Title
Model-free Approaches to Robotic Manipulation via Tactile Perception and Tension-driven Control

Author
Gutierrez, Kenneth

Publication Date
2021

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA

Los Angeles

Model-free Approaches to Robotic Manipulation via Tactile Perception and Tension-driven Control

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

by

Kenneth Gutierrez

2021

ABSTRACT OF THE DISSERTATION

Model-free Approaches to Robotic Manipulation
via Tactile Perception and Tension-driven Control

by

Kenneth Gutierrez
Doctor of Philosophy in Mechanical Engineering
University of California, Los Angeles, 2021
Professor Veronica J. Santos, Chair

To execute manipulation tasks in unstructured environments, robots use computer vision and a priori information to locate and grasp objects of interest. However, once an object has been grasped, cameras cannot perceive tactile- or force-based information about finger-object interactions. To address this, tactile and proprioception data are used to develop novel methodologies that aid in robotic manipulation once an object has been grasped.

In the first study, a method was developed for the perception of tactile directionality using convolutional neural networks (CNNs). The deformation of a tactile sensor is used to perceive the direction of a tangential stimulus acting on the fingerpad. A primary CNN was used to estimate the direction of perturbations applied to a grasped object. A secondary CNN provided a measure of uncertainty through the use of confidence intervals. Our CNN models were able to perceive tactile directionality on par with humans, outperformed a state-of-the-art force estimator network, and was demonstrated in real-time.

In the second study, novel controllers were developed for model-free, tension-driven manipulation of deformable linear objects (DLOs) using force-based data. Prior works on DLO manipulation have focused on geometric or topological state and used complex modeling and computer vision approaches. In tasks such as wrapping a DLO around a structure, DLO tension needs to be carefully controlled. Such tension control cannot be achieved using vision alone once the DLO becomes taut. Two controllers were designed to regulate the tension of a DLO and precede traditional motion controllers. The controllers could be used for tasks in which maintaining DLO tension takes higher priority over exact DLO configuration. We evaluate and demonstrate the controllers in real-time on real robots for two different utilitarian tasks: circular wrapping around a horizontal post and figure-eight wrapping around a boat cleat.

In summary, methods were developed to effectively manipulate objects using tactile- and force-based information. The model-free nature of the approaches allows the techniques to be utilized without exact knowledge of object properties. Our methods that leverage tactile sensation and proprioception for object manipulation can serve as a foundation for further enhancement with complementary sensory feedback such as computer vision.

The dissertation of Kenneth Gutierrez is approved.

Song-Chun Zhu

Jacob Rosen

Tsu-Chin Tsao

Veronica J. Santos, Committee Chair

University of California, Los Angeles

2021

To mom and dad . . .

who through their hard work, perseverance, and endless support

have always embodied the role-model I want to become

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I am extremely fortunate to be in the position I am in today. I have been provided with opportunities and support that have strengthened my skills as a researcher and engineer. I could not have advanced this far without help from the university and others.

First, I would like to thank Dr. Veronica J. Santos for her advisement and support throughout my graduate tenure. She has been a great advisor to work with, allowing me to be creative with my research while being available if I needed feedback. She is a great coworker and I am glad that I had the opportunity to work under her supervision.

I would like to thank my committee members Dr. Tsu-Chin Tsao, Dr. Jacob Rosen, and Dr. Song-Chun Zhu for their time and their guidance. I am grateful that I had the opportunity to learn from each of you, whether it be from classes, seminars, or research. Thanks to Dr. Tsao for assisting me and being available for discussions about the content about Chapter 4. I can say that your enthusiasm for your research has shaped my experience as a doctoral student.

For Chapters 2 and 3, I would like thank Dr. Randall Hellman for assistance with the robot hardware and its associated software. Thanks to Dr. Eunsuk Chong and Shengxin Jia for discussions concerning machine learning with tactile data for the work discussed in Chapter 3. The material presented in Chapters 2 and 3 were supported by National Science Foundation Graduate Research Fellowship #DGE-1144087 (to K. Gutierrez) and National Science Foundation Award #1461547 (to V. J. Santos). For Chapter 4, I would like to thank Dr. Eunsuk Chong and Lionel Zhang for feedback on the controller design and implementation. We thank Eric Peltola, Aaron Rovinsky, Christina Rice, and Annalise Wulf for assistance with the experimental set-up. We also thank Dr. Khalid Jawed and

Cole Ten for discussions about DLO modelling and manipulability measures, respectively. The work in Chapter 4 was supported in part by the National Science Foundation Graduate Research Fellowship Program under Award #DGE-1650604 and the Office of Naval Research under Award #N00014-18-1-2814. The UCLA Graduate Division has funded me through a Cota-Robles Fellowship for the last years of my doctoral tenure. In particular, I would like to thank Dr. Diana Azurdia and the NSF-LSAMP Bridge to the Doctorate program which provided financial and mental support throughout my doctoral tenure. Any opinions, findings, conclusions, or recommendations are those of the authors and do not necessarily represent the official views, opinions, or policies of the funding agencies.

A special thanks to all the members of the UCLA Biomechatronics Laboratory, past and present. I could not understate how much I learned from all the members of the lab, whether it be learning a new algorithm, coding, or prototyping experimental hardware. A special thanks for former lab member Dr. Randall Hellman for helping me get started with the robot hardware and its associated software.

VITA

2014–Present Graduate Student Researcher, UCLA Biomechatronics Lab, Mechanical and Aerospace Engineering Department, UCLA

2017 Graduate Mechanical Engineering Intern, Intel, Hillsboro, Oregon. Implemented algorithms with data from inertial sensors to aid in the development of an R&D prototype.

2016 M.S. Mechanical Engineering, UCLA, Los Angeles, California

2015 NSF-Graduate Research Fellowship Program fellow. Earned by 12.5% students out of a competitive pool of 16,000 applications.

2014 Process Engineering Intern, Kulicke & Soffa, Santa Ana, California. Developed software using Python to extract information from semiconductor manufacturing machines.

2014 NSF-LSAMP Bridge to the Doctorate fellow. One out of twelve graduate student cohort.

2014 B.S. Mechanical Engineering, UCI, Irvine, California

2012–2013 Summer Research Intern, Massachusetts Institute of Technology (MIT) Summer Research Program, Cambridge, Massachusetts. Performed graduate-level research in VTOL drone control and chemical reformation through combustion.

PUBLICATIONS

Peng, Y., Serfass, C.M., Kawazoe, A., Shao, Y., **Gutierrez, K.**, Hill, C.N., Santos, V.J., Visell,Y., Hsiao, L.C.. "Elastohydrodynamic friction of robotic and human fingers on soft micropatterned substrates." Nature Materials, 2021.

K. Gutierrez and V. J. Santos, "Perception of Tactile Directionality via Artificial Fingerpad Deformation and Convolutional Neural Networks," in IEEE Transactions on Haptics, vol. 13, no. 4, pp. 831-839, Oct.-Dec. 2020.

Gutierrez, K. and Santos, V.J. "Online perception of tactile directionality using Hidden Markov Models." Proc. RSS Workshop on "Tactile Sensing for Manipulation: Hardware, Modeling, and Learning," Boston, MA, July 15, 2017.

Gutierrez, K. and Santos, V.J. "Online perception of tactile directionality using Hidden Markov Models." SoCal Robotics Symposium, Los Angeles, CA, April 14, 2017.

Gutierrez, K. and Santos, V.J. "Haptic perception of object motion within a robot hand." Workshop on Interacting with Robots Through Touch, Irvine, CA, Sept. 13, 2016.

Gutierrez, K. and Santos, V.J., "Perception of object motion within the hand using multimodal tactile sensors." Emerging Researchers National Conf in STEM, Washington, DC, Feb. 25-27, 2016. Podium. Won 1st place in the Engineering and Technology category.

CHAPTER 1

Introduction

1.1 Motivation

The confluence of the fields of robotics, machine learning, and artificial intelligence has been on the rise since 2000. Experts have marked the time period between 2000 and the present as the fourth industrial revolution where these technologies are entering multiple fields and disciplines [2]. As society moves towards a more digital age, the reliance on robots is increasing and society is demanding more functions and features out of robotic hardware. This has opened the door for many research opportunities in robotics in pursuit of implementing hardware that is semi-autonomous and efficient.

A robot's ability to grasp and manipulate objects in unstructured environments is essential for impact outside the laboratory setting. Grasping involves a multi-stage process of perceiving the position and orientation of an object, calculating optimal grasp points, moving the robot to the optimal position needed to grasp the object, and reactively adjusting fingertip forces, as needed. Traditionally, object localization has been performed using computer vision in combination with machine learning, such as support vector machines, Adaboost, and neural networks [3]. Grasping is outside the scope of this doctoral dissertation, which is focused on sensor-driven techniques for manipulation.

Manipulation involves the changing of the state of an object through physical interactions

with a robotic manipulator. By actively monitoring the state of an object, a model is developed that couples the state of the object to the actions of the robot. Methods in this area involve dynamic movement primitives (DMPs), reinforcement learning, and imitation learning to capture the changes in state based on the actions of the robot [4]. In practice, perfect models of objects and the environment are unavailable for unstructured environments. As a result, grasp and manipulation are often approached in a model-free manner.

Due to the unpredictable nature of unstructured environments, many roboticists have focused on developing models that allow the robot to perceive their local environment. Perception is key to integrating robotic hardware into everyday society as it provides the foundation for grasping and manipulation tasks. The state-of-the-art methods in this field are usually based on data from a combination of optical and position-based sensors such as a RGB-D cameras, LiDAR, GPS antennas, and much more. These sensors have been shown to excel in grasping, especially when combined with learning for object pose estimation and localization. However, these sensors are mostly used for computer vision-based approaches to grasp and manipulation planning.

When it comes to manipulation, vision-based sensors are not sufficient for observing tactile and force-based states that are necessary for object manipulation tasks [5]. Therefore, tactile sensors are utilized to provide information about the contact state between the gripper and the object. Examples of these sensors consist of tactile arrays, optical sensors, and biomimetic sensors [6].

Tactile sensing is essential for optimizing the execution of object manipulation methods. While computer vision-based techniques for object manipulation have advanced, the sense of touch is needed to monitor physical contacts between the gripper and the object. Vision alone does not provide the instantaneous feedback needed to react to contact events and plan

future end-effector movements. Visual feedback can be limited or absent due to occlusion by the gripper digits, a lack of a direct line of sight, and a lack of visible effects of forces being applied to the object by the fingertips. Thus, there is a need for new tactile perception algorithms that can complement mature computer vision approaches to manipulation.

1.2 Contributions

The work in this dissertation presents multiple methods for manipulating objects using tactile sensing and force feedback. All of the methods developed are model-free, thus requiring no a priori knowledge of exact object properties. Through machine learning, we developed a technique to perceive tactile directionality through the tangential stimulation of a deformable fingerpad. Controllers were designed for the manipulation of deformable linear objects (DLOs) with a focus on regulating the tensile state of the DLO. The two described techniques enable tactile perception in any robotic system. With tactile perception capabilities, robots will be able to make context-based decisions during object manipulation tasks.

Chapters 2 and 3 present methods for the perception of tactile directionality through the use of deformable tactile sensors. The work presented in these chapters uses machine learning to develop classification and regression models. These models estimate the direction of perturbations being applied to a handheld object. In Chapter 2, Hidden Markov Models (HMMs) were used to train classification models that provide a likelihood of perturbation direction based on the pressure and skin deformation of a multimodal tactile sensor [7]. In order to enhance the angular resolution of the tactile directionality estimates, HMMs were exchanged for Convolutional Neural Networks (CNNs). In Chapter 3, CNN models were developed using tactile images generated from the fingerpad deformation of the tactile sensor. A CNN regression model was trained to produce a point estimate of tactile directionality while another CNN model produced confidence intervals around the point estimate [8]. The training and evaluation of each model is discussed along with a real-time demonstration on a real robot.

Chapter 4 describes the development of force controllers for use in the manipulation of DLOs. Two novel controllers are developed to enable tension-driven manipulation of DLOs. The controllers are designed to precede a traditional motion controller in any robotic system. The stability of these controllers are discussed along with an analysis of the implementation on real hardware during two utilitarian wrapping tasks. The first task explored in this section is the wrapping of a deformable linear object around a horizontal post. Afterwards, we utilized a controller for a figure-eight wrapping of a rope around a vertical boat cleat and provided guidance for the practical implementation of these controllers on real robots.

Chapter 5 summarizes this dissertation work and presents potential avenues for application and future work.

CHAPTER 2

Perception of Tactile Directionality via Artificial Fingerpad Deformation: Discrete Model Approach

This chapter was based on work presented at the 2017 RSS Workshop on Tactile Sensing for Manipulation [7].

2.1 Introduction

Robots have moved out of structured factory environments into unstructured human environments that are dynamic, unpredictable, and often contain unknown objects [1]. For tasks requiring physical interaction between robots and their environment, tactile feedback enables early detection and perception of forceful interactions between the hand (or handheld objects) and the environment. While computer vision remains a viable option for performing grasping [2], a parallel haptic system would complement the visual system and improve overall performance by providing additional information that cannot be gleaned visually. In this work, we develop capabilities for the haptic perception of tactile directionality, or the "ability to tell the direction of an object's motion across the skin" [3]. With proprioception and knowledge of gravity, a robot that can perceive tactile directionality might be able to

more efficiently perform an object handover or more gently place an object on a tabletop, for example.

In this chapter, we used Hidden Markov Models (HMMs) to process a time-series of multimodal tactile data for the perception of tactile directionality. In various fields such as natural language processing, forecasting, and computational biology, HMMs are used to learn probabilistic models for a time-series of data [9]. Through the use of a multimodal tactile sensor with multivariate continuous output, we trained HMMs using the skin deformation and internal pressure data during the perturbation of a handheld object. Through this methodology, the temporal information of the tactile data was used to perceive the direction of perturbation of a grasped object.

2.2 Methods

A table-mounted BarrettHand (Barrett Technology, Newton, MA) was equipped with two BioTac multimodal tactile sensor fingertips (SynTouch, Los Angeles, CA) (Figure 2.1b). The BioTac records information about internal fluid pressure, vibration, skin deformation of the fingerpad, and temperature. The DC pressure and impedance signals from each tactile sensor are sampled at 100 Hz. Each tactile sensor had a 50° contact angle with the surface of a rigid, parallel-faced object.

