for a homing sequence based on cable pretension during robot start-up. Motor velocities are determined by one-step backward finite difference. We send current commands to the motor controllers at a rate of 150 Hz, and each motor controller runs a low level current control loop at 8 kHz.

The tip position and orientation of the robot are tracked at 30 Hz using an electromagnetic

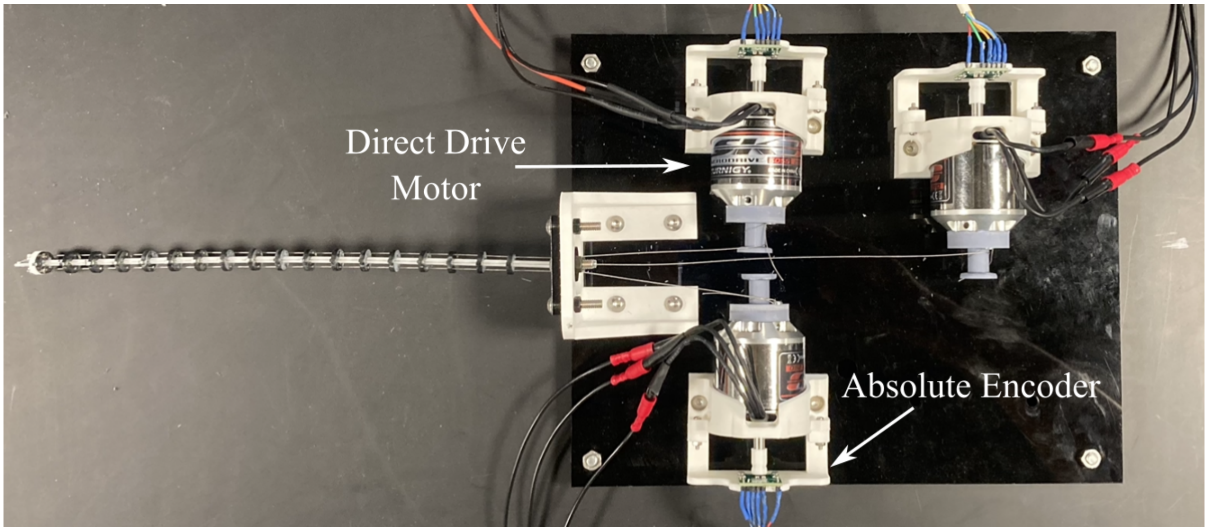


Figure 4.5. Overhead view of the robot hardware including motors, pulleys, cables, and encoders.

(EM) field generator (NDI Aurora) and sensor (NDI Aurora Micro). The reported accuracy of the sensor is 0.7 mm RMS in position and 0.3° RMS in orientation, although these values are subject to the level of ambient noise. Our algorithms for estimation and control as well as our code for data collection are all implemented in Python and run on a laptop with an Intel i7-5500U processor.

4.4.2 System Identification

Before using our proposed model, we must start by identifying its parameters, the procedure for which we detail here. We note that because our robot is negligibly compressible in the axial direction, we do not include *h* as a state variable and instead treat it as constant.

To identify the spring coefficient of the stainless steel cables, we tied one end of the cable to a force gauge and measured the stretch of the cable with a ruler for a set of applied loads. We manually stretched the cable to three different lengths and recorded the force. Then we computed the coefficient as the average ratio of force to stretched length. The lumped stiffness, $\mathbf{M}^{-1}\mathbf{K_b}$, and damping parameters, $\mathbf{M}^{-1}\mathbf{D}$, of the backbone can be determined by manually displacing and then releasing the robot backbone before routing any cables through it. We

Table 4.1. Identified parameters for the dynamic model

Parameter	Value	Description (Units)
ρ	4.2	Robot Radius (mm)
K_s	3.98	Cable Spring Constant ($\mathrm{N} \cdot \mathrm{mm}^{-1}$)
$\mathbf{K_b}$	diag(35.87, 38.16)	Robot Bending Stiffness ($\mathrm{N} \cdot \mathrm{mm}^2$)
$\mathbf{D}$	diag(781, 857)	Robot Damping ($\mathrm{kg} \cdot \mathrm{mm}^3 \cdot \mathrm{s}^{-1}$)
m	76.33	Robot Inertia ($\mathrm{kg} \cdot \mathrm{mm}^3$)
ϕ	0.0175	Actuator Inertia (kg)
b	48	Actuator Damping ($\mathrm{kg} \cdot \mathrm{s}^{-1}$)

find that the data from a single displacement is sufficient for parameter identification. We fit a constant curvature arc to the tracked tip positions over time and use the components of this fit as measured components of the robot state, $\mathbf{x} = [u, v]^T$. Using this data, it is straightforward to determine the lumped parameters following procedures that apply to any linearly damped oscillator [115]. We obtain a rough estimate of the actuator inertias by sending a step input to each of the motors and fitting lines to the measured velocities. We then average these inertia values and treat all the motor inertias as being equal, $\mathbf{\Phi} = \phi \mathbf{I}$.

In the following section, we choose the friction term to be proportional to the motor velocity, $\mathbf{f_m} = b\dot{\mathbf{q}}$. The coefficient b and the parameters of the robot mass matrix, $\mathbf{M}$, are determined by solving the following constrained optimization problem

$$\begin{aligned} &\min_{\boldsymbol{\alpha}} \sum_{t=0}^{T} ||\mathbf{x}(t) - \hat{\mathbf{x}}(t)||_2^2 + ||\dot{\mathbf{q}}(t) - \hat{\dot{\mathbf{q}}}(t)||_2^2 \\ &\text{s.t.} \quad \dot{\boldsymbol{\xi}} = \mathbf{f}_\alpha(\boldsymbol{\xi}, \boldsymbol{u}) \\ &\qquad \boldsymbol{\alpha} \in (\boldsymbol{\alpha}_{lb}, \boldsymbol{\alpha}_{ub}), \end{aligned} \tag{4.32}$$

which attempts to fit the open loop response of the robot configuration and motor velocities predicted by our dynamics model to measured values. The first constraint in this problem, enforces that the augmented state variable of our model, $\boldsymbol{\xi} = [\mathbf{q}, \dot{\mathbf{q}}, \mathbf{x}, \dot{\mathbf{x}}]^T$, evolves according to Eqs 4.13, where $\boldsymbol{\alpha}$ is a vector of all model parameters. The box constraints on the model parameters themselves, $\boldsymbol{\alpha} \in (\boldsymbol{\alpha}_{lb}, \boldsymbol{\alpha}_{ub})$, prevent the values b and $\mathbf{M}$ from being less than zero and keep the empirically determined parameters from deviating by more than 10% of their original values. We also assume that the robot inertia is the same for accelerations in either component of its curvature, $\mathbf{M} = m\mathbf{I}$, so that there are fewer parameters to estimate by optimization. There are many recommendations in the literature for the choice of data-generating input torque, $\boldsymbol{u}$, [118]. We choose to use step inputs of 9 N to each cable over a period of 4 s per step input.

4.4.3 Model Validation

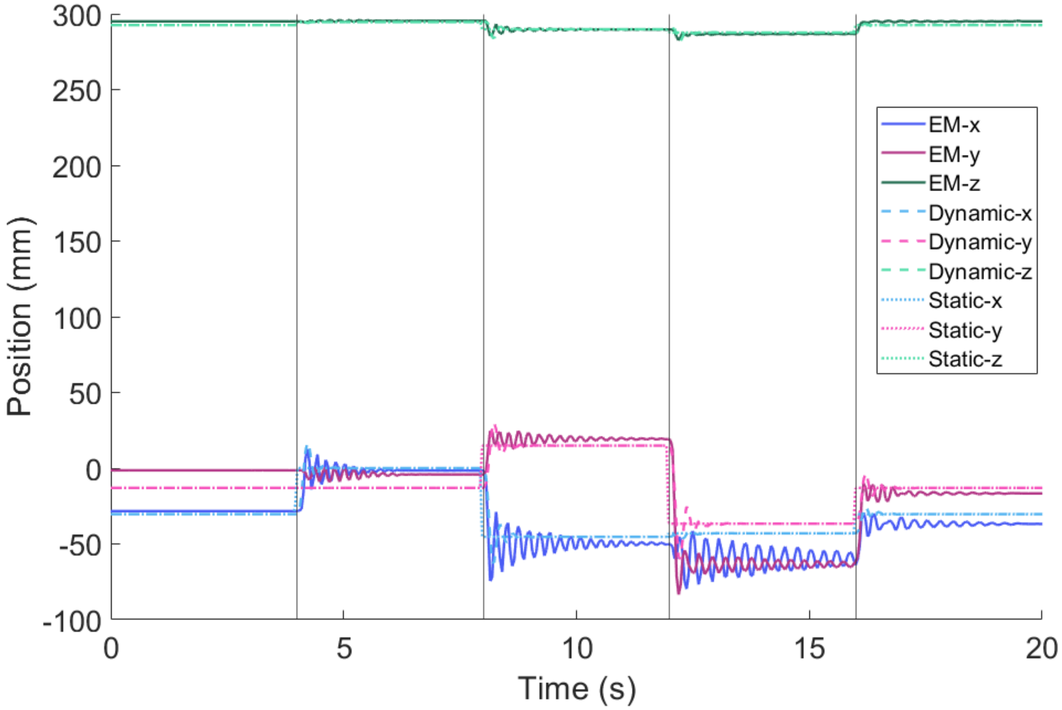


Figure 4.6. Plots showing the step responses for the continuum robot, where the robot starts at rest and each cable is then pulled on, one after the other with 6 N of force in 4 s intervals. The intervals are denoted by the black vertical lines. The plot depicts the resulting tip position measured by the EM tracker as well as the predicted tip position using our dynamic model and a kineto-static model.

We evaluate the accuracy of our model by predicting its open loop step response and comparing it with measured values from the EM tracking system. Over a period of 20 seconds, we send step inputs of 6 N to each of the three motors, one after the other, allowing sufficient

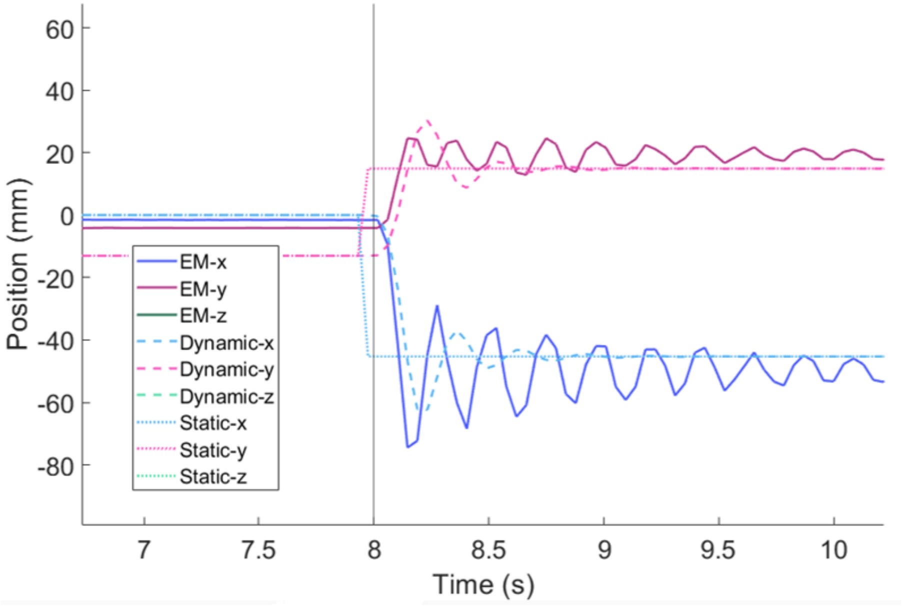


Figure 4.7. Zoomed in version of Fig 4.6, illustrating that the dynamic model captures important time-dependent behavior exhibited in the measured response, such as oscillation and lag between the input and the measured response.

dwell time between each command for the oscillations in the robot body to die out. For the first and last 4 seconds of the test, the tension command is 0 for all motors. We use the mapping described in Eq.'s 4.3 and 4.2 to project out constant curvature model value into Cartesian space so that we can compute prediction error between our model and the EM tracker directly. We also compute the prediction of a kineto-static model [31] for comparison. The results are depicted in Figures 4.6 and 4.7.

It is clear from the plots that the at steady state, the dynamic model prediction is consistent with the kineto-static one. This is expected given the assumptions of our dynamic model and the analogous system from which it is derived. Further, both model predictions do not perfectly align with the measured tip predictions. This is most likely a consequence of the constant curvature assumption, which is known to be less accurate compared to more sophisticated ODE-based methods [105], but other sources of error–such as nonlinear cable stretch or backbone disc misalignment–also contribute.

Despite this, the mean prediction error over the test is 14.70 mm for the kineto-static model and 13.98 mm for the dynamic model, which is 4.98% and 4.74% of the total robot length. Note that these errors are consistent with the literature regarding constant curvature assumption accuracy [104][105]. The small improvement in prediction error offered by the dynamic model indicates that it is able capture some of the oscillating features of the system response that the kineto-static model cannot. This is supported by the zoomed in plot, Fig 4.7, where it is clear that the prediction and response agree in the initial portion of the transient.