The object was attached to the end of a 7DOF Barrett WAM (Barrett Technology, Newton, MA), which was used to perturb the object within the plane perpendicular to the grip axis (axis connecting the two BarrettHand fingertips) (Figure 2.1a). Each perturbation was comprised of a 15 mm linear displacement of the grasped object in different directions tangential to the fingertip (Figure 2.1c). The perturbations ranged from $0° - 360°$ in $10°$

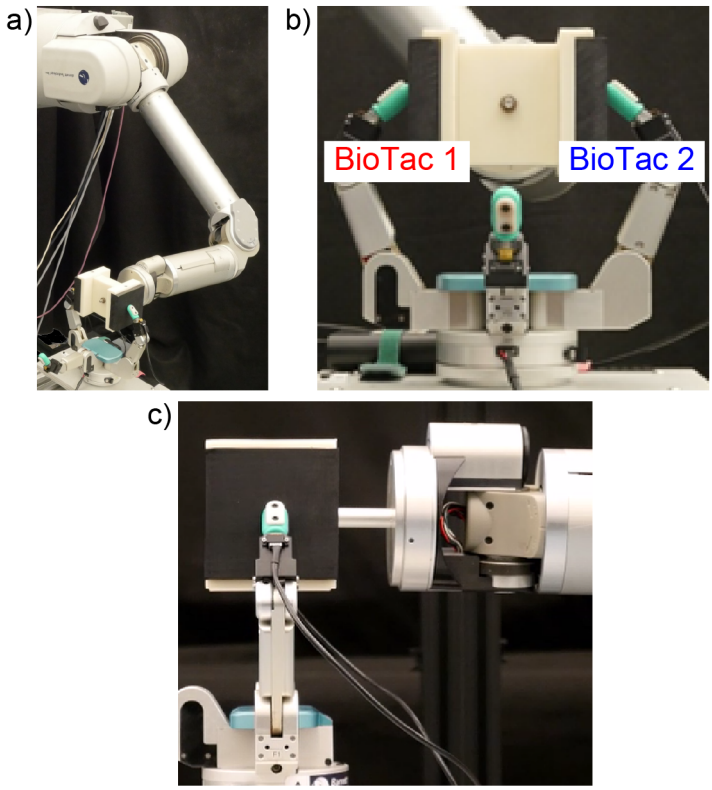

Figure 2.1: (a) A Barrett WAM perturbed an object grasped by a (b) BarrettHand equipped with BioTac tactile sensors. (c) The grasped object was displaced linearly in different directions tangential to the fingertip.

increments and were created through an inverse kinematic solver implemented in MATLAB (MathWorks, Natick, MA). Additionally, perturbation speed (2 cm/s, 4 cm/s) and grip force (low, high estimated from the BioTac fluid pressure) were varied similar to experiments on human perception of movement direction [10].

Data were collected in separate sessions over the course of three days. This was done to add variation to the training and test data and to prevent overfitting. Each experimental session included 3 replicates of each possible perturbation. Five separate experimental sessions, each consisting of 432 randomized trials (3 replicates x 2 forces x 2 speeds x 36 directions), resulted in a total of 2160 trials. Baseline tactile sensor data from each initial, static grasp of the object were subtracted from all subsequent data in each trial. The changes in tactile sensor data during the initial 500 ms of perturbation movement were used to train Hidden Markov Models (HMMs). HMMs are temporal probabilistic models that have been used for signal processing and classifying continuous data [11]. The size of the input observation matrix was 40 x 50 (1 pressure and 19 electrode changes for each of the two sensors in the 500 ms period with a sampling rate of 100 Hz) which contains a multivariate observation vector $\mathbf{x}_t$ for each time step.

$$\mathbf{x}_t = \{\mathbf{e}_1^T, p_{dc1}, \mathbf{e}_2^T, p_{dc2}\} \tag{2.1}$$

Each HMM was associated with a bin created by dividing the 360° into separate sectors. Based on a 30° angular resolution of human tactile direction sensibility [10], we divided the 360° circle into 12 distinct 30° sectors. For a review of the HMM training and parameter optimization, please refer to Appendix A.

2.3 Results and Discussion

2.3.0.1 5-Fold Cross Validation

A 5-fold cross validation was conducted to assess model performance; each fold was comprised of a separate data collection session. The data were pooled for both training and testing. To simulate a real-time scenario, all 12 HMM log likelihoods were calculated for each time step in the 500 ms perturbation period. For each time step, the preceding 250 ms of data (40 x 25 input observation matrix) was used for the log likelihood calculations.

Log likelihoods of the test trials were plotted over time for each HMM. Figure 2 shows the average log likelihoods of all 12 HMMs for trials that belong to the sector associated with a distal perturbation. In the initial 100 ms, all HMMs calculate relatively the same log likelihood. Afterwards, the log likelihoods of the incorrect HMMs start to decrease quickly compared to that of the correct HMM, for which the value remains consistent. The log likelihood for the correct HMM will be significantly larger than those of the incorrect HMMs. This trend is common to all 12 sector-specific HMMs.

The overall 5-fold cross validation error rate decreases as time elapses after the onset of the perturbation (Figure 3). This could be due to the increase in tactile sensor skin stretch as the displacement of the grasped object increases with time. Larger changes in skin stretch from unperturbed baseline values are easier to distinguish from noise. The mechanical deformation of the artificial fingerpad appears to encode directional information about tangential skin displacement, similar in principle to the biological fingerpad [10]. For each of the five folds, each sector had 36 test trials. The number of errors for each sector was averaged across the five folds and the overall error rate is reported in Figure 2.2 at specific time steps of interest. By 500 ms after the onset of perturbation, the maximum error rate

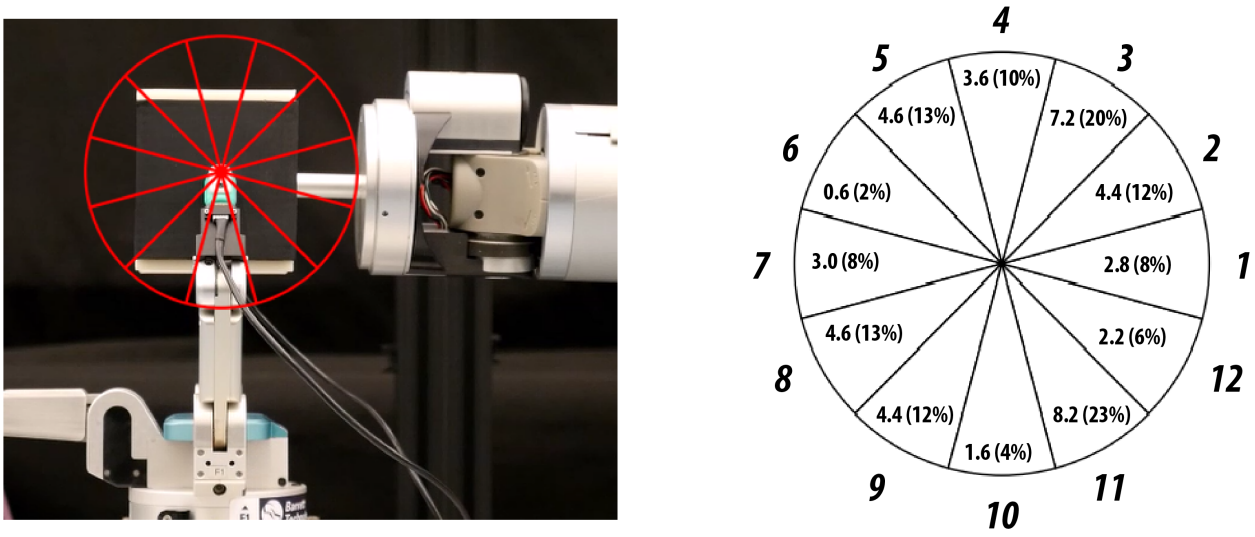

Figure 2.2: (a) The tangential plane of motion was divided into 12 distinct 30° sectors. (b) The 5 fold cross-validation results are shown at 500 ms after the start of the perturbation. Averaged across five folds each having 36 test trials, the number of errors is reported for each sector followed by the error rate in parentheses.

is 23% for Sector 8 (Figure 2.2b). The error rate at 500 ms is less than or equal to 13% for most sectors.

2.3.0.2 Real-Time Implementation

The trained HMMs were tested in a real-time hardware demonstration for an object handover and placement task (Figure 2.3). As with the 5-fold cross validation, the preceding 250 ms of data are utilized to calculate the log-likelihoods in real time. The robot was programmed to respond in a context-appropriate manner when a user-specified threshold criterion was met. Inspired by the threshold criterion in [6], the maximum log likelihood of a single HMM had to exceed all other log likelihoods by a minimum threshold β. There are speed and accuracy trade-offs with the selection of β. If β is too small, the robot may make a short latency, but incorrect response. If β is too large, the robot may wait too long to make its context-appropriate response.

To mimic an object handover, an experimenter attempted to pull a box in a distal direction relative to the robot's reference frame, and the robot perceived the resulting tangential stimulus (green pie chart sector in Figure 2.3a). The robot then proceeded to an object placement task. Upon contact between the handheld object and a horizontal platform, the robot perceived tangential stimulation in the radial direction relative to its reference frame (Figure 2.3b). The hand released the box once the β threshold was met. In this preliminary demonstration, a β threshold of 10,000 was met within 700 ms of the start of an object perturbation.

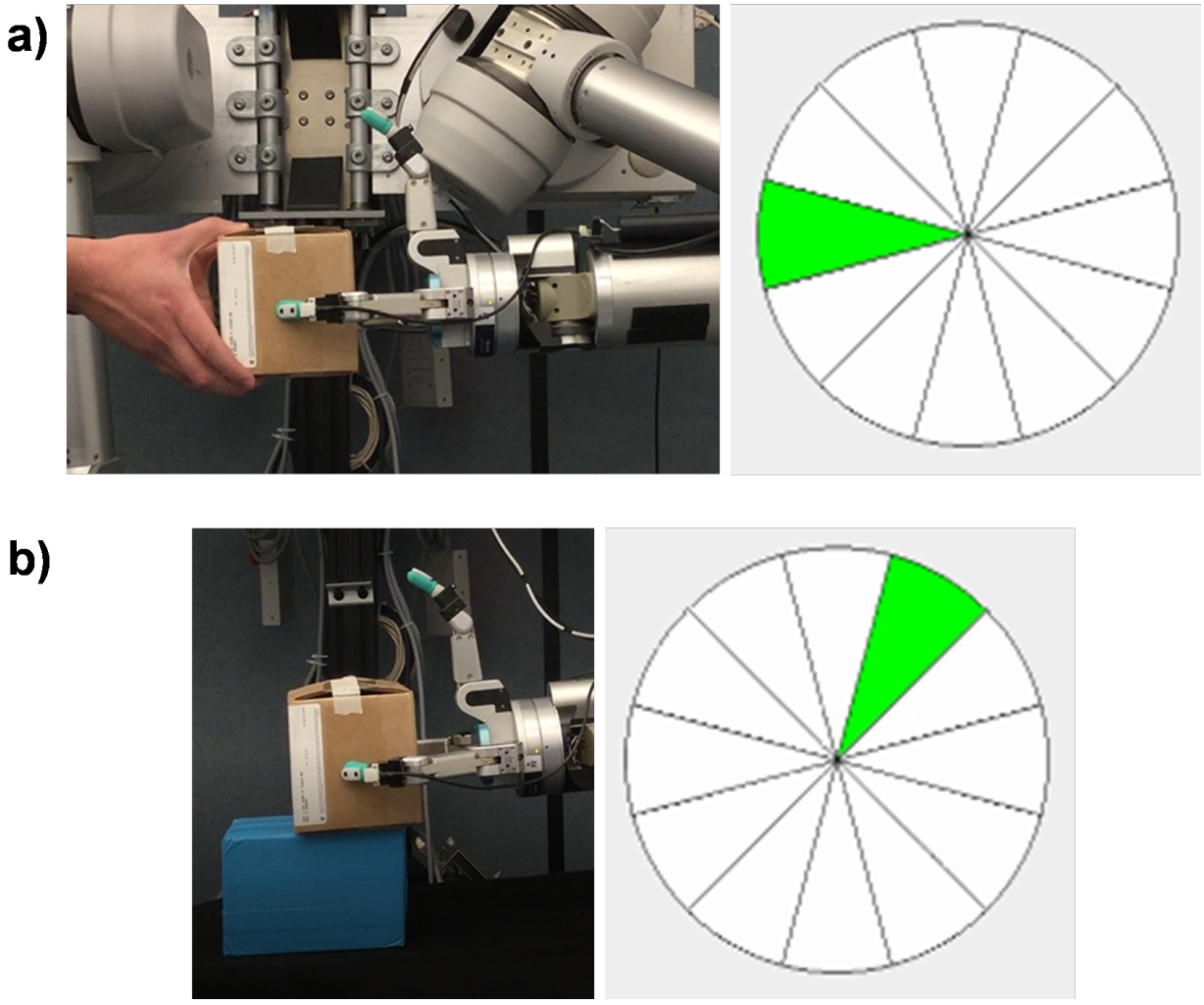

Figure 2.3: Online perception of tactile directionality during a) mock object handover and b) object placement.

2.4 Conclusion

We have developed capabilities for the online perception of tactile directionality using HMMs that can be used to complement visual feedback during grasp and manipulation. In this work, we discretized movement into 30° sectors based on human tactile direction sensibility [10], but this artificial angular resolution can be refined by using more precise robot movements for model training and collecting more training data to capture variation in the tactile sensor data. If a finer angular resolution is desired, a continuous model of tactile directionality could be developed using Recurrent Neural Networks or other models where regression can be utilized. Ongoing work includes the verification of HMM accuracy in real-time, unstructured scenarios. The generalizability of the model to other conditions (e.g. displacement rates) requires further investigation. Tasks could be performed more efficiently and safely if physical interactions with other agents and the environment could be perceived through handheld objects.

CHAPTER 3

Perception of Tactile Directionality via Artificial Fingerpad Deformation: Continuous Model Approach

This chapter was based on work published in the IEEE Transactions on Haptics [8].

3.1 Abstract

Humans can perceive tactile directionality with angular perception thresholds of $14-40^{\circ}$ via fingerpad skin displacement. Using deformable, artificial tactile sensors, the ability to perceive tactile directionality was developed for a robotic system to aid in object manipulation tasks. Two convolutional neural networks (CNNs) were trained on tactile images created from fingerpad deformation measurements during perturbations to a handheld object. A primary CNN regression model provided a point estimate of tactile directionality over a range of grip forces, perturbation angles, and perturbation speeds. A secondary CNN model provided a variance estimate that was used to determine uncertainty about the point estimate. A 5-fold cross-validation was performed to evaluate model performance. The primary CNN produced tactile directionality point estimates with an error rate of 4.3% for a 20°angular resolution and was benchmarked against an open-source force estimation network. The model was implemented in real-time for interactions with an external agent and the environment

with different object shapes and widths. The perception of tactile directionality could be used to enhance the situational awareness of human operators of telerobotic systems and to develop decision-making algorithms for context-appropriate responses by semi-autonomous robots.

3.2 Background

Humans leverage multimodal feedback modalities (e.g. visual, tactile, proprioceptive) during the execution of object manipulation tasks [12]. Visual feedback is especially useful for planning how to make contact with an object of interest or how to move a handheld object for completion of a task. When fingerpads make contact with an object, tactile feedback becomes essential for providing information about the mechanical interactions between the fingers and the object. For example, the task of lifting and replacing a box consists of phases punctuated by contact events [13]. Proprioception (e.g. musculotendon length, joint angles, musculotendon force) can also be used to provide information about the contact mechanics. For example, the size of a handheld object can be estimated through proprioceptive signals from joint afferents by signaling the distance between digits [14, 15]. However, proprioception does not encode the local contact state between a hand and an object, due to the lower sensitivity of non-digital mechanoreceptive afferents compared to those in fingertips [12, 16, 17, 18]. In this work, we focus on the use of tactile data, which are essential for tracking the contact phases and key events of manipulation tasks.

Robots can be designed to leverage tactile data during the execution of manipulation tasks. For tasks requiring physical interaction between robots and their environment, tactile feedback enables the perception of forceful interactions directly between the hand and the

environment or indirectly through a handheld object. While computer vision remains a viable option for planning grasps [19], a parallel haptic system would complement the visual system and improve overall performance by providing additional information that cannot be gleaned visually. With proprioceptive and tactile data, a robot can draw inferences about the physical interactions between itself and the environment or another agent.

Artificial tactile sensors have been utilized to complement other sensing modalities during robot manipulation tasks. While tactile sensors come in many different forms [6], such sensors are all typically used to perceive the contact state between a manipulator and an object. Tactile arrays have been utilized to estimate and manage grip force during the grasp of fragile objects [20]. A grip controller was developed that used the multimodal BioTac tactile sensor (SynTouch, Inc.) to estimate contact forces, detect slip events during pick and place tasks, and distinguish between linear and rotational slip [21].

Tactile information has, more recently, been incorporated into machine learning schemes. Machine learning has grown in popularity as a method for processing tactile sensor data, especially for scenarios in which models of tactile sensors do not exist or nonlinear relationships are sought between tactile stimuli and abstract features. Wettels et al. utilized machine learning techniques and algorithms to extract a force vector, point of force application, and object curvature using a BioTac sensor [22]. Multiple groups have also utilized neural networks (NNs) to determine the force acting on the BioTac during operation of a force feedback controller [21, 1].

Tactile sensor-based machine learning has been applied to slip classification tasks by using NNs on data from bio-inspired sensors [21] and optical sensors [23]. Su et al. used an NN on a time series of BioTac data for slip classification, but required 1 sec worth of tactile data for classification, which precludes sub-second haptic perception [21]. A deep neural

network was used to perceive whether an object was slipping from grasp using data from the GelSight optical sensor along with a camera mounted on the gripper [23]. An NN for the perception of slip direction was developed using BioTac data, but the model focused on four discrete directions (north, south, east, and west) and did not account for contact force magnitude [24].

The contributions of this chapter include the following novel items:

- A primary CNN that produces point estimates of tactile directionality
- A secondary network that provides variance estimates that can be used to determine uncertainty about the point estimates.
- Real-time implementation of the primary network used in interactions with an external agent with different objects shapes and widths.

While the tactile directionality model could be applied to situations in which a sensorized fingertip makes direct contact with an object, the model was trained for scenarios in which external stimuli perturb a handheld object, such as when a handheld object makes direct contact with the environment. The model and the robot's enhanced situational awareness are demonstrated in real-time for perturbations to a handheld object.

3.3 Experimental Procedure and Evaluation

3.3.0.1 Robot Testbed

Tactile data were collected using a 7DOF Barrett WAM manipulator equipped with a 3-digit BarrettHand gripper (Barrett Technology, Newton, MA) (Figure 3.1a). Multimodal

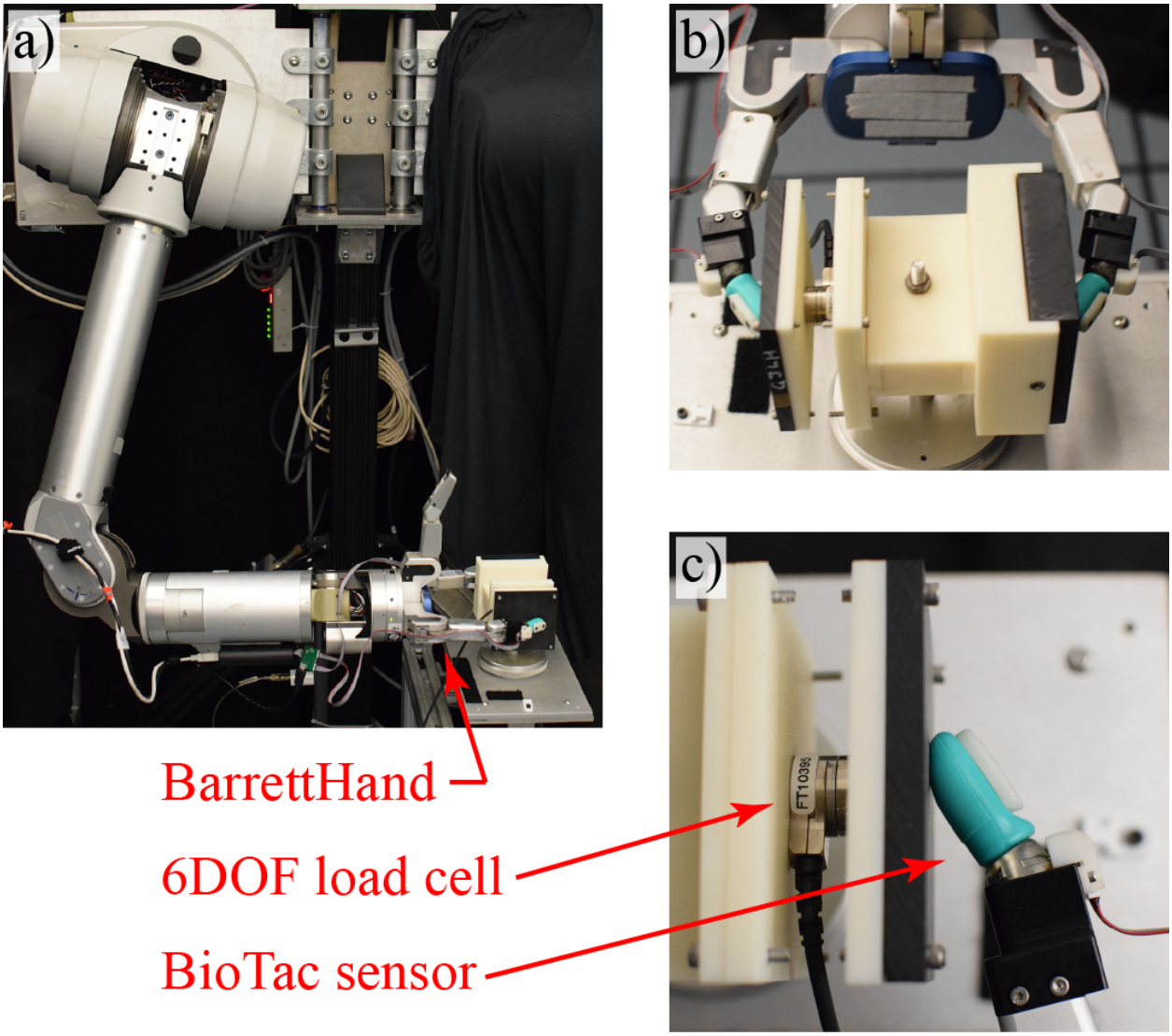

Figure 3.1: a) BarrettHand gripper grasps a parallel-faced object mounted on a low friction slider; b) Top view of the two-digit grasp; c) BioTac tactile sensor contacts the object (embedded with a load cell) at a 30° contact angle

BioTac tactile sensors were attached to two opposing digits of the BarrettHand. Each BioTac sensor enables the measurement of internal fluid pressure, vibration, electrode impedance associated with fingerpad deformation, and temperature flux. The BarrettHand grasped a 4.75 in. wide parallel-faced object with a contact angle of 30° at each fingertip (Figure 3.1b). A Nano17 6DOF force/torque sensor (ATI Industrial Automation, Apex, NC) was centered beneath one face of the object (Figure 3.1c). The object was mounted to a low friction slider whose single degree of freedom was parallel to the robot's grip axis, which connects the two fingertips. The slider automatically centered the object within the robot grasp and distributed grip force equally across the two digits. The object's 3 in. x 3 in. contact surface was composed of polycarbonate plastic that was sanded with 80 grit sandpaper to ensure a uniform texture.

3.3.1 Data Collection

Each trial began with the sensorized object in a stable two-digit grasp. Baseline electrode impedance values were extracted from 3 sec of stable grasp. The grip posture was kept constant throughout each trial, but the grip force ranged from 5-10 N. End-effector trajectories were designed to move the gripper parallel to the object's contact surfaces (Figure 3.1c). Each linear trajectory had randomized parameters of perturbation angle and perturbation speed. The perturbation angle θ (Figure 3.2) ranged from $0 - 360°$ in 10° increments. The end-effector movements were planned using MoveIt's inverse kinematics solver [25]. Inspired by human subject experiments [10, 26], perturbation speeds were varied between 2, 4, or 6 cm/s. Upon execution, each perturbation induced a 20 mm linear displacement of the gripper with respect to the fixed grasped object. Although prior works showed that displacements on the order of 0.2 to 0.5 mm were sufficient for tactile directionality perception by humans

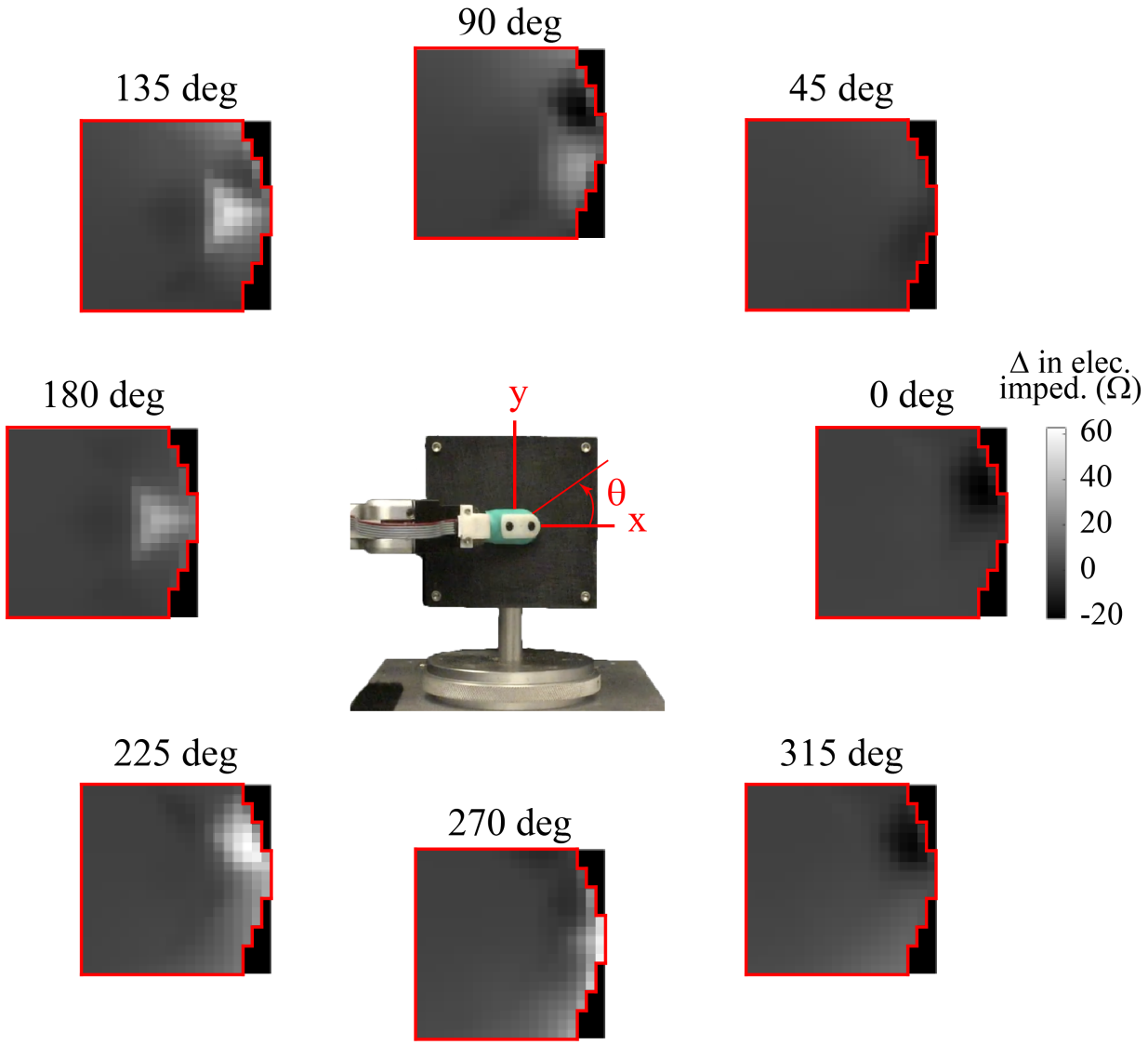

Figure 3.2: A representative set of tactile images. Light and dark pixels indicate areas of fingerpad compression and bulging, respectively. The red line marks the outline of the electrode array.