More significantly, it is also clear from this plot that the dynamic model captures the lag between the step input and the actual robot motion. While the kineto-static model predicts an instantaneous change in the robot response to a step input, the dynamic model more correctly shows that there is a time delay between the input tension signal and observed robot motion. More quantitatively, we see that the kineto-static model prediction leads the observed response by 0.17 s while the dynamic model prediction lags the observed response by 0.04 s. This improved phase alignment becomes increasingly important as researchers consider modeling choices for more high-performance continuum robot applications which would require commensurately more sophisticated control techniques to damp out oscillations in the robot motion.

4.4.4 Setpoint Regulation with Constant Disturbance

In this experiment, we test our compensation algorithm on a setpoint regulation task where the cable-driven continuum robot is subject to an unknown external disturbance. The particular disturbance used in our experiments is a weight hung at the tip of the robot (see Fig. 4.4), which mimics a payload that a continuum robot may have to manipulate. As is clear from the figure, the payload significantly affects the configuration of the robot.

Friction Modeling: As described in previous works [45][114], external loading also has a significant effect on how friction affects cable tension in the continuum robot body. We find that simply modeling friction as proportional to motor velocity is no longer sufficient in this case. While it is possible to analytically construct more accurate, robot-specific models [119], we choose to adopt a learning-based approach to friction modeling, similar to [45], for convenience and in order to retain generality. Specifically, we model the friction on each cable as a function of the motor positions, velocities, and commanded cable tensions using Gaussian Processes (GP), $\hat{f}_{m,i} = GP(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{u})$. This model is an appropriate choice for this task due to its very high data efficiency and well-behaved predictive behavior on out-of-distribution data [71].

We collect training data by hanging the weight from the robot and commanding it to reach various configurations around its workspace, starting from the home configuration using an uncompensated controller. Using the EM tracking system to determine the robot configuration as before, we can measure the expected cable tensions as $\mathbf{K_s}(\mathbf{q} - \mathbf{l}(\mathbf{x}))$. The friction opposing the cable tensions are then the difference between the commanded tensions and the measured tensions $\hat{\mathbf{f}}_\mathbf{m} = \mathbf{u} - \mathbf{K_s}(\mathbf{q} - \mathbf{l}(\mathbf{x}))$. In total we collect 85,500 samples which we downsample by a factor of 50 in order to make training tractable. We use an isotropic squared exponential kernel [71] in our GP model which has only two hyperparameters that can be optimized for in about 5 seconds on a standard laptop computer. The rest of the parameters used in our dynamics model remain the same as in Section 4.4.3.

Experimental Results: We evaluate our disturbance compensation strategy on a setpoint regulation task. We compare control with and without compensation used to drive the robot from its starting configuration to one of 6 target configurations while there is an unknown weight hanging from the tip of the robot. To prevent a large, sudden motion that would dislodge the weight and damage the robot, the desired configuration of the robot is interpolated from the straight configuration to the target configuration over a period of 10 s using a standard fifth-order position-velocity-time polynomial [120]. The targets are chosen by actuating the robot to cover a large region of its workspace without the weight and recording the tip position at 6 locations, so it is known that a well-designed controller could theoretically reduce the error to zero (Fig. 4.8).

The final position errors for each regulation task are recorded in Table 4.2. Fig 4.9 shows examples of the difference in robot configurations using each control method for one of the

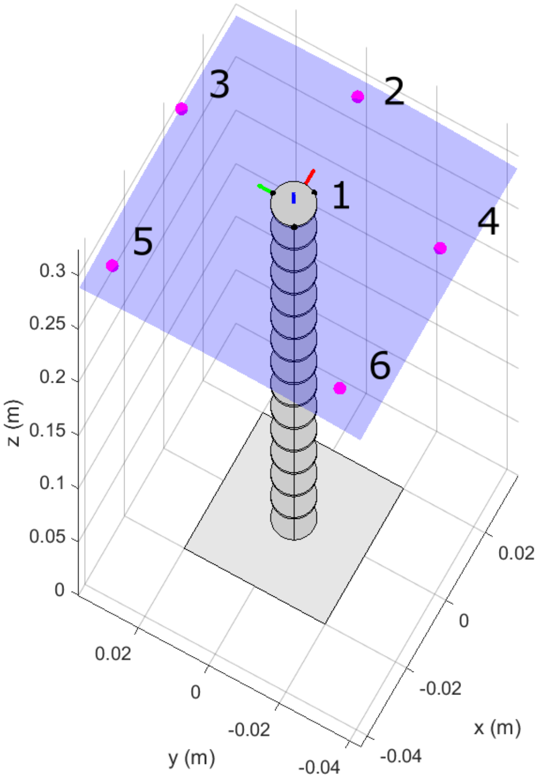


Figure 4.8. The robot was required to reach six different target tip positions, shown here as pink dots. The plotted surface shows feasible tip positions the robot could reach. The image of the continuum robot is generated with OpenCRProject software (https://www.cs.toronto.edu/ jbk/opencontinuumrobotics/software/)

regulation tasks. It is clear that the compensation-based method reduces the positioning error relative to the uncompensated method in all trials (by 20.5 mm on average) but does not achieve perfect tracking–the minimum final error for the compensated approach is 4.94 mm and the maximum is 26.49 mm. There are many possible causes for this imperfect tracking, that can be better understood through Fig 4.10, which compares the observer-based predictions of the robot configuration with the target and measured configuration values. From this graphic, it is clear that the observer-based estimate of the configuration converges to the desired configuration over time, but that this estimated value does not match the measured value. This indicates that the

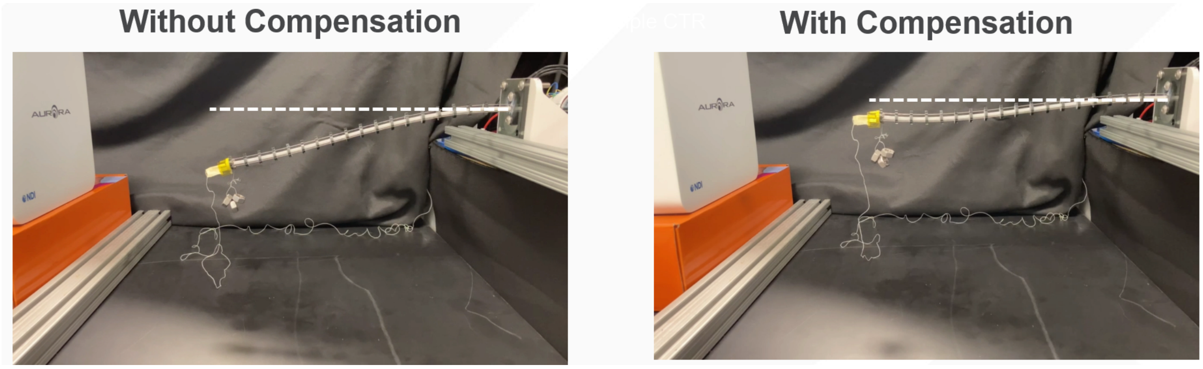


Figure 4.9. Image of the continuum robot when trying to reach a straight configuration both without (left) and with (right) compensation from feedback of the observer estimate of the robot state. It is clear that the compensated approach significantly reduces the positioning error.

observer is the source of the error, and not the controller. Potential reasons for the observer inaccuracy are discussed in Section 4.3.2. Despite these limitations, these results show that it is possible to significantly reduce the effect of a constant disturbance applied to a continuum robot without requiring additional sensors to be integrated into the body of the robot.

Table 4.2. Final tip position error for the compensated and uncompensated approaches for each setpoint

	Target 1	Target 2	Target 3	Target 4	Target 5	Target 6
Error without Compensation (mm)	40.77	45.51	26.31	65.56	7.71	25.58
Error with Compensation (mm)	9.08	26.49	11.84	16.47	4.94	19.64

4.5 Discussion

In this work, we have shown how a simple model for continuum robot dynamics can be derived using an analogous mechanical system that first gained popularity for quasi-static modeling. Given the similarity of these model equations to other more classical robotic systems, this model has the potential to enable the use of established tools in analysis, estimation, and control for continuum robots. As an example of this, we demonstrate how our model can be used to construct a reduced order observer for the continuum robot that estimates the state of the

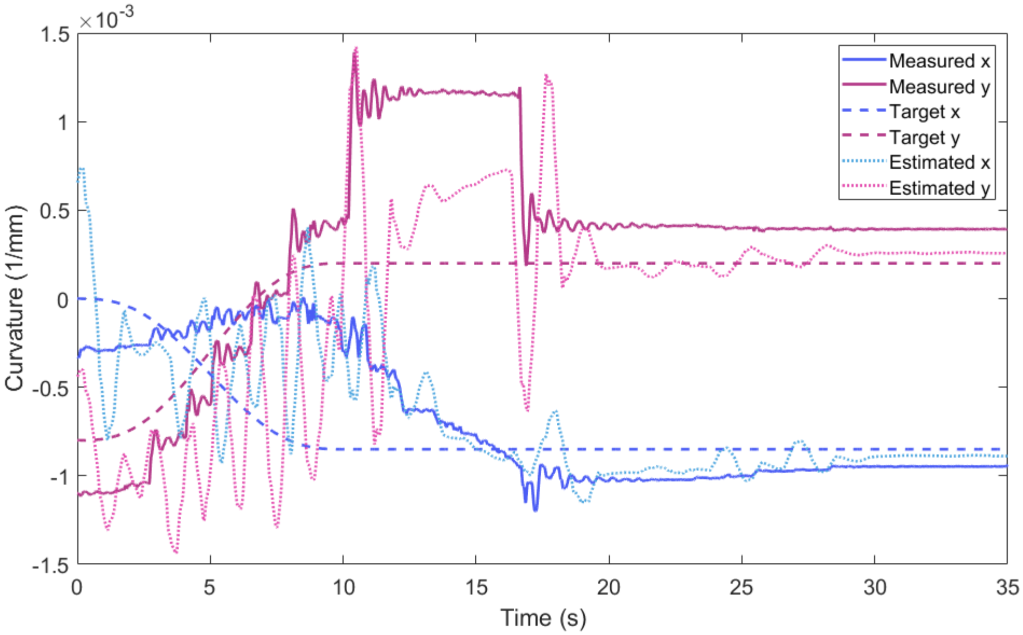


Figure 4.10. The target, measured, and estimated configuration of the robot over time for one of the setpoint regulation tasks. The results show that while the error between the estimate and the target is reduced nearly to zero, error between the estimated and measured configurations remains.

robot using only position and velocity measurements from the actuators. We show conditions under which the observer has provable convergence – which is a rare result for a continuum robot – as well as analyze its limitations and their implications for certain cable-driven robot designs. We develop a control strategy that uses the observer to reject disturbances applied to the robot body, even when the robot's actuator velocity measurements are corrupted by significant amounts of noise. Finally, we validated both our proposed modeling approach and observer-based control strategy on a hardware system which verify the real world applicability of the theoretical contributions in this work.

This work also suggests some immediate and exciting directions for future research. First, the observer developed here for state estimation from partial feedback can be adapted to include additional sensors incorporated on the robot. This would provide a principled means of sensor fusion for various measurements, such as cable tensions, robot strain measurements, robot position measurements, and motor position and velocity measurements. Additionally, adaptive control methods could be explored to compensate for situations in which the cable friction model is expected to change. Finally, more high-performance control ideas can be explored for non-medical continuum robot applications by using the dynamic model we have developed to damp out oscillations in the robot body that occur after sudden changes in its position. In conclusion, we believe that our work is an important step towards unlocking greater potential from these robots by moving away from their statics and towards dynamics.

4.6 Acknowledgements

This chapter contains material that will be submitted for publication under the title ”*Disturbance Rejection for Cable-Driven Continuum Robots without Integrated Sensing*.” in 2023. The dissertation author is the primary author of this work.

Chapter 5

Image Segmentation for Continuum Robots from a Kinematic Prior

5.1 Introduction

In the previous chapters of this dissertation, we have focused on algorithms for continuum robot control that rely on very limited sensor information—the robot's joint data and possibly its tip pose. Using such sparse amounts of data makes the problem of state estimation, and ultimately control, quite challenging. One possible approach to alleviate some of this difficulty, without requiring that more sensors be integrated into the robot, would be to use images from an external camera to estimate important aspects of the robot's state. Examples of such work that uses vision, include shape estimation [26, 27, 28, 29], the identification of inertial parameters used in dynamics models [17, 121], and the estimation of forces the robot applies to unknown surroundings [122].

Importantly, a common feature of these works on vision-based estimation is the requirement to have an accurate segmentation of the robot, that distinguishes the pixels belonging to the continuum robot body from the rest of the image. In order to obtain this segmentation reliably however, the images used in these previous works are taken under ideal conditions, which do not exist outside the laboratory, severely limiting their real-world applicability [123, 124]. To address this issue, we propose a new method for accurate continuum robot segmentation (Fig. 5.1) that is robust to more realistic imaging conditions and can be used to help further the growing body of research on vision-based estimation for continuum robots.