[10, 27], we used a greater displacement in order to collect tactile sensor signals over a wider range of fingerpad deformation. Upon completion of each perturbation, the gripper released the object and reset its position for the next trial.

Tactile data were collected in three independent experiment sessions in order to add variation to the training and test datasets. Each experiment session included 8 replicates of each perturbation case. Although the 36 perturbation angles were commanded randomly and with equal frequency, compliance in the manipulator led to imprecision in the actual perturbation angle (verified by the force/torque sensor within the grasped object). As a result, we undersampled the tactile dataset in order to minimize bias for any particular direction. Our model trained on 1560 perturbation trials in which the perturbation angles were equally distributed within the contact plane of fingertip travel. We divided the trials into five folds for use in cross-validation testing; 80% of the data were used for training while the remaining 20% were used to test the model.

3.3.2 Data Processing

It was hypothesized that tactile directionality information would be encoded in the BioTac electrode impedance values, which are directly affected by deformation of the soft BioTac fingerpad. The impedance signals are not well-modeled in the literature; no analytical model exists that captures the complex coupling of the deformation of the elastomer skin, mechanics of the conductive fluid, and the electrical current pathways within the fluid. As such, we mapped the nonlinear sensor data to tactile directionality by forming tactile images from impedance values and leveraging well-established machine learning architectures developed for image-based pattern recognition.

First, baseline electrode impedance values from the stable, pre-perturbation grasp were

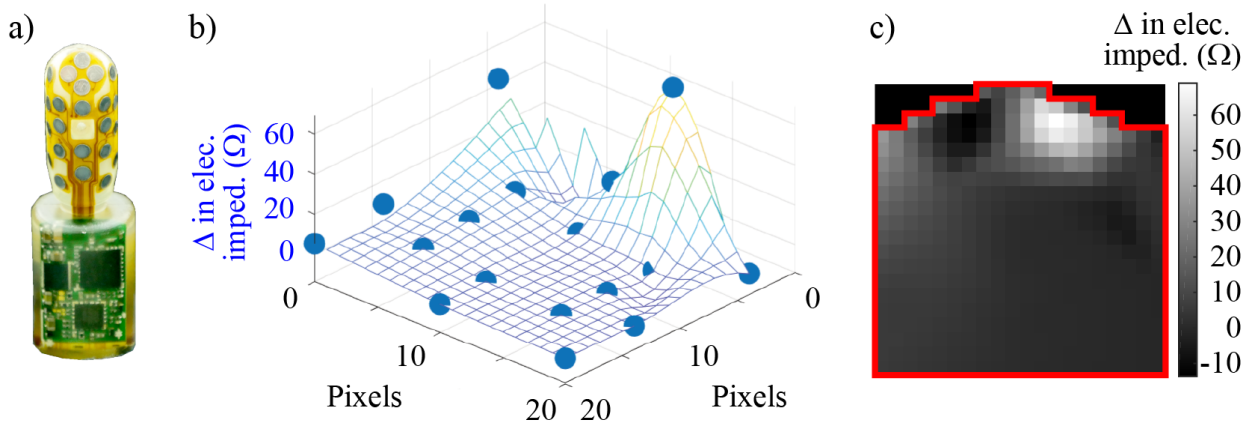

Figure 3.3: a) Exposed BioTac sensor core with 19 electrodes; b) Values for changes in electrode impedance mapped and interpolated over a 20 x 20 pixel grid based on electrode location; c) 2D grayscale tactile image generated from the 3D surface shown in b) in which light and dark pixels indicate areas of compression and bulging, respectively. The red line marks the outline of the electrode array.

subtracted from the post-perturbation values. The resulting changes in impedance for the BioTac's 19 electrodes were mapped onto a 2D spatial representation of the electrode layout (as shown in [28]) in order to create a 3D surface. In order to create a tactile image at each timestep, the changes in electrode impedance were mapped spatially using a triangle-based cubic interpolation that was flattened across a square 20 x 20 pixel grid (Figure 3.3). Positive and negative changes in electrode impedance indicated that the BioTac skin was compressed toward or bulged away from the sensor's core, respectively. The grayscale tactile images were used as inputs to a CNN for use in supervised learning.

With electrode impedance values sampled at 100 Hz, each trial yielded a total of 41 tactile images from the last 410 ms of each perturbation. The total number of tactile images was 63,960 (1560 samples x 41 timesteps). The tactile images were shuffled so that the CNN

would not learn from temporal dynamics, but rather from a single tactile image snapshot. While there may be information overlooked by not analyzing the temporal components of the tactile data, we deliberately focused on spatial features for computational efficiency and real-time implementation.

Based on the shear force measurements from the force/torque sensor, training labels were assigned to each perturbation trial for supervised learning. The changes in the 2D shear forces were used to calculate the perturbation angle (3.1).

$$\theta = \tan^{-1}\left(\frac{\Delta F_y}{\Delta F_x}\right) \tag{3.1}$$

where ΔF_x and ΔF_y were the changes in shear force along the distoproximal and radioulnar axes of the BioTac, respectively. Given that the training data were collected by moving the robot gripper relative to a fixed object, all tactile images were labeled with perturbation angles that had been shifted by 180^{o} in order to represent the perturbation of a handheld object relative to a fixed gripper. In order to uniquely estimate tactile directionality angles in the [0°, 360°] range, each trial was labeled with two continuous values, namely the sin and cos values of the perturbation angle.

3.3.3 Point Estimates for Tactile Directionality

Traditionally, CNNs are used to classify images by analyzing spatial features within the images. In this work, we leverage the CNN approach in order to analyze low-resolution tactile images created from data from a deformable tactile sensor. We built our CNN upon a Pytorch framework [29]. We based the network on a classifier that was used to discriminate classes from the MNIST handwritten digit dataset [30], as the MNIST dataset consists of low-resolution (28 x 28 pixel), grayscale images which are similar in size and resolution to

our dataset of tactile images.

The CNN architecture was composed of three convolutional layers with a fully connected (FCN) layer at the output. The input to the CNN was a 20 x 20 pixel, grayscale tactile image whose pixel intensity values corresponded to changes in tactile sensor electrode impedance. The first convolutional layer consisted of 64 filters, or feature maps, having a 4 x 4 pixel kernel size with a 2 x 2 pixel stride. The number of filters used for the second and third convolution layers were 128 and 256, respectively. Batch normalization was applied to the output of the second and third convolutional layers in order to normalize the layer outputs, allow for higher learning rates, and act as a regularizer to prevent overfitting [31]. A leaky rectified linear unit (ReLU) was also applied after each convolutional layer. The output of the third convolution layer was flattened onto 1024 nodes for input to the final fully connected layer. The final FCN layer output consisted of two continuous values that represented the sin and cos values of the perturbation angle. The sin and cos values were normalized in order to constrain the perturbation direction vector to a unit vector.

The model was trained by optimizing the CNN weights to minimize the mean square error cost function (3.2).

$$C_{MSE} = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3.2}$$

where y_i is the target value, $\hat{y}_i$ is the value estimated by the CNN, and n is the batch size during training of the network. To check for overfitting, the MSE loss during training was compared to that during testing. Training and testing MSE losses converged to a similar value, suggesting that the model was not overfit to the training data (Figure 3.4).

The model hyperparameters that affect the training of the CNN were defined as follows. The batch size is the number of random samples used in the loss function calculation before performing backpropagation to determine changes to the CNN weights. We set the batch

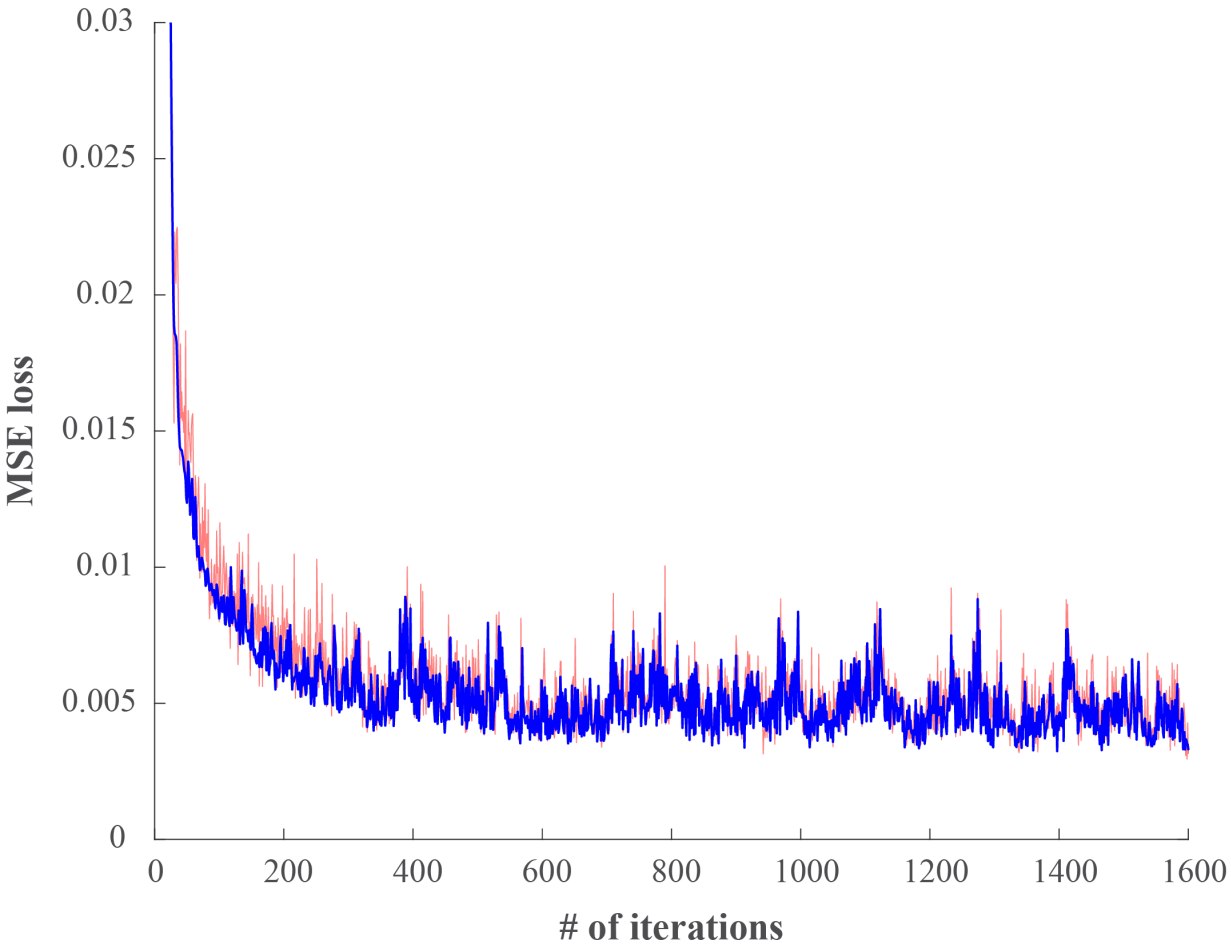

Figure 3.4: The MSE loss is shown for the CNN training batches (thin red line) and with the trained CNN applied to the test data (thick blue line) for a representative fold of cross–validation testing

size to 256 in order to update the CNN weights over the course of 200 iterations per epoch. The number of epochs defines the number of times that a forward pass is made through all of the training tactile images after the batch-style iterations have been completed. That is, one epoch would be completed after 200 iterations of backpropagation since the 51,168 training tactile images would be sampled in batches of 256 images per iteration. We chose 15 epochs, which were sufficient to ensure that the loss function converged. The optimizer type also affects how CNN weights are updated after backpropagation is performed. The Adam optimizer was used to update the CNN weights because it has been shown to be effective for a wide range of machine learning methods such as logistic regression, deep fully-connected layers, and deep convolutional neural networks by combining the advantages of two of the most popular optimization methods: AdaGrad and RMSProp [32]. An adaptive learning rate of 10-4 and a weight decay of 0.001 were selected based on manual tuning.

3.3.4 Variance Estimates for Determining Uncertainty about the Point Estimates

Although regression models enable continuous outputs, regression models have disadvantages that classification models do not exhibit. Trained classification models output discrete state scores, which can be used for estimating probabilities of classifier outputs [33]. In contrast, trained regression models output point estimates that do not inherently provide any information regarding uncertainty about the point estimates. To calculate the uncertainty about a point estimate, the variance of the point estimate is first determined by observing the estimate error. The uncertainty is, in turn, used to establish prediction intervals that can be expressed as point estimate $\pm$ uncertainty or as (point estimate - uncertainty, point estimate + uncertainty) [34]. The width of the prediction interval will be referred to as the

angular resolution of the tactile directionality model.

The mean-variance estimate (MVE) method enables the calculation of a prediction interval by training a secondary NN to estimate the variance of a point estimate produced by a primary NN regression model [35, 36]. Our secondary NN had the same architecture and hyperparameters as the primary NN. The only difference was that an exponential activation function was applied at the output of the fully connected layer prior to the NN output in order to ensure that variance estimates would be positive. The primary NN regression model was trained prior to the secondary variance NN model. This was accomplished through three training phases as suggested in [35].

The secondary NN model was trained by minimizing a cost function on the mean-variance estimate.

$$C_{MVE} = \frac{1}{2}\sum_{i=1}^{n}\left[\ln(\hat{\sigma}_i^2) + \frac{(y_i - \hat{y}_i)^2}{\hat{\sigma}_i^2}\right] \tag{3.3}$$

where $\hat{\sigma}_i^2$ is the estimated variance of the $\hat{y}_i$ point estimate from the primary NN regression model. Unlike the point estimates from the primary regression model, the variance values cannot be labeled a priori for supervised learning. To estimate the variance for each point estimate, we used (3.3), which was derived from the log likelihood calculation and assumes that the point estimates are normally distributed around the true value.