5.1.1 Related Works

In the last decade, most image segmentation research has been dominated by deep learning-based methods—largely spurred by the success of ImageNet [125]. It has become clear that, given sufficient computational power and training data, the results achieved by deep networks significantly outperform all other existing methods, especially on general image segmentation tasks.

However, amassing the requisite amount of labeled training data for the very specialized task of segmenting a continuum robot is a challenging undertaking. Even transfer learning approaches—where a successful deep network architecture that has first been trained on a very general set of labels is then retrained on a more specific set of labels using fewer images—require on the order of $\sim 10e4$ images. An individual collecting and labeling such a large amount of data is a highly impractical undertaking, especially given the large variety of continuum robot designs for which the process would need to be repeated. Online datasets, which have catalyzed advancements in other sub-disciplines of robotics, such as self-driving vehicles [126] and manipulation [127], do not yet exist for continuum robots. Finally, perhaps the most significant impediment to gathering useful training data for continuum robots is that many of the intended applications are surgical, which adds regulatory and privacy concerns for how the

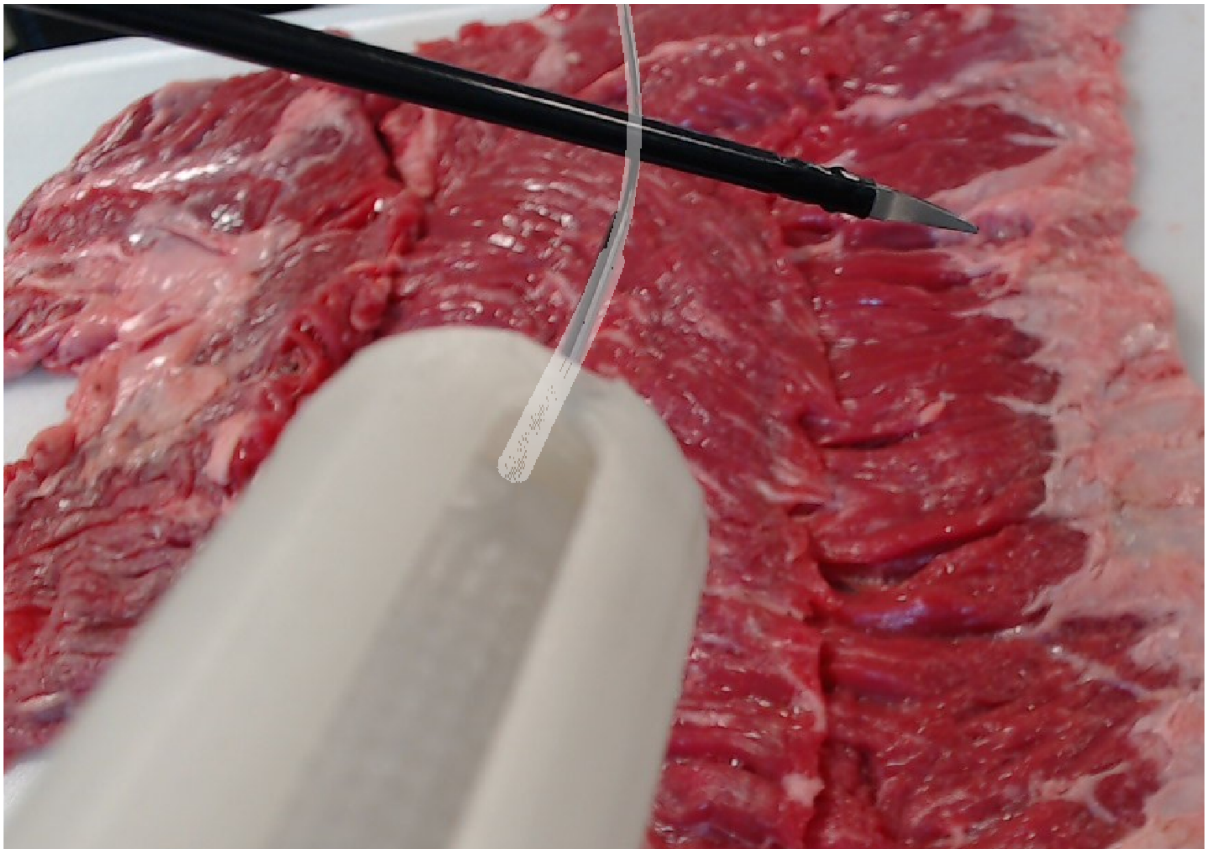


Figure 5.1. An example of robust image segmentation for a concentric tube continuum robot. The robot is successfully identified, as illustrated by the white overlay, despite the presence of multiple, partial occlusions and a background of similar color.

data is collected [128].

As an alternative approach to using deep networks for vision-based estimation for continuum robots, researchers often rely on composing an image scene to contain the entire robot body against a uniform, high-contrast background [26, 27, 29, 17, 121, 122]. This control over the composition of the scene, enables straightforward, background-subtraction techniques based on color or another photometric quantity to effectively segment the robot without the need for large amounts of training data. These simple strategies are not robust, however, in that as the complexity of the scene increases (e.g. the robot is only partially visible, foreign objects are present, the background color is non-uniform and similar to the robot body, etc.) the segmentation becomes less reliable [123, 124]. Although it is possible to improve color-based segmentation methods through proper coloring of the instrument or the addition of reflective markers, such modifications suffer from sterilization and integration challenges [129] [130].

In the decade preceding the explosion of deep learning approaches, researchers studied segmentation methods based on models of the 3D shape of a target object. By leveraging this prior knowledge, these methods [131, 132] can effectively segment an object in a more complicated scene where a simple color-based method would fail. In this work, we propose to augment these methods with a kinematic model for a continuum robot that can be used for segmentation, without the need for a large set of training data.

5.1.2 Contributions

Specifically, the contribution of this work is an algorithm for segmentation of continuum robots that: requires minimal training data, is general across continuum robot designs, and segments the robot in realistic image conditions that include partial views and challenging background conditions (where color-based segmentation of the robot alone performs poorly). In the following section we detail the components of our algorithm that leverage information about the robot's shape derived from a kinematic model to construct an optimization problem which can be solved to segment the robot. Next, we test our algorithm empirically using images of the robot collected from a simulated surgical environment and demonstrate its improved performance relative to current techniques. Finally, we conclude with a discussion of the current limitations of the work, along with future directions.

5.2 Methods

Our algorithm for continuum robot segmentation consists of three main parts (Fig. 5.2). First, the image to be processed undergoes an initial background subtraction. Second, joint measurements of the robot's configuration are fed into a kinematic model to produce an initial parameterization of the shape of the robot. Finally, this parameterization is refined through optimization to minimize an energy defined for the background-subtracted image, which produces a segmentation for the robot. An example of this workflow is shown in Fig. 5.4. The nomenclature used throughout this article is summarized in Table 5.1.

5.2.1 Background Subtraction

This first step is not expected to achieve a perfect segmentation—only to remove elements of the image that are grossly inconsistent with the expected appearance of the robot. Consequently, our algorithm *does not* require a specific background subtraction method. Instead, this choice is left to the user as a design decision, which may depend on the information available for the particular problem.

Consider an initial image I, a binary mask $M(I, \boldsymbol{\theta})$, and background-subtracted image $I' = I \circ M$, where $\circ$ is the Hadamard product. The mask, M, which delineates the foreground from the background, is a function of the image, I, and a vector of parameters, $\boldsymbol{\theta}$, used to encode any information about the background available to the user. For instance, if the user has access to images of the background *a priori*, $\boldsymbol{\theta}$ may be the weights of a Gaussian Mixture Model [133], neural network [134], or other learned function [135], meant to capture statistics about the background pixels. If, however, the user does not have data to train such a model, $\boldsymbol{\theta}$ may

Table 5.1. Nomenclature

$\hat{(\cdot)}$	conversion of an element of $\mathbb{R}^3$ to an element of $\mathfrak{so}(3)$
s	arclength variable of the continuum robot
L	total arclength of the robot
$\mathbf{u}$	curvature of the robot centerline
$\mathbf{p}$	position vector of the robot centerline
$\mathbf{R}$	orientation matrix of the robot centerline
$\mathbf{e}_3$	the third element of the standard basis
$\mathbf{q}$	vector of the robot joint positions
$f(\cdot)$	mapping from robot joint positions to centerline curvature
$\Gamma(\cdot)$	mapping from robot centerline pose to 3D shape
S	3D shape of the robot
$\mathbf{x}_0$	a 3D point on the surface of the robot in the robot frame
$\mathbf{\Omega}$	rotation between camera frame and robot frame
$\mathbf{t}$	translation between camera frame and robot frame
$\mathbf{x}$	a 3D point on the surface of the robot in the camera frame
$\pi(\cdot)$	a perspective projection mapping
$\mathbf{y}$	a 2D pixel coordinate in the domain of the image
I	raw image
M	binary mask of background
I'	background subtracted image
Φ	domain of image
B	subset of image domain belonging to background
R	subset of image domain belonging to projection of shape model
R^c	complement of R
$I(\mathbf{y})$	color value of the image of a pixel coordinate $\mathbf{y}$

be the parameters used to specify a simple color mask that removes unlikely robot colors. In our experiments, we adopt this later approach, but incorporating more complicated models is straightforward.

5.2.2 Continuum Robot Shape Model

The key element of our proposed approach is to leverage a model of the expected shape of the continuum robot to improve segmentation. Specifically, we consider a mapping

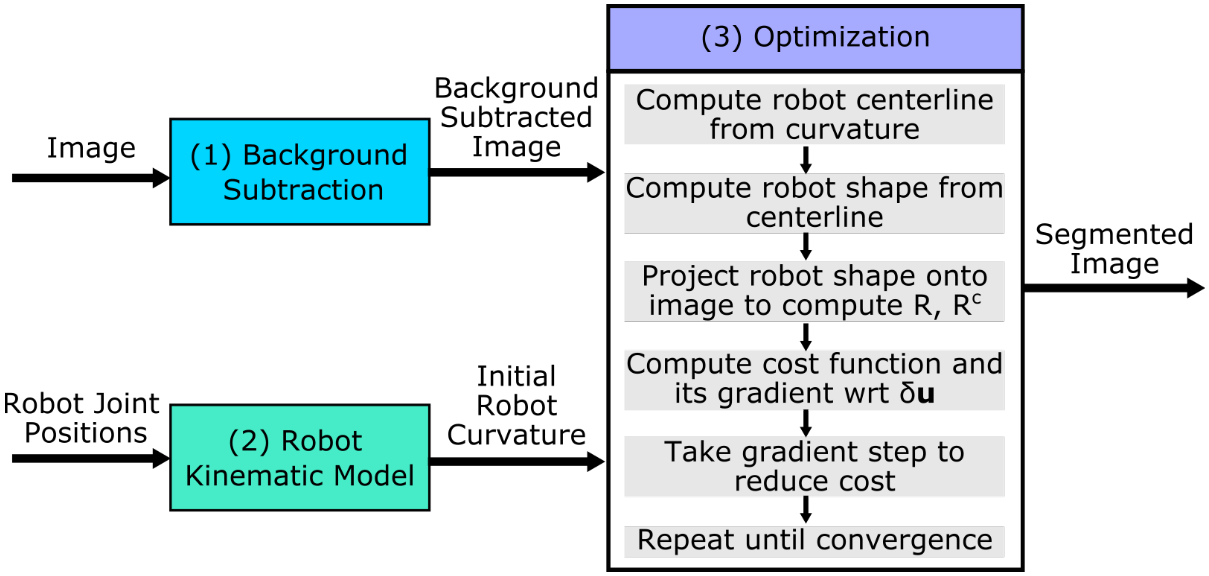


Image
(1) Background Subtraction
Background Subtracted Image
(3) Optimization
Compute robot centerline from curvature
Compute robot shape from centerline
Project robot shape onto image to compute R, R^c
Compute cost function and its gradient wrt δ**u**
Take gradient step to reduce cost
Repeat until convergence
Segmented Image
Robot Joint Positions
(2) Robot Kinematic Model
Initial Robot Curvature

Figure 5.2. Workflow diagram of the proposed algorithm.

$\mathbf{u}(s) = f(\mathbf{q})$ from the robot's joint configuration, $\mathbf{q} \in \mathbb{R}^m$, to the *curvature* of its centerline, $\mathbf{u}(s)$, which is parameterized by the arclength of the centerline, $s \in [0, L]$. There is extensive literature on forward kinematic models specific for different continuum robot designs that define such mappings (for a review see [16] and [136]). These models range in complexity from simple constant curvature models, which do not vary with respect to s, to more complicated mechanics-based models that define $f(\cdot)$ as the solution to a set of differential equations.