Uncertainty was calculated using the estimated variance of the point estimate in a two-stage process. First, the estimated $\hat{\sigma}_i^2$ values from the secondary NN model and a user-assigned confidence level (1-α) were used to calculate the uncertainty for each of the two $\hat{y}_i$ point estimates (3.4).

$$\hat{y}_i \pm z_{1-0.5\alpha}\sqrt{\hat{\sigma}_i^2} \tag{3.4}$$

where $z_{1-0.5\alpha}$ was the critical value associated with a (1-α) confidence level. Specifically, (3.4)

produced upper and lower bounds on the estimated sin and cos values associated with the original perturbation angle point estimate. Second, the inverse tangent function was applied to the four combinations of prediction interval bounds for the normalized sin and cos values in order to produce four unique angles. The offset between the original tactile directionality point estimate and the angle furthest from the point estimate was conservatively reported as uncertainty about the point estimate in degrees.

3.3.5 5-fold Cross-validation of the Trained CNN Regression Model

The tactile directionality model developed in this work was benchmarked against the open-source force estimation model from Sundaralingam et al. [1]. The data used for our 5-fold cross-validation were fed through the open-source network in order to produce 3D force vector estimates. The 3D force vectors were then projected onto the planar contact surface of our experimental set-up. The projected force vector directions were used to calculate error with respect to the ground truth labels. Both models were compared in terms of average error according to perturbation angle (Figure 3.5a) and overall error rate (Figure 3.5b). As shown in Figure 3.5a, the open-source force estimation model performed well for perturbations in and around the proximal direction (180°), but resulted in large average errors for perturbations in the distal direction (0°) and adjacent quadrants. The direction-dependent average error of the open-source force estimation model led to a high overall error rate as compared to our proposed CNN model (Figure 3.5b). One metric of performance is the uncertainty about the point estimate, which enables the assessment of how well the CNN performs by observing the percentage of test samples that fall outside of a range rather than reporting a potentially misleading overall average error. Average error rates were calculated across all five folds based on three different levels of uncertainty (5°, 10°, 15°; Table 1). The

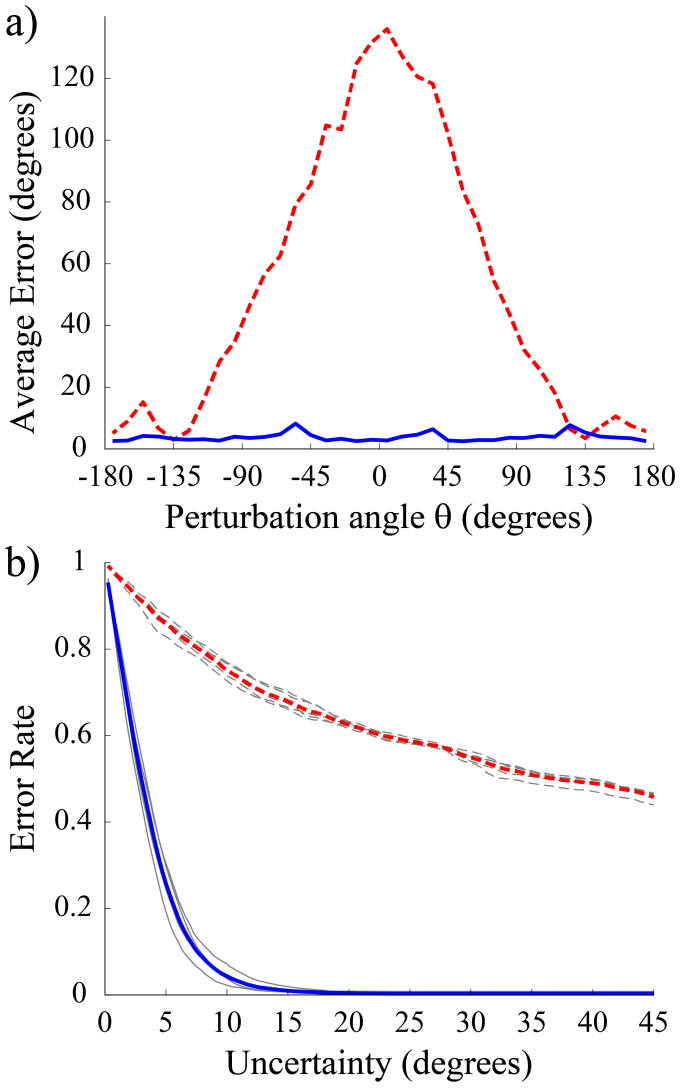

Figure 3.5: (a) Average error in estimated perturbation angle according to stimulus direction for the proposed CNN regression model (solid blue line) and the Sundaralingam et al. force estimation network [1] (dashed red line). (b) Error rates for both models assessed via 5-fold cross-validation. Thin and thick lines indicate the individual fold and average results, respectively.

		Uncertainty		
		5°	10°	15°
Fold	1	19.3%	2.2%	0.7%
	2	29.7%	3.9%	0.7%
	3	24.5%	4.6%	1.0%
	4	24.9%	3.8%	0.6%
	5	30.4%	7.2%	1.9%
Average Error Rate		**25.8%**	**4.3%**	**1.0%**

Table 3.1: Error rates based on a 5-fold cross-validation of the trained CNN regression model

		Confidence $(1-\alpha)\%$			
		80%	90%	95%	99%
Fold	1	6.2°	7.4°	8.3°	10.2°
	2	6.4°	7.5°	8.5°	10.4°
	3	7.5°	8.8°	9.8°	12.0°
	4	6.8°	7.9°	8.9°	10.7°
	5	7.0°	8.0°	8.9°	10.7°
Average Uncertainty		**6.8°**	**7.9°**	**8.9°**	**10.8°**

Table 3.2: Uncertainty about the point estimate based on a 5-fold cross-validation of the trained CNN variance model

performance of our CNN regression model is comparable to human perceptual thresholds for angular resolution of tactile directionality, which vary between $15-40°$ [37, 38, 39]. Depending on the level of uncertainty specified, the CNN estimation results varied. For an uncertainty of 7.5° (15° angular resolution), the error rate was 10.2%. For an uncertainty of

10° (20º angular resolution), the error rate dropped to 4.3% (Figure 3.5b, Table 3.1).

3.3.6 5-fold Cross-validation of the Trained CNN Variance Model

We evaluated the performance of our variance model by comparing the uncertainty associated with a specific error rate (Table 3.1, Figure 3.5b) with the estimated uncertainty associated with a specific confidence level (Table 3.2). Using 5-fold cross-validation, the average estimated uncertainty values were calculated using 80, 90, 95, and 99% confidence levels (Table 3.2). As expected, the higher the confidence level, the larger the uncertainty. An 80% confidence level resulted in an uncertainty of 6.8° or an angular resolution of 13.6º (Table 3.2). By comparison, a 20% error rate corresponded with an uncertainty of 5.7°, or an angular resolution of 11.4° (Figure 3.5b). A 90% confidence level resulted in an uncertainty of 7.9°, or an angular resolution of 15.8° (Table 3.2), while a 10% error rate corresponded with an uncertainty of 7.5º, or an angular resolution of 15.0° (Figure 3.5b).

When assessing the performance of the CNN regression model, the confidence level approach using the CNN variance model (Table 3.2) was always more conservative than the error rate approach (Figure 3.5b; Table 3.1). Ideally, angular resolution calculated for a 90% confidence level should be equivalent to that for a 10% error rate. In practice, the two-performance metrics are not equivalent, as the uncertainty estimation method assumes that the errors around the true value follow a normal distribution. Nonetheless, the results for both performance metrics are similar.

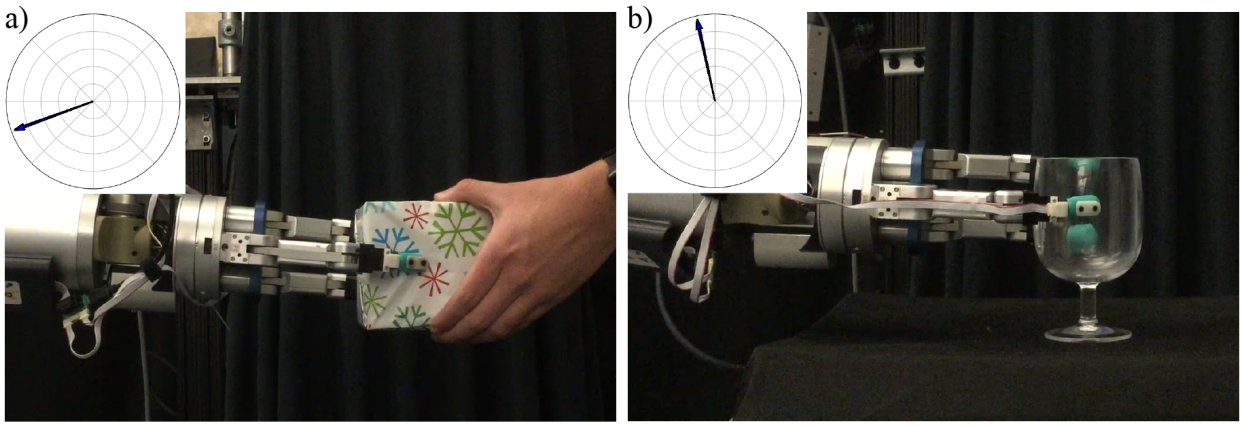

Figure 3.6: The CNN regression model was implemented in real-time as a number of external perturbations were imposed on a grasped object. The tactile directionality point estimates are shown in the inset figures. a) A humand agent pushed on a grasped box toward the robot's palm. b) The robot placed a plastic wine glass on a table.

3.3.7 Real-time Implementation

The trained CNN regression model was implemented in real-time on the robot testbed. The real-time BioTac data were fed through the trained CNN regression model using an Intel Core i7-5930K CPU, 16 GB RAM, and two GeForce GTX 970 GPUs via an SLI configuration. To highlight the real-time performance of the CNN regression model, we selected two scenarios for which tactile directionality could be easily verified: (i) perturbation of a grasped object caused by an external agent and (ii) placement of an object on a table. For the first scenario, an experimenter applied a number of perturbations to the grasped object (rectangular box). A push of the box towards the robot's palm is shown in Figure 3.6a. A real-time series of perturbations can be viewed in the supplementary video. In order to demonstrate a context-appropriate, or context-constrained response, the robot was programmed to release the grasped object only when the perturbation by the experimenter

was within ±15° of the distal direction of the fingertip ($\theta = 0°$, per Figure 3.2). That is, the robot only released the box when the experimenter pulled the box directly away from the robot's palm. CNN tactile directionality estimates were only considered for real-time visualization and context-appropriate behavioral responses if a user-defined threshold was met. This was done to prevent robot responses when no perturbations were occurring. The threshold was compared to the summation of electrode impedance derivative values across the entire fingerpad over the prior 100 ms period.

For the second scenario, an object placement task was performed to highlight how interactions between a handheld object (plastic wine glass) and the environment could be perceived (Figure 3.6b). With a stable grasp of the glass, the manipulator moved the glass downward towards a table. The robot was programmed to release the grasped object only when the perturbation by the environment was within ±15° of the radial direction of the fingertip ($\theta = 90°$, per 3.2). That is, the robot only released the glass when the presence of a supportive horizontal surface was perceived.

In terms of computational efficiency, the proposed CNN model performed faster than our previous work with Hidden Markov Models. The processing time for a single timestep of BioTac data through the CNN architecture was approximately 1.8 ms, which was shorter than the 10 ms wait time for each new BioTac sample. To compare with our prior work, Hidden Markov Models that incorporated temporal information required approximately 50 ms to process 250 ms worth of data before outputting its first tactile directionality estimate [7]. Additionally, the proposed CNN model offers improved angular resolution since it performs regression over the entire continuous range of perturbation directions ([0°, 360°]) rather than classifying perturbation angles using discrete 30° angular sectors.

3.4 Discussion

3.4.1 Application of CNNs for tactile directionality estimation

Our CNN regression model provides a continuous estimate of tactile directionality. Prior works, including our own, utilized classification in order to provide tactile directionality estimates with fixed angular resolution values. Abd et al. applied NNs to BioTac data in order to detect slip in four discrete directions, which yielded a 90° angular resolution [24]. Our prior work using Hidden Markov Models estimated tactile directionality using 12 discrete states, each having an angular resolution of 30° [7]. Kim and Park reported a similar directionality estimation capability (12 discrete states, 30° angular resolution) for their deformable three-axis load cell design [40]. In this work, we provide an overall error rate curve (Figure 3.5b) that enables the user to select the angular resolution appropriate for the target application.

CNNs have been shown to enable highly accurate estimates of force magnitude with deformable optical tactile sensors, such as the GelSight and TacTip. Yuen et al. applied CNNs to GelSight data in order to perceive 3D force and 1D torque during contact with objects of various curvatures, such as a planar surface, spheres, and cylinders [41]. The GelSight uses an embedded 30 Hz camera to record the deformation of the inner surface of an elastomer fingerpad. The input to the CNNs was the RGB difference between a current image and an initial non-contact image. Polic et al. used a convolutional autoencoder, a CNN designed to reduce the dimensionality of the input through unsupervised learning feature selection, on data from the TacTip optical sensor [42]. The TacTip uses an embedded camera to capture the movement of internal white-tipped pins against a black silicone skin. The input to the convolutional autoencoder was an interpolated grayscale image. Although

3D force magnitudes have been estimated using CNNs, neither Yuen et al. nor Polic et al. reported performance metrics related to force direction or tactile directionality.

Some works that estimate force magnitude using the BioTac also report the accuracy of force direction estimates. Su et al. uses NNs to determine the magnitude of a 3D force vector applied to the BioTac sensor [21]. A 3-layer NN is used with input features consisting of the BioTac's 19 electrode impedance signals. Similarly, Sundaralingam et al. developed a CNN for 3D force estimation using the BioTac sensor [1]. The Sundaralingam et al. work differs from the Su et. al. work by adding an estimated surface contact point to a 7-layer network input and formatting the input into voxels based on the electrode layout of the BioTac. While Su et al. did not cite force direction error directly, Sundaralingam et al. implemented the NN from Su et al. and reported the median angular error as 10° for the Su et al. method and as 3.5° for their own CNN method. For comparison, our network yielded a median angular error of 3.0. It should be noted that the force direction errors that are reported in [1] are for 3D force vectors and are, thus, the opening angle of a cone.

In contrast, the tactile directionality errors we report were calculated within the contact plane of a grasped object. In order to compare our model performance to the Sundaralingam et al. benchmark, we had to project their 3D force vector estimates into the contact plane. The benchmark model yielded an average error of 20° or less in the angle range of [125°, 235°], spanning approximately 110° in the proximal direction (Figure 3.5a). Tactile directionality error increased outside of the [125°, 235°] range, reaching a peak error of nearly 140° in the distal direction of the fingertip.

We suspect that the direction-dependent nature of the average errors for the Sundaralingam et al. force estimation model (Figure 3.5a) is due to an unintended bias in the training data for their network. Given that their model was aimed at estimating 3D force

vectors applied to the fingertip, it may be that experimenter-applied forces were more often directed proximally with respect to the BioTac's reference frame. Understandably, there may have been little motivation to collect training data for forces that were directed distally and/or near the narrow distal end of the BioTac fingertip. In contrast, our training data were collected specifically for the perception of tactile directionality and so we sought to thoroughly canvas all perturbation directions in the [0°, 360°] range in the 2D contact plane. Although our work conveniently estimates tactile directionality directly, one limitation of our work is that our CNN does not estimate a 3D force vector.

While CNNs can be applied to data from a variety of deformable sensors [41, 42, 43], there are benefits and limitations associated with different sensor designs. The data from optical tactile sensors naturally lend themselves as inputs to CNNs since the tactile signal is an image from a camera. Such sensors take full advantage of the CNN architecture, which was developed mainly for image classification within the field of computer vision [44]. However, optical tactile sensors are currently bulky and sampling rates can limit performance for real-time tactile applications. For example, the GelSight and TacTip sensors sample images at 30 Hz [41] and 20 Hz [45], respectively. In contrast, the electrode impedance signals of the BioTac sensor are sampled at 100 Hz, which enables faster tactile perception capabilities. Camera images from optical tactile sensors are also large, which can be advantageous for encoding visual details, but disadvantageous with regards to computational expense. The resolution of the GelSight and TacTip are 640 x 480 pixels and 300 x 300 pixels, respectively. Polic et al. downsample the TacTip data to 28 x 28 pixel images, incurring a slight increase in processing time dur to the downsampling process [42]. Comparatively, our tactile images generated from electrode impedance signals are constructed as 20 x 20 pixels and require less pre-processing than the native camera images.

3.4.2 Quantifying CNN Regression Model Uncertainty via Variance Estimates

While the CNN regression model performed well in estimating perturbation angle during the 5-fold cross-validation testing (Table 3.1), the model provided no information about uncertainty about the point estimates. Prediction intervals could be especially useful for tasks in which trust in the point estimate is critical for decision-making, as for the remote handling of high-consequence materials or safety-critical human-robot interactions.

Multiple methods have been developed to create prediction intervals for continuous NN outputs. A review paper by Khosravi et al. summarizes four different methods for estimating prediction intervals [36]. Out of the four methods surveyed, we selected the MVE method for our application of real-time perception of tactile directionality. The Delta and Bayesian methods were shown to be computationally expensive, reducing their appeal for real-time use. The MVE and bootstrap methods resulted in a lower computational demand through their use of NNs to predict the variance of a primary neural network. However, the bootstrap method requires multiple NNs while the MVE method only requires a single NN. The simplicity of the MVE method allows for fast calculation of a prediction interval, making it a strong candidate for real-time implementation.

A high-level decision-making algorithm could incorporate user-defined thresholds on both the point estimate and the uncertainty about the point estimate in order to execute context-appropriate responses. If one were to assume that every point estimate produced by the regression model were accurate, one would run the risk of enabling a robotic system to act on misinformation and respond inappropriately. The width of the prediction interval could be compared against a user-defined threshold to determine whether and how the real-time point estimate should be acted upon.

3.4.3 Model Robustness

This study focused on the perception of tactile directionality for a range of grip forces, perturbation angles, and perturbation speeds. The grip forces in our study varied between 5–10 N while human studies involved the application of normal forces to subjects' fingerpads that varied from 0.25–1 N [10, 27]. The variation in grip force magnitude had no discernible effects on model accuracy. We hypothesize that the fingerpad deformation and associated change in electrode impedance during a tangential stimulus do not change drastically over the 5-10 N force range. Lower grip forces were not explored, as grasp stability would have been degraded.

As part of a post-hoc study, we assessed the effects of perturbation speeds on the accuracy of tactile directionality estimates. We found that separating the data by perturbation speed had no meaningful impact on estimate accuracy. This result is consistent with a study with human subjects in which skin displacement rates faster than 2 mm/s had no impacts on tangential directional perception for skin displacement greater than 1 mm [10]. Our experiment employed tactile sensor skin displacements (20 mm) and perturbation speeds (minimum of 2 cm/s) in excess of those used in human subject experiments.

The real-time implementation of the CNN regression model highlighted the model's modest robustness. First, the grasp was changed from a parallel grip (Figure 3.1b) to a tripod grip (Figure 3.6). While we used a parallel grip during training in order to distribute the contact forces evenly between two opposing tactile sensors, a tripod grip was used for a more realistic real-time demonstration. Second, we substituted the object used in training (Figure 3.1) with two common objects (box and glass; Figure 3.6). The box had flat faces and parallel sides, similar to the training object. However, the width of the box was 4", compared to

4.75” for the training object. As a result, the fingertip contact angle for the demonstration with the box was slightly greater than that used during training (Figure 3.1c). The plastic wine glass had an outer diameter of 3.2”, which required a much smaller grip width than the box or training object. The glass also had a curved surface, for which the CNN regression model was never trained. Given the reliance of the tactile perception model on tactile sensor skin displacement, the robustness of the model to different object shapes, object widths, and fingertip contact angles was a pleasant surprise. These results highlight that the tactile directionality model can, to some extent, generalize to different handheld objects without requiring retraining.

3.5 Conclusion

In this work, we developed a CNN regression model to perceive tactile directionality associated with relative movements between a deformable fingerpad and a handheld object. We additionally developed a secondary CNN regression model to estimate the uncertainty about the tactile directionality point estimates. The tactile directionality model was implemented in real-time for the perception of forceful interactions between a handheld object and an external agent as well as the environment.

The perception of tactile directionality could be used to enhance situational awareness and initiate context-appropriate behavioral responses by robotic agents. Leveraging a tactile directionality model along with knowledge of a task, robots could more seamlessly manipulate objects, use tools, and interact with other agents and the environment. While this work focused on perception for action, such as object manipulation, one may also consider exploiting our model for action for perception, such as the exploration of the environment

(bare-fingered or with a handheld tool). Also, when using handheld tools to perform a task, the haptic perception of tool-environment interactions is essential [46, 47, 48, 49, 50, 51]. An interesting future direction would be the development of capabilities for a robot to distinguish between tactile stimuli induced by self-movement [52] as opposed to external agents or the environment.

From a practical controls perspective, the real-time perception of tactile directionality could be combined with the biological concept of an "efferent copy" [53] in order to verify that robot movements and physical interactions are proceeding as expected. In the future, the CNN variance model could be implemented in real-time and used to visualize uncertainty about a tactile directionality point estimate for robot teleoperation (as in the Figure 4 insets) or for decision-making by a semi-autonomous robotic system. With hand-crafted control algorithms, the robot could use its estimate of the state of a handheld object to respond in a context-appropriate manner.

CHAPTER 4

Controllers for Model-free, Tension-driven Manipulation of Deformable Linear Objects

4.1 Abstract

Deformable linear objects (DLOs), such as ropes and cables, are difficult to manipulate due to their infinite degrees-of-freedom. Prior works on DLO manipulation have focused on geometric or topological state through complex modeling and computer vision. In some tasks, the tensile state of the DLO needs to be controlled throughout the task. Tracking different gradations of tension cannot be achieved using vision once the DLO becomes taut. We present two novel controllers for use in model-free, tension-driven manipulation of DLOs: the force trajectory (FT) and the force-velocity trajectory (FVT) controllers. The controllers are designed for use on any robotic system that uses traditional motion control and is capable of measuring end-effector forces. The controllers could be used for tasks in which the tensile state of the DLO is more important than its configuration even if the elastic properties of the DLO are unknown. We demonstrate the controllers in real-time on a real robotic system for two utilitarian tasks: the circular wrapping of a DLO around a horizontal post and the figure-eight wrapping of a DLO around a boat cleat. We evaluate the performance of each controller and provide an open-source ROS package for implementation by the robotics

community.

4.2 Introduction

Manipulation of deformable linear objects (DLOs) is an essential skill needed in robotic systems. DLOs are manipulated in numerous applications spanning many fields such as tele-surgery, automotive assembly, and food processing [54]. State-of-the art techniques, such as computer vision [55], deep neural networks [56], and reinforcement learning [57] are used to learn policies for modeling and manipulating DLOs. However, these techniques are driven by the geometric state of the DLO rather than internal forces generated by physical interactions with the environment. Work has been limited to pick-and-place and knot-tying tasks, highlighting the difficulty of manipulating DLOs. Robotic manipulations that control both the DLO tension and movement while maintaining stability during interactions with an uncertain environment are required in many tasks.

This manuscript introduces two novel controllers for use in tension-driven manipulation tasks in 3D space: the force trajectory (FT) controller and the force-velocity trajectory (FVT) controller. Our controllers regulate the tensile state of the DLO using feedback from a wrist-mounted, 6-DOF force/torque transducer. The controllers prioritize control of DLO tension while following a reference motion rather than control of the exact DLO configuration.

The contributions of this manuscript include:

- Design of two controllers for manipulating DLOs in 3D space while controlling the tensile state and closely matching the desired end-effector motion.

- Implementation and evaluation of these controllers on real robot hardware without requiring explicit modeling, simulation, or a priori knowledge of the elastic properties of the DLO. Also, the controllers use robot joint position commands without needing to access the motor torques for force control.
- Open-source ROS package to facilitate implementation of the FT and FVT controllers by the robotics community.

The manuscript is organized as follows. Section II is a literature review of state-of-the-art techniques for DLO manipulation. Section III details the two novel controllers. Section IV describes the experimental evaluation of each controller on a real robotic system. Section V summarizes novel contributions, limitations of the work, and future work.

4.3 Related Work

Many works on the manipulation of DLOs focus on modeling DLO dynamics. Simulating the elasticity of a DLO is typically accomplished in one of two ways: adopt a simple mass-spring model or use the more complex Finite Element Method (FEM). Both techniques have been used to model a variety of deformable objects, including 1D ropes [58], 2D cloth sheets [59], and 3D shapes [60]. While mass-spring models have a low computational cost and enable the visualization of simulated DLOs in real-time [61, 62], the accuracy of the estimated internal forces is limited. FEM allows for a more accurate simulation of object deformation, which can be run in a real-time manner, such as in simulating a suturing procedure [63]. In both cases, the physical properties of a DLO, such as mass, damping coefficient, and elasticity, must be known or estimated in order to simulate the state of the DLO [64].

Other state-of-the-art approaches for modeling DLOs involve computer vision and machine learning techniques. Wang et al. used RGB-D images to track the geometric state of partially-occluded DLOs using convex geometric constraints [65]. Nair et. al. used Convolutional Neural Networks (CNNs) with human demonstration to develop a predictive model that relates robot actions to a DLO's geometric state [56]. Song et al. used CNNs to build a model for a DLO's topological state [66]. Han et al. used a sample-efficient, model-based reinforcement learning method to learn a policy for manipulating a DLO into a target state [57]. Such computer vision and machine learning techniques are effective at generating an inverse model and tracking the pose of a DLO in real-time. However, all aforementioned techniques focus on the geometric and/or topological state of the DLO as opposed to its internal state (e.g. tensile state) or forceful interactions between the DLO and the environment.

In some works, contact between the DLO and the environment is explicitly addressed and exploited during motion planning and the tuning of control parameters for manipulation. Zhu et. al developed a motion planning framework that purposely causes the DLO to contact the environment in order to reach a desired geometric state [55]. Bretl and McCarthy used the constraint of a DLO being held on both ends to suggest a motion-planning methodology for geometric equilibrium configurations [67]. When using indirect force control for the manipulation of DLOs, control parameters are often tuned in order to adapt to the deformable environment. Kautic et. al used a neural network to classify the deformable environment and to adjust the parameters of a compliance controller in simulation [68]. Lee et. al used impedance control with time-varying gains in order to achieve different levels of robot compliance at each stage of a knot-tying task [69]. The adaptive compliance was used only at the end of the task in order to tighten the knot. Tasks that require a precise DLO tension level would suffer from such an indirect tension control approach; for example,

breakage of delicate DLOs. In this paper, we present methods for direct tension control.

For DLO manipulation, force transducers provide information about the current force state of the DLO and enable model-free force control. Through a force transducer, Yue et. al used end-effector adjustment motions [70] while Ding et. al used sliding mode control [71] to suppress the vibration occurring at the free-end of a DLO without an explicit model. Hybrid position/force controllers that simultaneously regulate end-effector motion and contact force have been developed for surgical tools for skin harvesting [72] and to regulate the contact force of a tool tip during a cardiac ablation procedure [73]. However, these controllers require an accurate model of the medical tool and knowledge of the elasticity of the tissue.

This paper introduces novel, model-free controllers for use in tension-driven manipulation of DLOs. Our controller can generalize to different DLOs and environments without having to tune mass-spring model parameters or run computationally expensive FEMs in real-time. To the best of our knowledge, this is the first implementation of 3D force and motion control for tension-driven manipulation of deformable linear objects in a model-free manner. Manipulation of DLOs has been addressed using geometric or topological states for scenarios in which the DLO lies in a 2D plane. Previous work has been done to develop strategies for the manipulation of DLOs that interact with the environment, but there was no direct force feedback involved [55, 74]. Unlike the referenced works, our approach actively uses force feedback throughout the real-time manipulation of DLOs that interact with the environment.

4.4 Methods

In this section, we describe the design of two controllers that enable the tension-driven manipulation of DLOs. Our controllers are designed to determine an end-effector velocity

trajectory that enables the control of the tensile force state of a DLO and serves as the input to a traditional robot motion controller. The DLO controllers are aptly named based on the reference input needed to enact each controller: Force Trajectory (FT) and Force-Velocity Trajectory (FVT) controllers. A force transducer, as would be commonly mounted on a robot wrist, is required for the real-time measurement of 3D forces on the end-effector. The FT and FVT controllers can be used for tasks, such as reeling in or redirecting rope. We will demonstrate the controllers in the context of wrapping a DLO about another object. The wrapping task involves moving one end of a DLO with respect to an object while keeping the DLO taut, with a nonzero, internal tensile force state.