Given such a mapping, it is then straightforward to determine the position, $\mathbf{p}(s)$, and orientation, $\mathbf{R}(s)$, of the robot centerline as a function of arclength by integrating the system of equations,

$$\begin{aligned} \dot{\mathbf{p}} &= \mathbf{R}\mathbf{e}_3 \\ \dot{\mathbf{R}} &= \mathbf{R}\hat{\mathbf{u}}, \end{aligned} \tag{5.1}$$

with an appropriate set of initial conditions, $\mathbf{p}(0), \mathbf{R}(0)$, where $\hat{(\cdot)}$ maps the robot curvature to $\mathfrak{so}(3)$. It is important to note that the 1D centerline of the robot is distinct from its expected *shape*, which is a surface embedded in 3D space. Therefore it is necessary to define another mapping, $S = \Gamma(\mathbf{p}, \mathbf{R}, s)$, which maps information about the robot centerline computed in Eq. 5.1 to the shape of the robot, S, effectively "drawing" the robot shape from its centerline. The exact definition of $\Gamma(\cdot)$ depends on the specific design of the robot to be segmented and is straightforward to determine given that continuum robot designs to date have cross sections that are radially symmetric about their centerline.

5.2.3 Segmentation by Shape Optimization

Consider an image I with domain Φ and area element $d\Phi$. Our approach to segmentation is inspired by methods such as [131][137], which pose an optimization problem to minimize energies of the form

$$E = \int_R E_{in}(I)d\Phi + \int_{R^c} E_{out}(I)d\Phi, \tag{5.2}$$

where E_{in} and E_{out} are functions that are integrated to capture statistics about pixels within a region of the image, $R \subset \Phi$, and within its complement, $R^c = \Phi \backslash R$, respectively. The region R is determined from a model of the surface of the object to be segmented by first transforming the object into the reference frame of the camera and then projecting the object onto the image domain, Φ, using a projection model. Specifically, consider a model of an object with a shape defined by surface S. Every point on the surface, $\mathbf{x}_0 \in S$, is transformed into the camera frame by

$$\mathbf{x} = \mathbf{\Omega x_0} + \mathbf{t}, \tag{5.3}$$

where Ω and $\mathbf{t}$ are the orientation and translation of the camera frame relative to the coordinate frame in which S is defined. This transformation is followed by a projection,

$$\mathbf{y} = \pi(\mathbf{x}), \tag{5.4}$$

which maps 3D points in the reference frame of the camera, $\mathbf{x} = [x,y,z]^T$, to 2D pixel locations in the image domain, $\mathbf{y}$. The region R is then defined to be the set of all points $\mathbf{y}$ in the image domain (Fig. 5.3). The choice of $\pi(\cdot)$ is another design decision afforded to the user. Here, we assume an ideal perspective projection model so that $\mathbf{y} = [\frac{x}{z}, \frac{y}{z}]^T$.

Prior works typically assume that the pose of the target object in the camera frame, $\{\blacksquare, \mathbf{t}\}$, is unknown and solve for these variables by minimizing Eq. 5.2. In our case, however, the target object is a continuum robot for which the relative pose of the camera is usually known. In many robotic applications, the camera is either mounted in a fixed position relative to the robot [138] or is under direct robotic control—as is the case for many surgical robots [139]. Consequently, we instead choose to segment a continuum robot from an image by minimizing an energy functional like Eq. 5.2 over a parameterization of the robot shape.

As described in Section 5.2.2, a model of the shape of a continuum robot is completely defined by its curvature, $\mathbf{u}$, which can be determined from the robot's joint configuration, $\mathbf{q}$. We pose a modified optimization problem to find a new robot curvature, $\mathbf{u}' = \mathbf{u} + \delta\mathbf{u}$, that solves

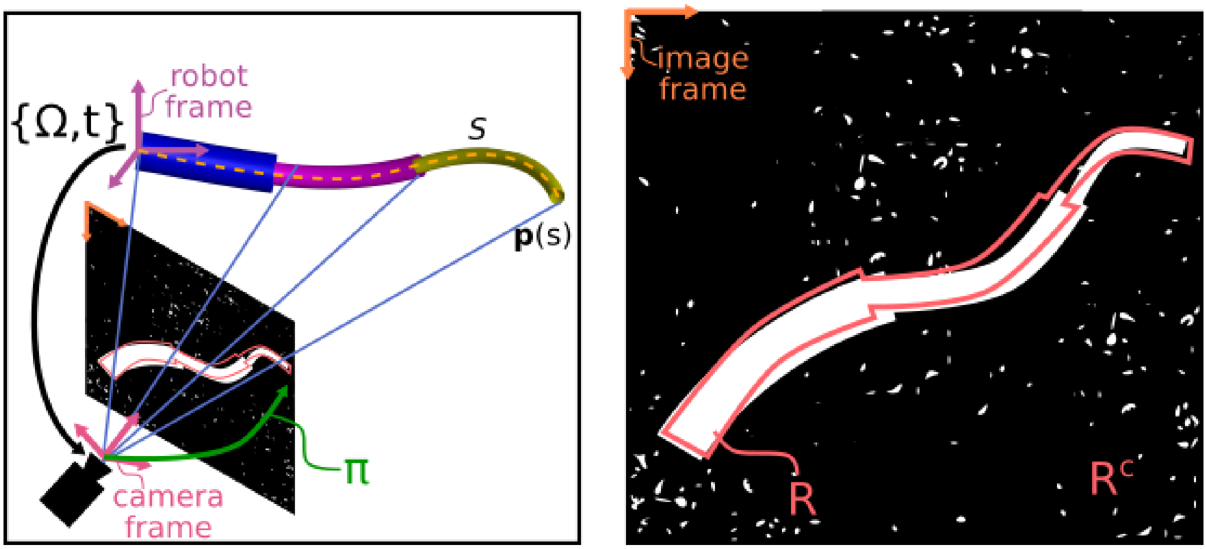


Figure 5.3. A schematic of a continuum robot shape model and its projection onto an image. The centerline of the model, $\mathbf{p}(s)$, is determined from the robot's joint configuration and a forward kinematic model. Points on the surface of the robot, S, are transformed from the robot reference frame to the camera reference frame, given the relative pose of the camera as defined by $\{\blacksquare, \mathbf{t}\}$, and are then projected into the image frame with a mapping defined by $\pi(\cdot)$.

the following:

$$\min_{\delta\mathbf{u}} \gamma_1 \int_{R-B} E_{in}(I'(\mathbf{y}))d\Phi + \gamma_2 \int_{R^c-B} E_{out}(I'(\mathbf{y}))d\Phi \\ + \underbrace{\gamma_3 \int_L ||\delta\mathbf{u}||^2 ds}_{E_{reg}} + \underbrace{\gamma_4 \int_{R\cap B} d\Phi}_{E_{pen}}. \quad (5.5)$$

Here, E_{in} and E_{out} are the same as in Eq. 5.2, but we use a background subtracted version of the image (Section 5.2.1), and the domains of integration are restricted to not include any pixels in the domain of the background, $B \subset \Phi$. Compared to Eq. 5.2, we also add two new terms to our modified objective function. The first term, E_{reg}, is a regularization term that penalizes large changes to the curvature of the robot and is integrated over the arclength of the robot centerline. The second additional term, E_{pen}, is the integral over all pixels that are common to both the background and the projection of the robot shape onto the image. This term penalizes solutions where large regions of the robot projection lie in the domain of the background. The weights, γ_{1-4}, are used to adjust the relative importance of each of the terms. We find that setting the weights such that all terms in the cost function have the same order of magnitude tends to produce effective results. There are many possible choices for the functions E_{in} and E_{out} [131]. Here we adopt the method from [137]:

$$\begin{aligned} E_{in,out}(I(\mathbf{y})) &= (I(\mathbf{y}) - \boldsymbol{\mu}_{in,out})^2, \\ \boldsymbol{\mu}_{in,out} &= \frac{\int_{R,R^c} I(\mathbf{y})d\Phi}{\int_{R,R^c} d\Phi}, \end{aligned} \quad (5.6)$$

where $I(\mathbf{y}) \in \mathbb{R}^3$ is the color at pixel location $\mathbf{y}$, and $\boldsymbol{\mu}_{in,out}$ is the average color over the region R, R^c.

5.3 Experimental Evaluation

In this section, we empirically evaluate the performance of our algorithm under different conditions. We do so for the segmentation of a concentric tube robot (CTR), which is a specific class of continuum robot. We outline the details of the experiment, discuss metrics for characterizing the performance, and finally analyze the results.

5.3.1 Experimental Setup

CTRs consist of precurved, flexible, telescoping tubes which can be translated and rotated relative to one another by an actuation unit. As the tubes are actuated, they interact in bending and torsion to reach mechanical equilibrium and produce different robot configurations [14, 95]. The specific CTR used in this experiment has the same tubes and actuation unit as described in [97]. The kinematic model we use to determine the robot shape from its joint angles is based on the mechanics of Cosserat rods and is detailed in [18]. To draw the shape of the robot from its

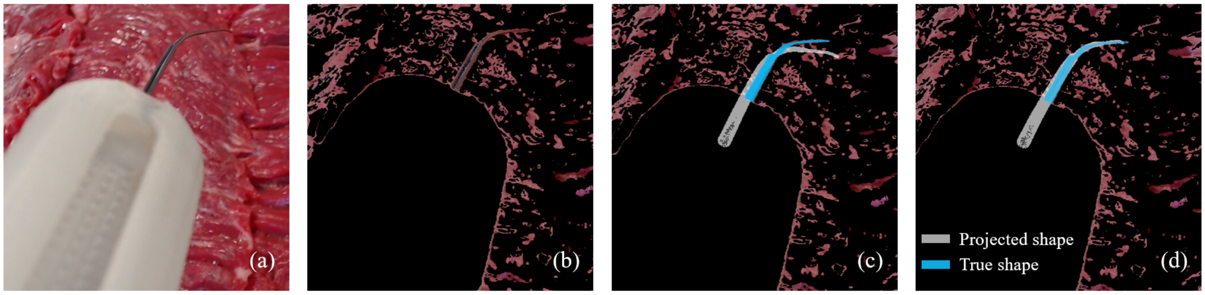


Figure 5.4. (a) Example of a raw, color image prior to segmentation by our proposed algorithm. (b) The image after background subtraction has removed colors inconsistent with the robot body. (c) Projection of the initial robot shape (white) onto the background subtracted image clearly deviates from the true robot shape (blue). (d) Projection of the optimized robot shape (white) onto the background subtracted image. The optimized shape aligns much better with the true robot shape (blue).

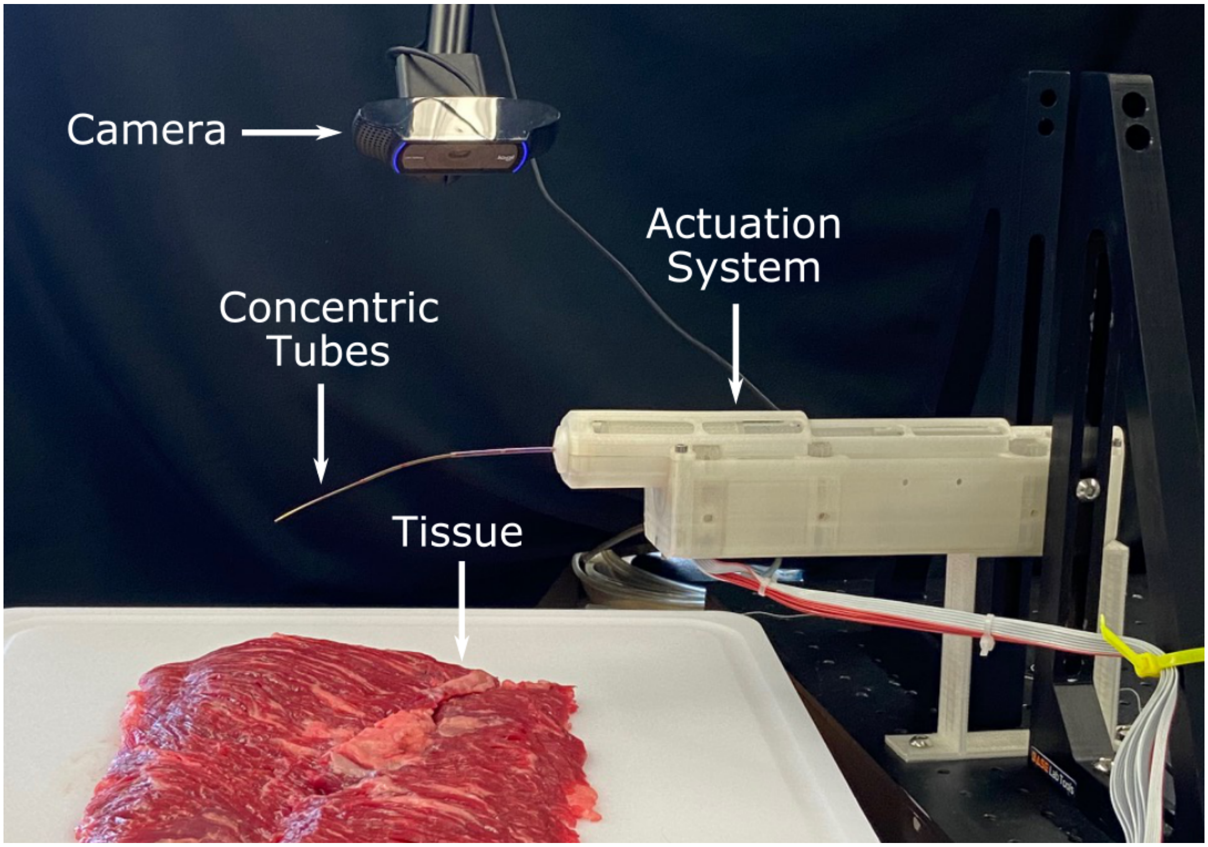


Figure 5.5. The experimental setup consists of a concentric tube robot mounted above a piece of tissue. An external camera is attached to an articulating locking arm to enable images to be taken from both an "orthogonal" and an "endoscopic" viewpoint.

centerline, we define the function

$$S(s,\phi) = \Gamma(\mathbf{p},\mathbf{R},s,\phi) = \mathbf{p}(s) + \mathbf{R}(s)\mathbf{R}_z(\phi)\begin{pmatrix}\rho(s)\\0\\0\end{pmatrix} \tag{5.7}$$

where $\mathbf{R}(s)\mathbf{R}_z(\phi)$ is a rotation of ϕ radians about the tangent to the robot centerline and $\rho(s)$ is the diameter of the outermost robot tube at arclength location s.

To evaluate our proposed method, we gather 4 sets of 20 images of this robot from an external camera (Logitech C920) as the tip of the CTR follows a trajectory that spans a large region of its workspace (Fig. 5.5). At the time when each image is taken, the robot joint values are also recorded for use in the algorithm. Half of the image data collected (i.e. 40 images) comes from an “orthogonal” viewpoint—where the camera is oriented orthogonal to the insertion axis—while the other half comes from an “endoscopic” viewpoint—where the camera is oriented nearly parallel to the insertion axis of the robot. The first viewpoint is more common in lab settings while the second, more challenging viewpoint is similar to how a camera might be mounted for use with a minimally invasive surgical robot. The endoscopic viewpoint is more challenging since differences in the robot’s configuration are difficult to discern and because the robot is often partially occluded by its own body and/or its actuation system.

From each viewpoint, we collect 2 sets of data, one in which the robot is partially occluded by a laparoscopic instrument (Endopath monopolar cautery scissors, Ethicon) and one in which it is not. The camera pose relative to the robot is determined from markers affixed to the robot at known locations. All images are taken against a background of animal tissue which is similar in color to the robot tubes and meant to simulate a surgical environment (Fig. 5.5). After collection, the images are hand-segmented to produce ground truth labels that can be used to evaluate the performance of the algorithm. We use the Intersection Over Union (IoU) of the algorithm segmentation against the ground truth as the performance metric, which is computed as:

$$IoU = \frac{|A \cap B|}{|A \cup B|}. \tag{5.8}$$

Higher IoUs indicate better performance and an IoU of 1 indicates a perfect match between the algorithm segmentation and ground truth. We also report IoUs for the segmentation produced by the unoptimized projection of the robot (i.e. $\delta \mathbf{u} = 0$) and for the background subtraction for comparison with our algorithm. The color mask used for background subtraction (Section 5.2.1) and the weights in the objective function used for optimization are the same for all images.

Table 5.2. Comparison of IoUs for the different segmentation methods across various image sets. Adjusted IoU scores are reported to the right of the | (see Section 5.3.2 for details)

	Orthogonal	Orthogonal Occluded	Endoscopic	Endoscopic Occluded
Background Subtraction	0.016 ± 0.001	0.015 ± 0.001	0.022 ± 0.003	0.013 ± 0.003
Unoptimized Projection	0.284 ± 0.087	0.292 ± 0.101	0.284 ± 0.092 | 0.415 ± 0.143	0.254 ± 0.088 | 0.427 ± 0.157
Optimized Projection	**0.665** ± **0.063**	**0.603** ± **0.134**	0.414 ± 0.104 | **0.636** ± **0.153**	0.3 ± 0.109 | **0.52** ± **0.171**

5.3.2 Results

We record the mean IoU plus or minus one standard deviation for each method on each data set in Table 5.2. From these numbers, it is clear that background subtraction by color thresholding alone is a poor way to segment the robot when it is similar in appearance to the background (top row of Table 5.3), regardless of the viewing angle, which is consistent with previous works [123, 124]. This result motivates the need for a more advanced approach for segmentation, even when the entire robot body is visible in the image.

Projecting the unoptimized shape of the robot gives better results than background subtraction alone but still fails to accurately capture all of the pixels of the robot (middle row of Table 5.3). Although we use a sophisticated mechanical model to predict the shape of the robot, there are many sources of error that contribute to the misalignment between the projection of the robot shape model and the actual image of the robot. Examples of possible sources of error include, but are not limited to, the effects of unmodeled phenomena like friction, backlash, and fabrication tolerances [100]; uncertainty in model parameters such as bending stiffness or tube curvatures; and small calibration errors in camera pose or joint offsets.

Our proposed algorithm, however, results in significantly higher IoUs across all test conditions. This improved result is due to the fact that our method uses data from the image to refine the model of the robot's shape, consequently producing a projection that more closely agrees with the ground truth labels (bottom row of Table 5.3). Note that for the endoscopic viewpoint datasets, we report two differnt IoUs: a raw IoU, calculated as previously described, and an adjusted IoU, which excludes segmentation pixels that are in occluded regions. The rationale for this adjusted IoU is that the ground truth labels only capture regions of the image where the robot is visible. The projection of the robot shape model, however—whether optimized or not—does not account for any occlusions in the environment. This feature is beneficial in that it provides the user with a segmentation of the entire robot—even portions that are not visible—but makes the IoUs of the optimized and unoptimized projections more difficult to distinguish numerically because a large percentage of both projections is occluded by the actuation unit of the robot. By removing the pixels projected into the occluded regions from the IoU calculation, the improvement due to optimization of the robot shape model becomes more apparent.

5.4 Discussion

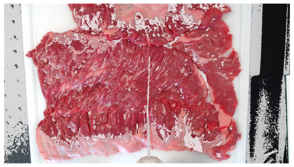
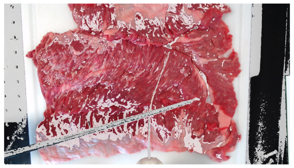
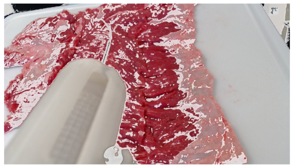
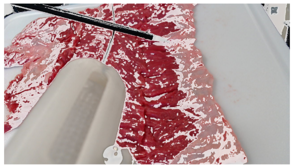
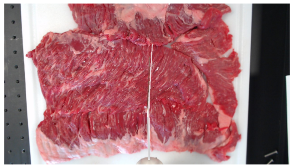
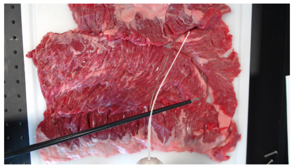
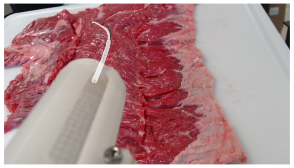
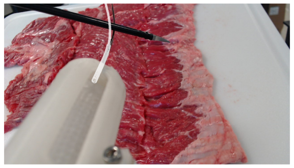
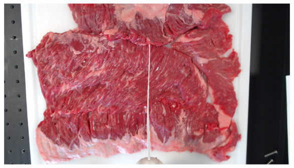
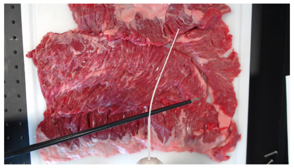
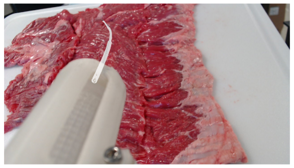
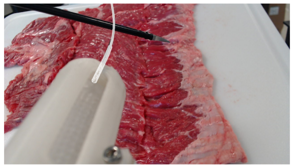
In conclusion, we have presented a new algorithm for robust continuum robot segmentation that does not require labeled training data. Our approach poses an optimization over changes to a parameterization of the shape of the robot. It produces a segmentation of the robot even for challenging cases, such as when the background is similar in color to the robot or the robot is only partially visible. We validated our proposed approach empirically and demonstrated that our method shows significant, quantifiable improvement over color-based segmentation approaches, and is robust to errors in the model used to describe the shape of the robot and its projection.

There are several possible extensions of this method and directions for future work. First, our current method assumes that the robot operates in free space, unaffected by external forces arising from, for example, tissue interaction that may be present in a real-world scenario. To account for this, our approach can be adapted to utilize more sophisticated kinematic models that account for the effect of external loads on the robot shape [122]. Second, the computational efficiency of the algorithm can be improved by investigating lower-dimensional parameterizations of the robot shape, warm-starting the optimization, and exploiting parallel computations that could enable use in real-time applications. Finally, this method could be combined with learning-based segmentation techniques for improved performance, similar to the automated labeling method described in [140].

5.5 Acknowledgements

This chapter is, in part, a reprint of material from C. Watson, A. Nguyen, and T.K. Morimoto. “*Image Segmentation for Continuum Robots from a Kinematic Prior*.” IEEE/RAS Int. Conf. on Robotics and Automation (ICRA), 2023. The dissertation author is the primary author of this work.

Table 5.3. Different image segmentation methods applied to each of the four datasets described in Section 5.3.1. The resulting segmentation is shown as a transparent, white overlay on top of each of the original images.

	Orthogonal	Orthogonal Occluded	Endoscopic	Endoscopic Occluded
Background Subtraction				
Unoptimized Projection				
Optimized Projection				

Chapter 6

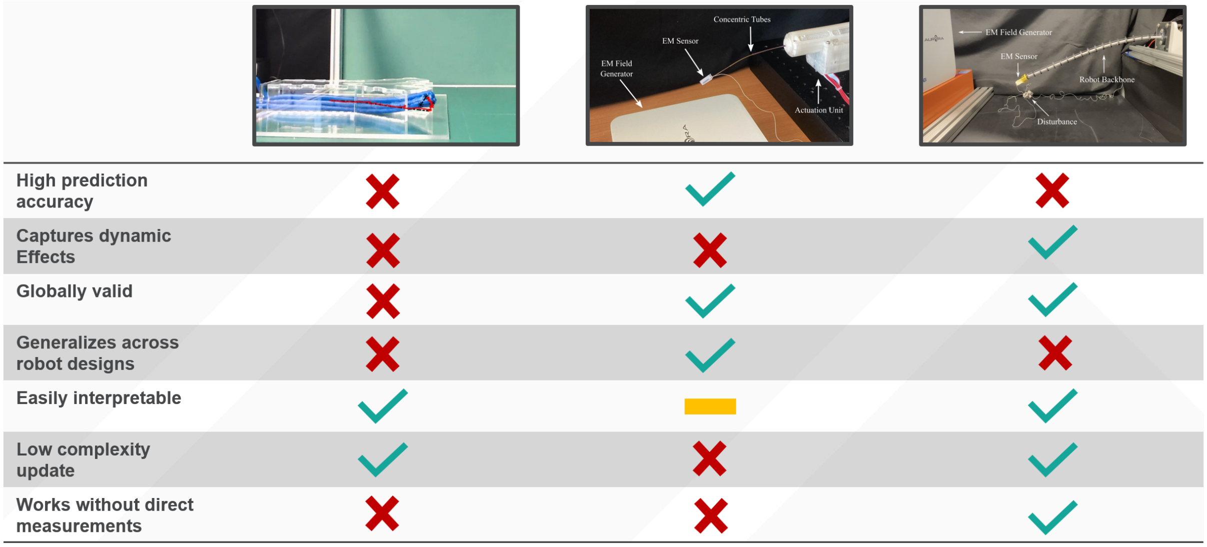
Conclusions

This dissertation has presented several different contributions to the problems of modeling, estimation, and control for continuum robots particularly in applications where the robot is subjected to constraint forces from unknown obstacles in its surroundings. In this chapter, we will briefly summarize and compare these methods, as well as suggest potential areas to extend this work.

6.1 Summary and Comparison

Overall, we have presented four new algorithms related to the modeling, estimation, and control of continuum robots and have highlighted the benefits of these new approaches and their potential applications. Chapter 2 presented a novel control strategy for growing continuum robots, utilizing real-time position and orientation measurements. The effectiveness of this approach was demonstrated in both constrained and unconstrained scenarios with a physical system. Chapter 3 introduced online generalized product of experts (OGPoE), an online learning algorithm based on Gaussian Processes. The algorithm's capabilities were showcased through online learning of the kinematics of a concentric tube robot, in both free space and in the presence of dynamic environmental constraints. Chapter 4 focused on the development of a dynamic model specific to cable-driven continuum robots. This model facilitated the design of an observer capable of estimating the robot's full state using only partial state feedback. The observer was integrated into a control loop aimed at rejecting a constant disturbance applied to the robot body, and the algorithm was validated on hardware. Chapter 5 addressed the challenge of segmenting a continuum robot from a noisy image without the need for training data. An algorithm was proposed to accomplish this task, offering a geometric solution to this particular segmentation problem.