We observe that the tension-driven control of a DLO lies in the class of robot control under geometric constraints. The problem of moving a robot end-effector along a rigid surface while maintaining contact has been studied, resulting in the conclusion that force control is necessary [75]. Such force control generally takes the form of impedance control or hybrid force/position control.

The present problem of wrapping a DLO around an object differs from the classical contact problem in important ways. Specifically, the problem of wrapping involves different geometric constraints and representations and different motion requirements for completing tasks. The tension of the DLO cannot be observed visually, which makes grasp and manipulation planning difficult. Indirect force control methods are infeasible for the tension-driven manipulation of DLOs because robot pose alone cannot be used to estimate tensile forces when the nonlinear stiffness of the DLO is unknown.

4.4.1 Force Trajectory Control

The Force Trajectory (FT) controller determines the end-effector velocity required to maintain a desired tensile, 3D force of the DLO. Tension-driven manipulation of a DLO is accomplished by prescribing a 3D force vector reference trajectory as expressed by a desired force magnitude f^* and desired 3D force direction in unit vector form $\underline{\hat{f}}^*$. By design, the end-effector's motion is not explicitly prescribed. From this point forward, we will refer to the internal tensile force of the DLO as simply the force state of the DLO.

The FT controller decomposes the desired 3D force vector reference signal into three orthogonal components and simultaneously regulates the force magnitude along each orthogonal direction. The error between the desired and actual 3D force vector, as measured directly by a force transducer, drives the motion of the end-effector in order to reduce error in the 3D force vector trajectory.

Consider a scenario in which one end of a DLO is tethered to an object and a manipulator has grasped the free end of the DLO at a location that we refer to as the *grasp point* g (Fig. 4.1a). As g moves in 3D space, the fixed end of the DLO will pivot around the tether point. We will refer to the tether point as the *proximal contact point* p because it is the closest point to the end-effector, when tracing the DLO away from the end-effector, that is making contact with an object in the environment. In the example shown in Fig. 4.1a, p is fixed in space, but we will see in Sections 4.4.2 and 4.5.2 that $p(t)$ can be dynamic and is often unknown. We assume that the free portion of a DLO effectively has a ball joint degree-of-freedom at p. Here, we point out a minor advantage of our controls problem over the classical problem of moving an end-effector along a rigid surface. Specifically, when considering the manipulation of the free end of a DLO in an open air environment, the end-effector does not experience

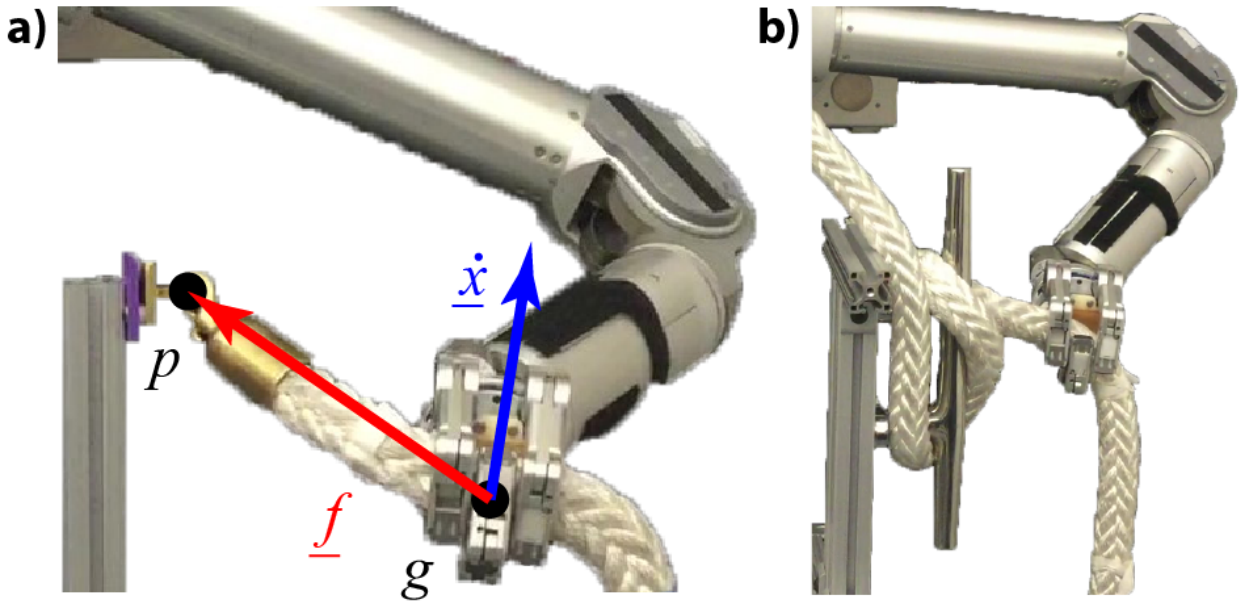

Figure 4.1: (a) A 7-DOF manipulator outfitted with a 6-DOF force/torque transducer is shown grasping a tethered DLO. The grasp point g, proximal contact point p, actual force vector $\underline{f}$, and actual velocity vector $\dot{\underline{x}}$ are shown. (b) Experimental set-up for a figure-eight wrap of a rope around a boat cleat.

friction.

In this work, we assume that the DLO is nearly inextensible, although this property is not strictly required. We refer to the length of the DLO between p and g as the *free length* $L(t)$. Assuming that the DLO is close to inextensible and that p is fixed in space, any movement of the end-effector just outside a sphere of radius L will cause the DLO to transition from a slack configuration to a taut configuration. In a taut configuration, the 3D force vector for the DLO will have a force magnitude f and unit vector direction $\underline{\hat{f}}$ that is coincident with the vector that points from an origin at the grasp point g to the proximal contact point p (Fig. 4.2a). For the scenario in which the proximal contact point p and free length L are known, the location of the end-effector can be used to determine $\underline{\hat{f}}$. In practice, a force transducer can be used to directly measure f and $\underline{\hat{f}}$ without knowledge of p or L.

Due to the decomposition of the reference 3D force vector into three orthogonal components, the FT controller is comprised of three PD controllers that influence the end-effector velocity output. As shown in Fig. 4.3, the FT controller precedes a traditional PD motion controller and utilizes a force transducer in its feedback loop.

In real-time, the force vector error $\underline{e}_f$ is calculated as

$$\underline{e}_f = \underline{f}^* - \underline{f} \tag{4.1}$$

where $\underline{f}^*$ is the desired force vector and $\underline{f}$ is the measured force vector. Since the FT controller is based on PD control, (4.1) is used to calculate the FT controller output $\underline{u}$

$$\underline{u} = K_P \underline{e}_f + K_D \dot{\underline{e}}_f \tag{4.2}$$

where $\dot{\underline{e}}_f$ is the rate of change in force vector error as calculated by differences between current and previous timesteps. The diagonal matrices K_P and K_D contain the proportional

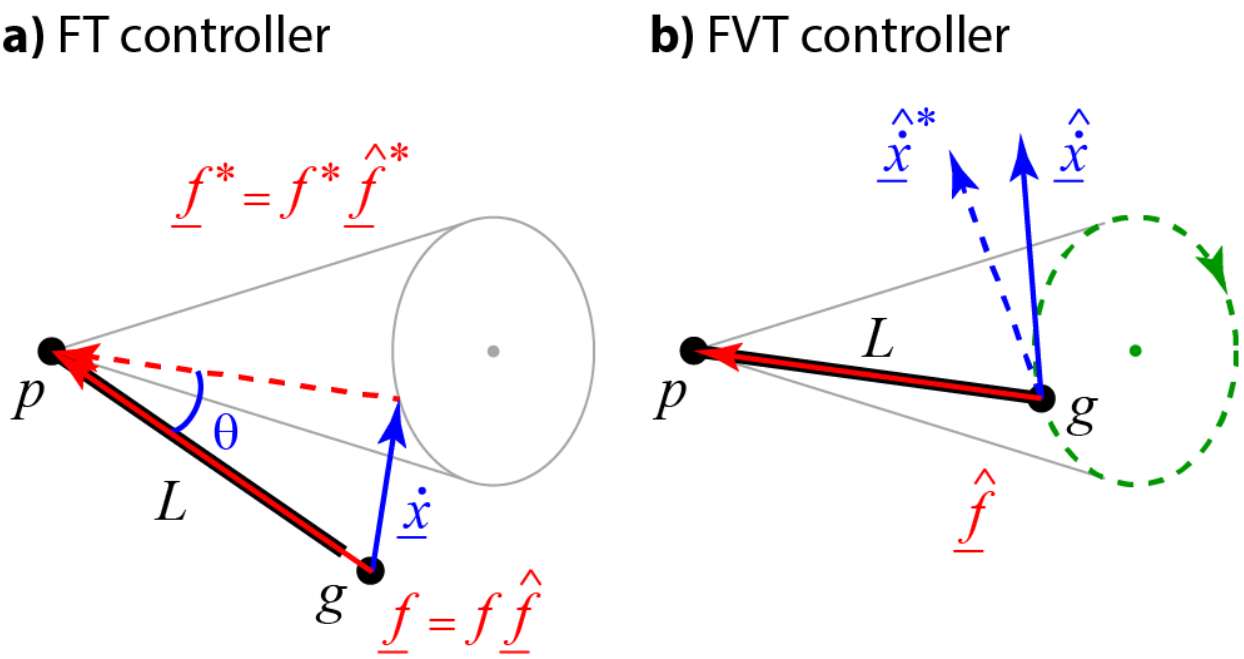

Figure 4.2: (a) The FT controller takes as input a reference 3D force vector $\underline{f}^*$. (b) The FVT controller takes as input a reference force magnitude f^* and feedforward velocity trajectory $\underline{\dot{x}}^*$. Note that underlined variables are vectors, $\hat{}$ indicates a unit vector, and * indicates a desired variable.

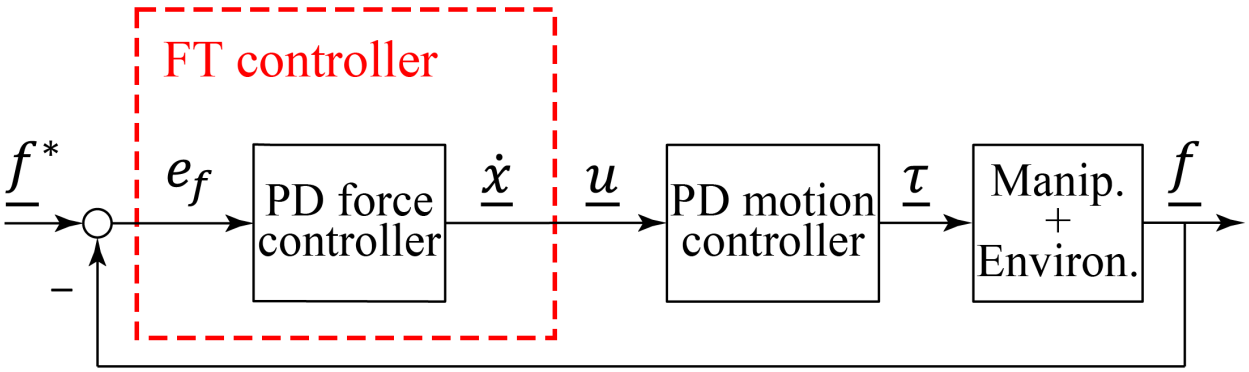

Figure 4.3: Block diagram of a complete robot control system with the FT controller highlighted by the dashed red line. The FT controller takes as input a reference 3D force vector trajectory $\underline{f}^*$ and produces a velocity command to be used as the input to a traditional motion controller.

and derivative gains, respectively, for the PD control of each orthogonal component of the force unit vector $\hat{\underline{u}}_f$.

The controller output in (4.2) is directly translated to end-effector velocity in operational space $\dot{\underline{x}}$.

$$\underline{u} = \dot{\underline{x}} = [v_x, v_y, v_z]^T \tag{4.3}$$

where v_x, v_y, and v_z comprise the orthogonal components of 3D end-effector linear velocity expressed in operational space. Equation (4.3) serves as the input to a traditional, operational space PD motion controller that follows the FT controller.

We refer the reader to Section 4.4.3 for details on the stability of the FT controller, which is based on PD control. We show that the closed loop feedback system will be asymptotically stable using PD gain tuning even when the nonlinear stiffness of a DLO is unknown.

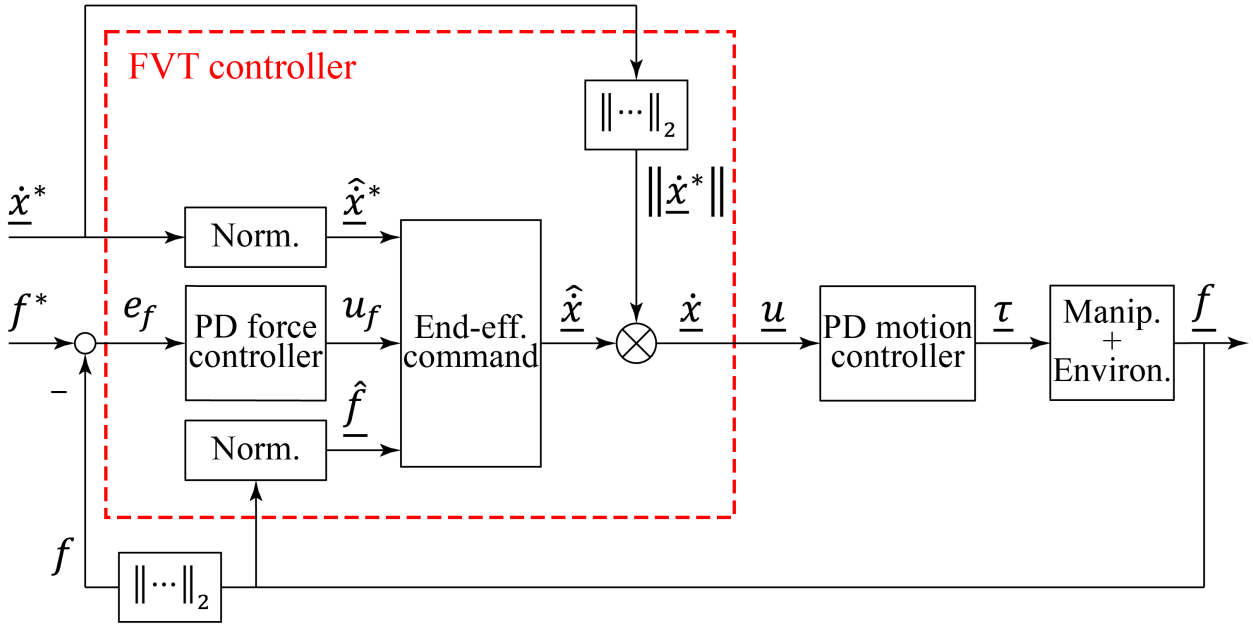

Figure 4.4: Block diagram of the feedback control system with the FVT controller highlighted by the dashed red line. The FVT controller takes as input a reference force magnitude f^* and feedforward velocity trajectory $\underline{\dot{x}}^*$ and produces a velocity command to be used as the input to a traditional motion controller.