The works presented in Chapters 2-4 are directly comparable in that these works require models of the continuum robot that must be updated online from sensor data in order to reflect how unknown obstacles effect the robot shape. The ideal continuum robot model to use in this case, where the robot might interact with unknown obstacles, would have several important qualities, including: high prediction accuracy, ability to capture dynamic effects, global validity, easy interpretability, ability to generalize across robot designs, low complexity parameter updates, ability to update without integrated sensor measurements. Of course, no one model satisfies all of these qualities, but we show in Fig 6.1 the subset that each of our proposed methods does achieve.


Concentric Tubes
EM Sensor
EM Field Generator
Actuation Unit
EM Field Generator
EM Sensor
Robot Backbone
Disturbance

High prediction accuracy	✗	✓	✗
Captures dynamic Effects	✗	✗	✓
Globally valid	✗	✓	✓
Generalizes across robot designs	✗	✓	✗
Easily interpretable	✓	▬	✓
Low complexity update	✓	✗	✓
Works without direct measurements	✗	✗	✓

Figure 6.1. A graphic comparing the modeling attributes of the works presented in Chapters 2 (left column), 3 (center column), and 4 (right column).

The Jacobian-based method shown in Chapter 2 is the least complex and least computationally expensive to use for prediction due to the very low number of parameters it uses to represent the robot Jacobian. However, this method is limited in that it is based on a linear approximation and is specific to the growing robot design. The Gaussian Process based method detailed in Chapter 3 avoids all of these previous issues: it is general across designs, captures the full non-linearity of the problem, and builds a global map that could be used in a multi-step planning algorithm. The cost of these advantages is a drastic increase in complexity and computational cost. Finally, the modeling approach shown in Chapter 4 is unique in that it can predict time-varying behavior of the robot, which makes it more compatible with the vast field of control theory developed for classical robotic systems. Compared with the other approaches it has a moderate level of complexity and relatively low computational cost.

6.2 Continuum Robot Modeling

Chapters 2, 3 and 4 each present a different approach to the problem of modeling a continuum robot subjected to unknown external forces. Given the various tradeoffs of these approaches, a direction of future work might involve unifying the proposed approaches in a way that takes advantage of their respective strengths, while mitigating their weaknesses. One possible approach could be to *learn* low parameter *symbolic* models of the robot dynamics with the goal being that the learned symbolic representation retains some of the high predictive accuracy learning methods are known for, while also being interpretable to the user. Some researchers are exploring algorithms along this line of thought for high dimensional PDE systems already [141]. Given that we have already shown how it is possible to achieve good modeling results with analytical approaches alone, it is reasonable to expect that incorporating symbolic learning may further improve their accuracy without sacrificing interpretability. It may even be possible to learn a model that is guaranteed to be controllable [142] provided the learning problem is posed correctly and its computational tractability is carefully considered.

6.3 Continuum Robot Estimation

Due to the limited nature of the sensor information available for continuum robots, there are usually components of the robot's dynamic or kinematic model that need to be modified online, especially if the robot is being subjected to unknown external forces. The models used in Chapters 2, 3, 4, and 5 all have parameters that must be updated from sensor information. In Chapters 3, and 5, this update is posed as a large, nonlinear optimization which typically finds parameters that fit the sensor data quite well. However, there are many practical challenges associated with translating these approaches outside the laboratory setting due to numerical problems, such as sensitivity to initial conditions and convergence to poor local optima, as well as long run times for nonlinear optimization problems. Nonetheless, improved hardware and optimized implementation of the algorithms may be sufficient to address these issues and are one avenue for future work [143].

The algorithms in Chapter 2 and Chapter 4 avoid the problems of nonlinear optimization by specifying a set of rules that update model parameters in computationally inexpensive and numerically stable ways. These approaches, consequently, are much more readily amenable to real world use. Potential future work in this area of estimation would involve using more analytical techniques from control theory to improve these update rules and guarantee their long term behaviors [144]. Additional future work may also include how best to fuse the many different kinds of sensor measurements that have been used for continuum robots in the past including force-torque, position, tension, image, and Fiber Bragg grating sensors. The opportunity to apply rigorous methods to these continuum robot estimation problems is a relatively new in the literature and we believe that Chapter 4 is a significant contribution to this research.

6.4 Continuum Robot Control

As is seen in Chapters 2 and 4, the problems of estimation and control are tightly coupled, and both depend on the amount of sensor information that is available. Chapter 2 focuses on the scenario where there is sensing integrated into the robot that can be used in the control and estimation loop, while Chapter 4 focuses on the case where this sensing is not available. In both control loops, we assume the robot behaves quasi-statically and use low-pass filtering to address issues of sensor noise. We show that despite these assumptions and the limited sensor data available, it is possible to achieve accurate continuum robot control, even when there are external forces acting on the robots.

There are many possibilities to extend the methods presented in these works. Robust control methods can be developed to mitigate the effect of sensor noise [145] and adaptive approaches may be useful in cases where there are slowly time-varying processes (such as friction) that effect the robot dynamics [146]. Finally, the dynamics model proposed in Chapter 4 could be used in a feedback-linearization style control scheme to achieve high-performance continuum robots capable of rapid movements and minimal oscillations. This will certainly require commensurately high-performance hardware and the question of whether or not the model accuracy is sufficient remains to be answered.

6.5 Broader Considerations

Ultimately, while the development of better algorithms will continue to be a key part in improving continuum robots, it is not the only factor. Design, sensing, modeling, estimation, and control are all fundamentally coupled problems for these robots that will need to be considered in tandem in order to continue making progress. We hope that the work in this dissertation will help to advance the state of the art for these robots, generate new ideas for future researchers, and increase adoption of continuum robot technology outside of the lab where it may more directly impact people's lives.

ACKNOWLEDGEMENTS

Chapter 2 is, in part, a reprint of material from C. Watson, R. Obregon, T.K. Morimoto, ”*Closed-Loop Position Control for Growing Robots Via Online Jacobian Corrections*.” IEEE Robotics and Automation Letters, vol. 6, no. 4, pp. 6820-6827, 2021. Chapter 3 is, in part, a reprint of material from C. Watson, T.K. Morimoto, ”*Learning Non-Parametric Models in Real Time via Online Generalized Product of Experts*.” IEEE Robotics and Automation Letters, vol. 7, no. 4, pp. 9326-9333, 2022. The dissertation author is the primary author of both of these papers.

Chapter 4 contains material that will be submitted for publication under the title ”*Disturbance Rejection for Cable-Driven Continuum Robots without Integrated Sensing*.” in 2023. The dissertation author is the primary author of this work. Chapter 5 is, in part, a reprint of material from C. Watson, A. Nguyen, and T.K. Morimoto. ”*Image Segmentation for Continuum Robots from a Kinematic Prior*.” IEEE/RAS Int. Conf. on Robotics and Automation (ICRA), 2023. The dissertation author is the primary author of this work.

Bibliography

[1] Shigeo Hirose and Makoto Mori. Biologically inspired snake-like robots. In *2004 IEEE International Conference on Robotics and Biomimetics*, pages 1–7. IEEE, 2004.

[2] Deepak Trivedi, Christopher D Rahn, William M Kier, and Ian D Walker. Soft robotics: Biological inspiration, state of the art, and future research. *Applied bionics and biomechanics*, 5(3):99–117, 2008.

[3] Elliot W Hawkes, Laura H Blumenschein, Joseph D Greer, and Allison M Okamura. A soft robot that navigates its environment through growth. *Science Robotics*, 2(8):eaan3028, 2017.

[4] Takeshi Aoki, Akiyoshi Ochiai, and Shigeo Hirose. Study on slime robot: development of the mobile robot prototype model using bridle bellows. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, volume 3, pages 2808–2813. IEEE, 2004.

[5] Guoxin Li, Jingjun Yu, Dailin Dong, Jie Pan, Haoran Wu, Shengge Cao, Xu Pei, Xindong Huang, and Jianqing Yi. Systematic design of a 3-dof dual-segment continuum robot for in situ maintenance in nuclear power plants. *Machines*, 10(7):596, 2022.

[6] Dixit Nahar, Paul M Yanik, and Ian D Walker. Robot tendrils: Long, thin continuum robots for inspection in space operations. In *2017 IEEE aerospace conference*, pages 1–8. Ieee, 2017.

[7] Taoming Liu, Nate Lombard Poirot, Dominique Franson, Nicole Seiberlich, Mark A Griswold, and M Cenk Çavuşoğlu. Modeling and validation of the three-dimensional deflection of an mri-compatible magnetically actuated steerable catheter. *IEEE Transactions on Biomedical Engineering*, 63(10):2142–2154, 2015.

[8] Pierre E. Dupont, Jesse Lock, and Evan Butler. Torsional kinematic model for concentric tube robots. *IEEE Int. Conf. Robot. Automat.*, 2009:2964–2971, May 2009.

[9] Takahisa Kato, Ichiro Okumura, Sang-Eun Song, Alexandra J Golby, and Nobuhiko Hata. Tendon-driven continuum robot for endoscopic surgery: Preclinical development and validation of a tension propagation model. *IEEE/ASME Transactions on Mechatronics*, 20(5):2252–2263, 2014.

[10] Pierre E Dupont, Nabil Simaan, Howie Choset, and Caleb Rucker. Continuum robots for medical interventions. *Proceedings of the IEEE*, 110(7):847–870, 2022.

[11] Matthew Kroh, Kevin El-Hayek, Steven Rosenblatt, Bipan Chand, Pedro Escobar, Jihad Kaouk, and Sricharan Chalikonda. First human surgery with a novel single-port robotic system: cholecystectomy using the da vinci single-site platform. *Surgical endoscopy*, 25:3566–3573, 2011.

[12] Dong Hoon Koh, Won Sik Jang, Jae Won Park, Won Sik Ham, Woong Kyu Han, Koon Ho Rha, and Young Deuk Choi. Efficacy and safety of robotic procedures performed using the da vinci robotic surgical system at a single institute in korea: experience with 10000 cases. *Yonsei medical journal*, 59(8):975–981, 2018.

[13] Thien-Dang Nguyen and Jessica Burgner-Kahrs. A tendon-driven continuum robot with extensible sections. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2130–2135. IEEE, 2015.

[14] Robert J Webster, Allison M Okamura, and Nah J Cowan. Toward active cannulas: Miniature snake-like surgical robots. In *2006 IEEE/RSJ international conference on intelligent robots and systems*, pages 2857–2863. IEEE, 2006.

[15] Yoonho Kim, German A Parada, Shengduo Liu, and Xuanhe Zhao. Ferromagnetic soft continuum robots. *Science Robotics*, 4(33):eaax7329, 2019.

[16] Robert J Webster III and Bryan A Jones. Design and kinematic modeling of constant curvature continuum robots: A review. *The International Journal of Robotics Research*, 29(13):1661–1683, 2010.

[17] John Till, Vincent Aloi, and Caleb Rucker. Real-time dynamics of soft and continuum robots based on cosserat rod models. *The International Journal of Robotics Research*, 38(6):723–746, 2019.

[18] D Caleb Rucker, Bryan A Jones, and Robert J Webster III. A geometrically exact model for externally loaded concentric-tube continuum robots. *IEEE Trans. Robot.*, 26(5):769–780, 2010.

[19] M. C. Yip and D. B. Camarillo. Model-less feedback control of continuum manipulators in constrained environments. *IEEE Trans. Robot.*, 30(4):880–889, 2014.

[20] Kai Xu and Nabil Simaan. Actuation compensation for flexible surgical snake-like robots with redundant remote actuation. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 4148–4154. IEEE, 2006.

[21] D. C. Rucker and R. J. Webster III. Statics and dynamics of continuum robots with general tendon routing and external loading. *IEEE Trans. Robot.*, 27(6):1033–1044, 2011.

[22] Nabil Simaan, Russell Taylor, and Paul Flint. A dexterous system for laryngeal surgery. In *IEEE International Conference on Robotics and Automation, 2004. Proceedings. ICRA'04. 2004*, volume 1, pages 351–357. IEEE, 2004.

[23] Christoff Heunis, Jakub Sikorski, and Sarthak Misra. Flexible instruments for endovascular interventions: Improved magnetic steering, actuation, and image-guided surgical instruments. *IEEE robotics & automation magazine*, 25(3):71–82, 2018.

[24] J. D. Greer, T. K. Morimoto, A. M. Okamura, and E. W. Hawkes. Series pneumatic artificial muscles (spams) and application to a soft continuum robot. In *2017 IEEE Int. Conf. Robot. Automat.*, pages 5503–5510, 2017.

[25] Panagiotis Polygerinos, Dinusha Zbyszewski, Tobias Schaeffter, Reza Razavi, Lakmal D Seneviratne, and Kaspar Althoefer. Mri-compatible fiber-optic force sensors for catheterization procedures. *IEEE Sensors Journal*, 10(10):1598–1608, 2010.

[26] J. M. Croom and J. M. Romano. Visual sensing of continuum robot shape using self-organizing maps. *IEEE Robotics and Automation Letters*, 2(3):4591–4596, May 2010.

[27] Roman A Manakov, Dmitry Yurjevich Kolpashchikov, Vyacheslav Vladimirovich Danilov, Nikita Vitalievich Laptev, IP Skirnevskiy, and Olga Mikhailovna Gerget. Visual shape and position sensing algorithm for a continuum robot. In *IOP Conference Series: Materials Science and Engineering*, volume 1019, page 012066. IOP Publishing, 2021.

[28] Bernhard Weber, Paul Zeller, and Kolja Kühnlenz. Multi-camera based real-time configuration estimation of continuum robots. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3350–3355. IEEE, 2012.

[29] Mohsen Moradi Dalvand, Saeid Nahavandi, and Robert D Howe. Fast vision-based catheter 3d reconstruction. *Physics in Medicine & Biology*, 61(14):5128, 2016.

[30] Chaoyang Shi, Xiongbiao Luo, Peng Qi, Tianliang Li, Shuang Song, Zoran Najdovski, Toshio Fukuda, and Hongliang Ren. Shape sensing techniques for continuum robots in minimally invasive surgery: A survey. *IEEE Transactions on Biomedical Engineering*, 64(8):1665–1678, 2016.

[31] D. B. Camarillo, C. F. Milne, C. R. Carlson, M. R. Zinn, and J. K. Salisbury. Mechanics modeling of tendon-driven continuum manipulators. *IEEE Trans. Robot.*, 24(6):1262–1273, 2008.

[32] Gregory S Chirikjian. Hyper-redundant manipulator dynamics: A continuum approximation. *Advanced Robotics*, 9(3):217–243, 1994.

[33] Valentin Falkenhahn, Tobias Mahl, Alexander Hildebrandt, Rüdiger Neumann, and Oliver Sawodny. Dynamic modeling of bellows-actuated continuum robots using the euler–lagrange formalism. *IEEE Transactions on Robotics*, 31(6):1483–1496, 2015.

[34] Frederic Boyer, Vincent Lebastard, Fabien Candelier, and Federico Renda. Dynamics of continuum and soft robots: A strain parameterization based approach. *IEEE Transactions on Robotics*, 37(3):847–863, 2020.

[35] Caleb Rucker, Eric J. Barth, Joshua Gaston, and James C. Gallentine. Task-space control of continuum robots using underactuated discrete rod models. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10967–10974, 2022.

[36] Reinhard Grassmann, Vincent Modes, and Jessica Burgner-Kahrs. Learning the forward and inverse kinematics of a 6-dof concentric tube continuum robot in se (3). In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5125–5132. IEEE, 2018.

[37] Alan Kuntz, Armaan Sethi, Robert J Webster, and Ron Alterovitz. Learning the complete shape of concentric tube robots. *IEEE transactions on medical robotics and bionics*, 2(2):140–147, 2020.

[38] David A Haggerty, Michael J Banks, Patrick C Curtis, Igor Mezić, and Elliot W Hawkes. Modeling, reduction, and control of a helically actuated inertial soft robotic arm via the koopman operator. *arXiv preprint arXiv:2011.07939*, 2020.

[39] Pierre E Dupont, Jesse Lock, Brandon Itkowitz, and Evan Butler. Design and control of concentric-tube robots. *IEEE Transactions on Robotics*, 26(2):209–225, 2009.

[40] Thomas George Thuruthel, Egidio Falotico, Matteo Cianchetti, and Cecilia Laschi. Learning global inverse kinematics solutions for a continuum robot. In *ROMANSY 21-Robot Design, Dynamics and Control: Proceedings of the 21st CISM-IFToMM Symposium, June 20-23, Udine, Italy*, pages 47–54. Springer, 2016.

[41] Caleb Rucker, Eric J Barth, Joshua Gaston, and James C Gallentine. Task-space control of continuum robots using underactuated discrete rod models. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10967–10974. IEEE, 2022.

[42] Mohamed Taha Chikhaoui, Josephine Granna, Julia Starke, and Jessica Burgner-Kahrs. Toward motion coordination control and design optimization for dual-arm concentric tube continuum robots. *IEEE Robotics and Automation Letters*, 3(3):1793–1800, 2018.

[43] Pierre E Dupont, Jesse Lock, and Brandon Itkowitz. Real-time position control of concentric tube robots. In *2010 IEEE International Conference on Robotics and Automation*, pages 562–568. IEEE, 2010.

[44] Mohsen Mahvash and Pierre E Dupont. Stiffness control of surgical continuum manipulators. *IEEE Transactions on Robotics*, 27(2):334–345, 2011.

[45] Rashid Yasin and Nabil Simaan. Joint-level force sensing for indirect hybrid force/position control of continuum robots with friction. *The International Journal of Robotics Research*, 40(4-5):764–781, 2021.

[46] M. C. Yip and D. B. Camarillo. Model-less hybrid position/force control: A minimalist approach for continuum manipulators in unknown, constrained environments. *IEEE Robot. Automat. Lett.*, 1(2):844–851, 2016.

[47] Michael C. Yip, Jake A. Sganga, and David B. Camarillo. Autonomous control of continuum robot manipulators for complex cardiac ablation tasks. *J. Med. Robot. Res.*, 02(01):1750002, 2017.

[48] Minhan Li, Rongjie Kang, David T Branson, and Jian S Dai. Model-free control for continuum robots based on an adaptive kalman filter. *IEEE/ASME Transactions on Mechatronics*, 23(1):286–297, 2017.

[49] Ali Sadeghi, Alessio Mondini, and Barbara Mazzolai. Toward self-growing soft robots inspired by plant roots and based on additive manufacturing technologies. *Soft Robot.*, 4(3):211–223, 2017.

[50] Patrick Slade, Alex Gruebele, Zachary Hammond, Michael Raitor, Allison M Okamura, and Elliot W Hawkes. Design of a soft catheter for low-force and constrained surgery. In *2017 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, pages 174–180.

[51] Laura H Blumenschein, Margaret Koehler, Nathan S Usevitch, Elliot W Hawkes, D Caleb Rucker, and Allison M Okamura. Geometric solutions for general actuator routing on inflated-beam soft growing robots. *arXiv preprint arXiv:2006.06117*, 2020.

[52] Jamie Luong, Paul Glick, Aaron Ong, Maya S deVries, Stuart Sandin, Elliot W Hawkes, and Michael T Tolley. Eversion and retraction of a soft robot towards the exploration of coral reefs. In *2019 2nd IEEE Int. Conf. Soft Robot.*, pages 801–807.

[53] Joseph D Greer, Laura H Blumenschein, Allison M Okamura, and Elliot W Hawkes. Obstacle-aided navigation of a soft growing robot. In *2018 IEEE Int. Conf. Robot. Automat.*, pages 1–8.

[54] Joseph D Greer, Laura H Blumenschein, Ron Alterovitz, Elliot W Hawkes, and Allison M Okamura. Robust navigation of a soft growing robot by exploiting contact with the environment. *Int. J. Robot. Res.*, 39(14):1724–1738, 2020.

[55] David A Haggerty, Nicholas D Naclerio, and Elliot W Hawkes. Characterizing environmental interactions for soft growing robots. In *2019 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, pages 3335–3342.

[56] Joseph D Greer, Tania K Morimoto, Allison M Okamura, and Elliot W Hawkes. A soft, steerable continuum robot that grows via tip extension. *Soft Robot.*, 6(1):95–108, 2019.

[57] Nicholas D Naclerio and Elliot W Hawkes. Simple, low-hysteresis, foldable, fabric pneumatic artificial muscle. *IEEE Robot. Automat. Lett.*, 5(2):3406–3413, 2020.

[58] M Selvaggio, LA Ramirez, ND Naclerio, B Siciliano, and EW Hawkes. An obstacle-interaction planning method for navigation of actuated vine robots. In *2020 IEEE Int. Conf. Robot. Automat.*, pages 3227–3233.

[59] Margaret M Coad, Laura H Blumenschein, Sadie Cutler, Javier A Reyna Zepeda, Nicholas D Naclerio, Haitham El-Hussieny, Usman Mehmood, Jee-Hwan Ryu, Elliot W Hawkes, and Allison M Okamura. Vine robots: Design, teleoperation, and deployment for navigation and exploration. *IEEE Robot. Automat. Mag.*, 27(3):120–132, 2019.

[60] Haitham El-Hussieny, Usman Mehmood, Zain Mehdi, Sang-Goo Jeong, Muhammad Usman, Elliot W Hawkes, Allison M Okarnura, and Jee-Hwan Ryu. Development and evaluation of an intuitive flexible interface for teleoperating soft growing robots. In *2018 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, pages 4995–5002.

[61] Ahmad Ataka, Taqi Abrar, Fabrizio Putzu, Hareesh Godaba, and Kaspar Althoefer. Observer-based control of inflatable robot with variable stiffness *. In *2020 IEEE/RSJ Int. Conf. Intell. Robots Syst.*

[62] Zhongkai Zhang, Jeremie Dequidt, Alexandre Kruszewski, Frederick Largilliere, and Christian Duriez. Kinematic Modeling and Observer Based Control of Soft Robot using Real-Time Finite Element Method. In *IROS2016 - IEEE/RSJ Int. Conf. Intell. Robots Syst.*

[63] Mrinal Verghese, Florian Richter, Aaron Gunn, Phil Weissbrod, and Michael Yip. Model-free visual control for continuum robot manipulators via orientation adaptation. 2019. arXiv:1909.00450. [Online]. Available https://arxiv.org/abs/1909.00450.

[64] III Robert J. Webster and Bryan A. Jones. Design and kinematic modeling of constant curvature continuum robots: A review. *Int. J. Robot. Res.*, 29(13):1661–1683, 2010.

[65] R. Grassmann, V. Modes, and J. Burgner-Kahrs. Learning the forward and inverse kinematics of a 6-dof concentric tube continuum robot in se(3). In *2018 IEEE/RSJ Int. Conf. Intell. Robots Syst.*

[66] Connor Watson and Tania K Morimoto. Permanent magnet-based localization for growing robots in medical applications. *IEEE Robot. Automat. Lett.*, 5(2):2666–2673, 2020.

[67] Claus Gramkow. On averaging rotations. *Int. J. Comput. Vision*, 42(1):7–16, Apr 2001.

[68] R. Hartley, J. Trumpf, Yuchao Dai, and Hongdong Li. Rotation averaging. *Int. J. Comput. Vision*, 103:267–305, 2012.

[69] John Till, Vincent Aloi, Katherine E. Riojas, Patrick L. Anderson, Robert James Webster III, and Caleb Rucker. A dynamic model for concentric tube robots. *IEEE Trans. Robot.*, 36(6):1704–1718, 2020.

[70] Laura H. Blumenschein, Margaret M. Coad, David A. Haggerty, Allison M. Okamura, and Elliot W. Hawkes. Design, modeling, control, and application of everting vine robots. *Frontiers Robot. AI*, 7:153, 2020.

[71] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer school on machine learning*, pages 63–71. Springer, 2003.

[72] Haitao Liu, Yew-Soon Ong, Xiaobo Shen, and Jianfei Cai. When gaussian process meets big data: A review of scalable gps. *IEEE Neural Netw. Learn. Syst.*, 31(11):4405–4423, 2020.

[73] Matthias W Seeger, Christopher KI Williams, and Neil D Lawrence. Fast forward selection to speed up sparse gaussian process regression. In *Int. Workshop on Artif. Intell. Statistics*, pages 254–261. PMLR, 2003.

[74] Girish Chowdhary, Hassan A Kingravi, Jonathan P How, and Patricio A Vela. Bayesian nonparametric adaptive control using gaussian processes. *IEEE Trans. Neural Netw. Learn. Syst.*, 26(3):537–550, 2014.

[75] Christopher Williams and Matthias Seeger. Using the nyström method to speed up kernel machines. *Adv. neural inf. process. syst.*, 13, 2000.

[76] Michalis Titsias. Variational learning of inducing variables in sparse gaussian processes. In *Artificial intelligence and statistics*, pages 567–574. PMLR, 2009.

[77] Thang D Bui, Cuong Nguyen, and Richard E Turner. Streaming sparse gaussian process approximations. In *Adv. neural inf. process. syst.*, volume 30. Curran Associates, Inc., 2017.

[78] Franziska Meier and Stefan Schaal. Drifting gaussian processes with varying neighborhood sizes for online model learning. In *2016 IEEE Int. Conf. Robot. Autom.*, pages 264–269. IEEE, 2016.

[79] Haitao Liu, Jianfei Cai, Yi Wang, and Yew Soon Ong. Generalized robust bayesian committee machine for large-scale gaussian process regression. In *Int. Conf. Mach. Learn.*, pages 3131–3140. PMLR, 2018.

[80] Yanshuai Cao and David J. Fleet. Generalized product of experts for automatic and principled fusion of gaussian process predictions, 2014.

[81] Duy Nguyen-Tuong, Matthias Seeger, and Jan Peters. Model learning with local gaussian process regression. *Advanced Robotics*, 23(15):2015–2034, 2009.

[82] Seniha Esen Yuksel, Joseph N Wilson, and Paul D Gader. Twenty years of mixture of experts. *IEEE Trans. Neural Netw. Learn. Syst.*, 23(8):1177–1193, 2012.

[83] Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

[84] Brian Wilcox and Michael C. Yip. Solar-gp: Sparse online locally adaptive regression using gaussian processes for bayesian robot model learning and control. *IEEE Robot. Autom. Lett.*, 5(2):2832–2839, 2020.

[85] Michael E. Kepler, Alec Koppel, Amrit Singh Bedi, and Daniel J. Stilwell. Wasserstein-splitting gaussian process regression for heterogeneous online bayesian inference. In *2021 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, pages 9833–9840, 2021.

[86] Ajay Kumar Tanwani and Sylvain Calinon. Small-variance asymptotics for non-parametric online robot learning. *Int. J. Robot. Research*, 38(1):3–22, 2019.

[87] Tamara Broderick, Brian Kulis, and Michael Jordan. Mad-bayes: Map-based asymptotic derivations from bayes. In *Int. Conf. Mach. Learn.*, pages 226–234. PMLR, 2013.

[88] David Blackwell and James B MacQueen. Ferguson distributions via pólya urn schemes. *The annals of statistics*, 1(2):353–355, 1973.

[89] Yining Wang and Jun Zhu. Dp-space: Bayesian nonparametric subspace clustering with small-variance asymptotics. In *Int. Conf. Mach. Learn.*, pages 862–870. PMLR, 2015.

[90] Lehel Csató and Manfred Opper. Sparse on-line gaussian processes. *Neural computation*, 14(3):641–668, 2002.

[91] Ching-An Cheng and Byron Boots. Incremental variational sparse gaussian process regression. *Advances in Neural Information Processing Systems*, 29, 2016.

[92] Marc Deisenroth and Jun Wei Ng. Distributed gaussian processes. In *Int. Conf.Mach. Learn.*, pages 1481–1490. PMLR, 2015.

[93] David Burt, Carl Edward Rasmussen, and Mark Van Der Wilk. Rates of convergence for sparse variational Gaussian process regression. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proc. 36th Int. Conf.Mach. Learn.*, volume 97 of *Proceedings of Machine Learning Research*, pages 862–871. PMLR, 09–15 Jun 2019.

[94] Sethu Vijayakumar and Stefan Schaal. Locally weighted projection regression: An o (n) algorithm for incremental real time learning in high dimensional space. In *Proc. 17th Int. Conf.Mach. Learn.*, volume 1, pages 288–293. Morgan Kaufmann, 2000.

[95] Patrick Sears and Pierre Dupont. A steerable needle technology using curved concentric tubes. In *2006 IEEE/RSJ international conference on intelligent robots and systems*, pages 2850–2856. IEEE, 2006.

[96] Stuart S. Antman. *The Special Cosserat Theory of Rods*, pages 259–324. Springer New York, New York, NY, 1995.

[97] Cédric Girerd and Tania K. Morimoto. Design and control of a hand-held concentric tube robot for minimally invasive surgery. *IEEE Transactions on Robotics*, 37(4):1022–1038, 2021.

[98] Stanford Artificial Intelligence Laboratory et al. Robotic operating system.

[99] Mohsen Khadem, John O'Neill, Zisos Mitros, Lyndon da Cruz, and Christos Bergeles. Autonomous steering of concentric tube robots via nonlinear model predictive control. *IEEE Trans. Robot.*, 36(5):1595–1602, 2020.

[100] Junhyoung Ha, Georgios Fagogenis, and Pierre E Dupont. Modeling tube clearance and bounding the effect of friction in concentric tube robot kinematics. *IEEE Trans. Robot.*, 35(2):353–370, 2018.

[101] Jessica Burgner, Philip J Swaney, D Caleb Rucker, Hunter B Gilbert, Scott T Nill, Paul T Russell, Kyle D Weaver, and Robert J Webster. A bimanual teleoperated system for endonasal skull base surgery. In *2011 IEEE/RSJ Int. Conf. Intell. Robots Syst.*, pages 2517–2523. IEEE, 2011.

[102] Alexandre Capone, Armin Lederer, and Sandra Hirche. Gaussian process uniform error bounds with unknown hyperparameters for safety-critical applications. *arXiv preprint arXiv:2109.02606*, 2021.

[103] D Caleb Rucker and Robert J Webster III. Mechanics of continuum robots with external loading and general tendon routing. In *Experimental Robotics: The 12th International Symposium on Experimental Robotics*, pages 645–654. Springer, 2014.

[104] Priyanka Rao, Quentin Peyron, Sven Lilge, and Jessica Burgner-Kahrs. How to model tendon-driven continuum robots and benchmark modelling performance. *Frontiers in Robotics and AI*, 7:630245, 2021.

[105] Mohamed Taha Chikhaoui, Sven Lilge, Simon Kleinschmidt, and Jessica Burgner-Kahrs. Comparison of modeling approaches for a tendon actuated continuum robot with three extensible segments. *IEEE Robotics and Automation Letters*, 4(2):989–996, 2019.

[106] Christian DeBuys, Florin C Ghesu, Jagadeesan Jayender, Reza Langari, and Young-Ho Kim. Design, modeling, and evaluation of separable tendon-driven robotic manipulator with long, passive, flexible proximal section. *arXiv preprint arXiv:2301.00337*, 2023.

[107] Thomas F Allen, Levi Rupert, Timothy R Duggan, Gabriel Hein, and Kevin Albert. Closed-form non-singular constant-curvature continuum manipulator kinematics. In *2020 3rd IEEE International Conference on Soft Robotics (RoboSoft)*, pages 410–416. IEEE, 2020.

[108] MW Spong. Modeling and control of elastic joint robots. *Mathematical and Computer Modelling*, 12(7):912, 1989.

[109] Mohammad A Khosravi and Hamid D Taghirad. Dynamic analysis and control of cable driven robots with elastic cables. *Transactions of the Canadian Society for Mechanical Engineering*, 35(4):543–557, 2011.

[110] Mrdjan Jankovic. Observer based control for elastic joint robots. *IEEE Transactions on Robotics and Automation*, 11(4):618–623, 1995.

[111] Jinna Qin, Francois Léonard, and Gabriel Abba. Experimental external force estimation using a non-linear observer for 6 axes flexible-joint industrial manipulators. In *2013 9th Asian Control Conference (ASCC)*, pages 1–6. IEEE, 2013.

[112] Joonyoung Kim and Elizabeth A Croft. Full-state tracking control for flexible joint robots with singular perturbation techniques. *IEEE Transactions on Control Systems Technology*, 27(1):63–73, 2017.

[113] Hassan K Khalil. Nonlinear systems third edition.

[114] Roger E Goldman, Andrea Bajo, and Nabil Simaan. Compliant motion control for multisegment continuum robots with actuation force sensing. *IEEE Transactions on Robotics*, 30(4):890–902, 2014.

[115] Gene F Franklin, J David Powell, Abbas Emami-Naeini, and J David Powell. *Feedback control of dynamic systems*, volume 4. Prentice hall Upper Saddle River, 2002.

[116] Richard M Murray. Reference inputs and integral action, 2003.

[117] Shiqing Fang, D. Franitza, M. Torlo, F. Bekes, and M. Hiller. Motion control of a tendon-based parallel manipulator using optimal tension distribution. *IEEE/ASME Transactions on Mechatronics*, 9(3):561–568, 2004.

[118] Wisama Khalil and Etienne Dombre. *Modeling identification and control of robots*. CRC Press, 2002.

[119] Rajarshi Roy, Long Wang, and Nabil Simaan. Modeling and estimation of friction, extension, and coupling effects in multisegment continuum robots. *IEEE/ASME Transactions on Mechatronics*, 22(2):909–920, 2016.

[120] Kevin M Lynch and Frank C Park. *Modern robotics*. Cambridge University Press, 2017.

[121] John Till, Vincent Aloi, Katherine E Riojas, Patrick L Anderson, Robert James Webster III, and Caleb Rucker. A dynamic model for concentric tube robots. *IEEE Transactions on Robotics*, 36(6):1704–1718, 2020.

[122] Vincent A Aloi and D Caleb Rucker. Estimating loads along elastic rods. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 2867–2873. IEEE, 2019.

[123] S. Dambreville, Y. Rathi, and A. Tannen. Shape-based approach to robust image segmentation using kernel pca. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 1, pages 977–984, 2006.

[124] Xiang Fu, Chien-Yi Wang, Chen Chen, Changhu Wang, and C.-C. Jay Kuo. Robust image segmentation using contour-guided color palettes. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[125] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017.

[126] Yue Kang, Hang Yin, and Christian Berger. Test your self-driving algorithm: An overview of publicly available driving datasets and virtual testing environments. *IEEE Transactions on Intelligent Vehicles*, 4(2):171–185, 2019.

[127] Jeffrey Mahler, Florian T Pokorny, Brian Hou, Melrose Roderick, Michael Laskey, Mathieu Aubry, Kai Kohlhoff, Torsten Kröger, James Kuffner, and Ken Goldberg. Dex-net 1.0: A cloud-based network of 3d objects for robust grasp planning using a multi-armed bandit model with correlated rewards. In *2016 IEEE international conference on robotics and automation (ICRA)*, pages 1957–1964. IEEE, 2016.

[128] Mahendra Bhandari, Trevor Zeffiro, and Madhu Reddiboina. Artificial intelligence and robotic surgery: current perspective and future directions. *Current opinion in urology*, 30(1):48–54, 2020.

[129] Danail Stoyanov. Surgical vision. *Annals of biomedical engineering*, 40:332–345, 2012.

[130] David Bouget, Max Allan, Danail Stoyanov, and Pierre Jannin. Vision-based and marker-less surgical tool detection and tracking: a review of the literature. *Medical image analysis*, 35:633–654, 2017.

[131] Samuel Dambreville, Romeil Sandhu, Anthony Yezzi, and Allen Tannenbaum. A geometric approach to joint 2d region-based segmentation and 3d pose estimation using a 3d shape prior. *SIAM journal on imaging sciences*, 3(1):110–132, 2010.

[132] Hang Chang, Qing Yang, and Bahram Parvin. A bayesian approach for image segmentation with shape priors. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2008.

[133] Zoran Zivkovic. Improved adaptive gaussian mixture model for background subtraction. In *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.*, volume 2, pages 28–31. IEEE, 2004.

[134] Marc Braham and Marc Van Droogenbroeck. Deep background subtraction with scene-specific convolutional neural networks. In *2016 international conference on systems, signals and image processing (IWSSIP)*, pages 1–4. IEEE, 2016.

[135] Nuria M Oliver, Barbara Rosario, and Alex P Pentland. A bayesian computer vision system for modeling human interactions. *IEEE transactions on pattern analysis and machine intelligence*, 22(8):831–843, 2000.

[136] Hunter B Gilbert. On the mathematical modeling of slender biomedical continuum robots. *Frontiers in Robotics and AI*, 8:732643, 2021.

[137] Tony F Chan and Luminita A Vese. Active contours without edges. *IEEE Transactions on image processing*, 10(2):266–277, 2001.

[138] Marc Remacle, Vyas MN Prasad, Georges Lawson, Laetitia Plisson, Vincent Bachy, and Sébastien Van der Vorst. Transoral robotic surgery (tors) with the medrobotics flex™ system: first surgical application on humans. *European Archives of Oto-Rhino-Laryngology*, 272(6):1451–1455, 2015.

[139] Anthony R Lanfranco, Andres E Castellanos, Jaydev P Desai, and William C Meyers. Robotic surgery: a current perspective. *Annals of surgery*, 239(1):14, 2004.

[140] Cristian da Costa Rocha, Nicolas Padoy, and Benoit Rosa. Self-supervised surgical tool segmentation using kinematic information. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8720–8726. IEEE, 2019.

[141] Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the national academy of sciences*, 113(15):3932–3937, 2016.

[142] Sumeet Singh, Spencer M Richards, Vikas Sindhwani, Jean-Jacques E Slotine, and Marco Pavone. Learning stabilizable nonlinear dynamics with contraction-based regularization. *The International Journal of Robotics Research*, 40(10-11):1123–1150, 2021.

[143] Erik Smistad, Thomas L Falch, Mohammadmehdi Bozorgi, Anne C Elster, and Frank Lindseth. Medical image segmentation on gpus–a comprehensive review. *Medical image analysis*, 20(1):1–18, 2015.

[144] Shangke Lyu and Chien Chern Cheah. Data-driven learning for robot control with unknown jacobian. *Automatica*, 120:109120, 2020.

[145] Michael Green and David JN Limebeer. *Linear robust control*. Courier Corporation, 2012.

[146] Miroslav Krstic, Petar V Kokotovic, and Ioannis Kanellakopoulos. *Nonlinear and adaptive control design*. John Wiley & Sons, Inc., 1995.