4.4.2 Force-Velocity Trajectory Control

While the FT controller uses a 3D force vector trajectory as its reference input, the Force-Velocity Trajectory (FVT) controller takes as time-varying inputs a force magnitude f^* and a 3D velocity $\underline{\dot{x}}^*$ for the end-effector. The FVT controller enables the user to regulate the end-effector's motion in addition to the tension magnitude of the DLO. The FVT controller also has the advantage that the reference inputs are more intuitive for the user to design; end-effector motion is easier to prescribe than unit vector directions for a dynamic 3D force vector. As with the FT controller, the FVT controller precedes a traditional robot motion controller (Fig. 4.4).

The goal of the FVT controller is to maintain a desired tension of the DLO while following a desired end-effector motion. This is achieved by (i) moving the end-effector along the measured direction $\underline{\hat{f}}$ in order to regulate the tensile force of the DLO, and (ii) moving the end-effector along $\underline{\hat{x}}^*$, the unit vector direction of the desired end-effector velocity $\underline{\dot{x}}^*$. The FVT controller requires a nonzero tension, which can be achieved by increasing the distance between the proximal contact point and the DLO grasp point.

In the case that the reference force magnitude and 3D velocity of the end-effector cannot be simultaneously satisfied, the FVT controller is designed to prioritize the regulation of tension in the DLO over the strict tracking of the 3D velocity trajectory. This is useful when the tensile state of the DLO is critical for task completion and specific end-effector movements are not. In Section IV, we will demonstrate the effectiveness of the FVT controller when end-effector movement is driven by a human demonstration.

The FVT controller utilizes a single PD controller to regulate the magnitude of the tensile force of the DLO. The force magnitude error e_f is calculated as the L2 norm of the force vector error $\underline{e}_f$, as defined in (4.1).

$$e_f = \|\underline{e}_f\|_2 \tag{4.4}$$

The force magnitude error in (4.4) is used to determine the force magnitude controller output u_f based on PD control.

$$u_f = k_P e_f + k_D \dot{e}_f \tag{4.5}$$

where $\dot{e}_f$ is the rate of change in force magnitude error as calculated by differences between current and previous timesteps. The scalar k_P and k_D are the PD controller's proportional and derivative gains, respectively.

The controller output u_f from (4.5) drives the unit vector direction of the end-effector velocity $\hat{\underline{\dot{x}}}$ as follows.

$$\hat{\underline{\dot{x}}} = \frac{\hat{\underline{\dot{x}}}^* - u_f \hat{\underline{f}}}{\|\hat{\underline{\dot{x}}}^* - u_f \hat{\underline{f}}\|_2} \tag{4.6}$$

where $\hat{\underline{\dot{x}}}^*$ is the unit vector direction of the desired 3D velocity $\underline{\dot{x}}^*$ and $\hat{\underline{f}}$ is the unit vector direction of the measured 3D force $\underline{f}$. Equation (4.6) provides a unit vector direction for end-effector movement that is directly influenced by the measured force direction and desired end-effector velocity. Note that (4.6) prioritizes measured force direction due to the absence of a control gain for the desired velocity term.

The complete end-effector velocity command $\underline{u}$ is calculated for operational space as $\underline{\dot{x}}$ using the unit vector direction $\hat{\underline{\dot{x}}}$ from (4.6) and the L2 norm magnitude of the desired 3D velocity $\underline{\dot{x}}^*$ for the end-effector.

$$\underline{u} = \underline{\dot{x}} = \|\underline{\dot{x}}^*\|_2 \hat{\underline{\dot{x}}} \tag{4.7}$$

When the DLO is slack, rather than using (4.6), we specify the end-effector velocity as follows.

$$\hat{\underline{\dot{x}}} \approx -\hat{\underline{f}} \tag{4.8}$$

The approximation in (4.8) results from the large gain u_f produced in (4.5) when there is no tensile force in the slack DLO. Once the DLO becomes taut and the error in the force magnitude decreases, the desired velocity term begins to influence the end-effector motion (4.6). Effectively, the force and velocity direction terms compete with one another in order

to determine the end-effector motion, with priority given to the regulation of the tensile force of the DLO.

Specifying reference trajectories is a more intuitive process for the FVT controller than the FT controller. This is because the 3D force vector trajectory cannot always be simply defined for the FT controller, whereas 3D end-effector motion can be roughly approximated via computer simulation or human demonstration. Additionally, it is challenging to specify a 3D force vector trajectory when the position of the proximal contact point p is dynamic and unknown.

4.4.2.1 Generating a Reference 3D Velocity Trajectory via Human Demonstration

For tasks in which the motion of the DLO is complex, as in performing a figure-eight wrap around a boat cleat, it is difficult to specify a 3D force vector trajectory for an FT controller. Instead, we use human demonstration to generate a reference 3D velocity trajectory in operational space for an FVT controller and additionally specify a reference force magnitude.

As shown in Algorithm 1, synthesizing a feedforward velocity profile for the FVT controller involves smoothing a human-guided Cartesian path of the end-effector $\mathbf{X}$, mapping the path at the specified robot control frequency and end-effector velocity, differentiating the path into a motion profile, and scaling the profile temporally in order to generate a reference 3D velocity trajectory in operational space $\dot{\mathbf{X}}^*$.

Depending on the quality of the human demonstration, the nominal velocity profile may be insufficient for successful execution of the FVT controller, which prioritizes force magnitude over motion. Thus, we use a temporal scaling factor γ to modify the nominal velocity

Algorithm 1: Velocity Profile Generation

Input: Cartesian poses $\mathbf{X} = \{\underline{x}_0, ..., \underline{x}_N\}$, subsampling timestep t_{SUB}, robot control frequency f_R, moving average filter window size n, desired end-effector speed $\dot{x}^*$, temporal scaling factor γ

Output: $\dot{\mathbf{X}}^*$

1 Subsample $\mathbf{X}$ at every t_{SUB} to yield $\mathbf{X}_{SUB} = \left\{\underline{x}_0, \underline{x}_{t_{SUB}}, \underline{x}_{2*t_{SUB}}, ..., \underline{x}_T\right\}$

2 Smooth $\mathbf{X}_{SUB}$ with a moving average filter having a window size n to yield $\mathbf{X}_{FILT}$

3 Calculate the total time T needed to travel the total length of smoothed $\mathbf{X}_{FILT}$ at $\dot{x}^*$

4 Perform cubic spline of $\mathbf{X}_{FILT}$ onto time scale incremented at f_s up to T to yield $\mathbf{X}_{OUT}$

5 Differentiate and normalize $\mathbf{X}_{OUT}$ to yield $\hat{\dot{\mathbf{X}}}_o^*$

6 Multiply the total time T by γ to yield the scaled total time T_γ

7 Perform cubic spline of $\hat{\dot{\mathbf{X}}}_o^*$ onto time scale with total time T_γ to yield $\dot{\hat{\dot{\mathbf{X}}}}^*$

8 **return** $\dot{\mathbf{X}}^* = \dot{x}^* \hat{\dot{\mathbf{X}}}^*$

profile. A γ value greater than one serves to increase the total time that the manipulator has to perform the velocity trajectory. Subsequently, this expands the position trajectory in space, which allows the manipulator to make larger movements in order to achieve the desired force magnitude. From preliminary experiments, the user-defined γ value can lie in the range of one to four, but the actual value will depend on the robotic task and desired end-effector speed.

4.4.3 Closed loop Stability of the FT and FVT Controllers

We used a simplified model of an end-effector interacting with a DLO in order to facilitate the stability analysis of our proposed FT and FVT controllers. Both controllers operate in 3D Cartesian space. However, the FT controller uses PD control to simultaneously regulate each of three, orthogonal, scalar force magnitude components. Thus, we can reduce the dimensions of our stability analysis of the FT controller from a 3D force vector to a 1D force component. We make the same simplification for the analysis of the FVT controller, but for the reason that the FVT controller regulates the scalar force magnitude directly.

First, we consider the stability of the FT controller. The force feedback control loop is shown in Fig. 4.5 for a standard robotic system. Note that the stability analysis is performed in joint space in order to explicitly account for the properties of the driving motors. The FT controller, simply represented by compensator $C_f(s)$, is applied to a robotic system actuated by a driving motor under PI control $C_v(s)$. The mechanical properties of the driving motors are represented by motor torque constant K_m, motor inertia J, and viscous friction B. High gear reductions are assumed. Joint position and velocity are represented by $\underline{q}$ and $\underline{\dot{q}}$, respectively.

The force feedback control loop can be separated into a linear time-invariant (LTI) block $H(s)$ and a separate memoryless nonlinear block K that represents the DLO's unknown displacement-force relation in the Cartesian domain. This 3-by-3 K matrix includes the directional cosine transformation, which is defined by the vector formed by the proximal contact point p and grasp point g. The scalar DLO stiffness elements in K are bounded by 0 when in the slack state and k_{max} when in a taut state.

To simplify the system for a stability analysis, we further assume diagonal dominance,

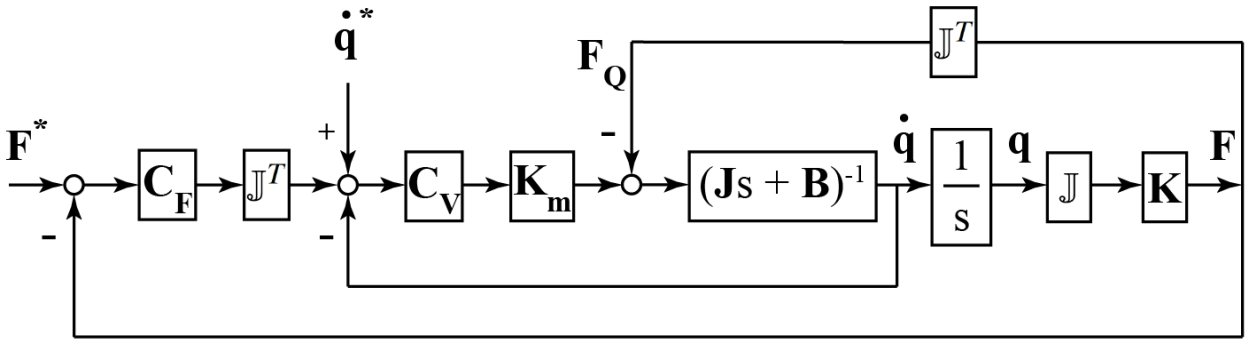

Figure 4.5: Force feedback control loop for a standard robotic system, represented with driving motors in joint space. Bold variables indicate vector and matrix quantities.

which is appropriate for industrial robots with high gear reductions, and commute the geometric Jacobian transformation $\mathbb{J}$ and C_F, which is appropriate for a slowly varying $\mathbb{J}$ and diagonal C_F. The nonlinear block, including the switching between the DLO's slack and taut states, the semi-positive directional cosine of the DLO tensile force, and the positive definite configuration-dependent transformation $\mathbb{J}\mathbb{J}^T$, may be considered an arbitrary static nonlinearity sector bounded between 0 and k_{max}:

$$F_Q = f(q) = k(q)q \tag{4.9}$$

where $0 \leq k < k_{max}$.

Thus, the absolute stability condition based on the circle criterion is

$$Re(H(j\omega)) > -\frac{1}{k_{max}} \tag{4.10}$$

where

$$H(s) = \frac{K_m C_v(s) C_f(s) + 1}{(Js + B) + K_m C_v(s)} \left(\frac{1}{s}\right) \tag{4.11}$$

Fig. 4.6 shows that an FT controller based on PD control always results in an absolutely

stable closed loop system while an FT controller based on PID control does not. For the Nyquist plot, we used motor properties based on the four proximal joints of the Barrett WAM. The estimated upper limit of DLO stiffness k_{max} was calculated based on the axial stiffness of a 3.2 cm braided nylon rope. The PID force controller gains were based on a pilot study while the PI velocity controller gains were set for stability of the closed loop motor velocity. Even when the stiffness K of the DLO is unknown, a set of proportional and derivative control gains can be identified for the FT controller that results in asymptotic stability of the closed loop system.

Gain tuning becomes especially important for ensuring closed loop stability if the compensator $C_f(s)$ is based on PID control since absolute stability is no longer guaranteed. Nonetheless, the integrator in $H(s)$ ensures a zero steady-state error for constant force commands if the closed loop system is stable, even if $C_f(s)$ is based on PD control.

Now, we consider the stability of the FVT controller. The representation of the FVT controller as $C_f(s)$ is similar to that for the FT controller, but is influenced by a feedforward joint velocity magnitude $\dot{q}^*$ (feedforward velocity input, but represented in joint space for this stability analysis). The feedforward velocity input is treated as a disturbance to the PD force controller $C_F(s)$. The introduction of the feedforward velocity disturbance does not affect the stability of the closed loop system, which implies that the FVT controller with embedded PD force control is also absolutely stable.

4.5 Experimental Procedure and Evaluation

We demonstrated our FT and FVT controllers for two utilitarian tasks: (i) wrapping a DLO around a horizontal post and (ii) performing a figure-eight wrap around a boat cleat.

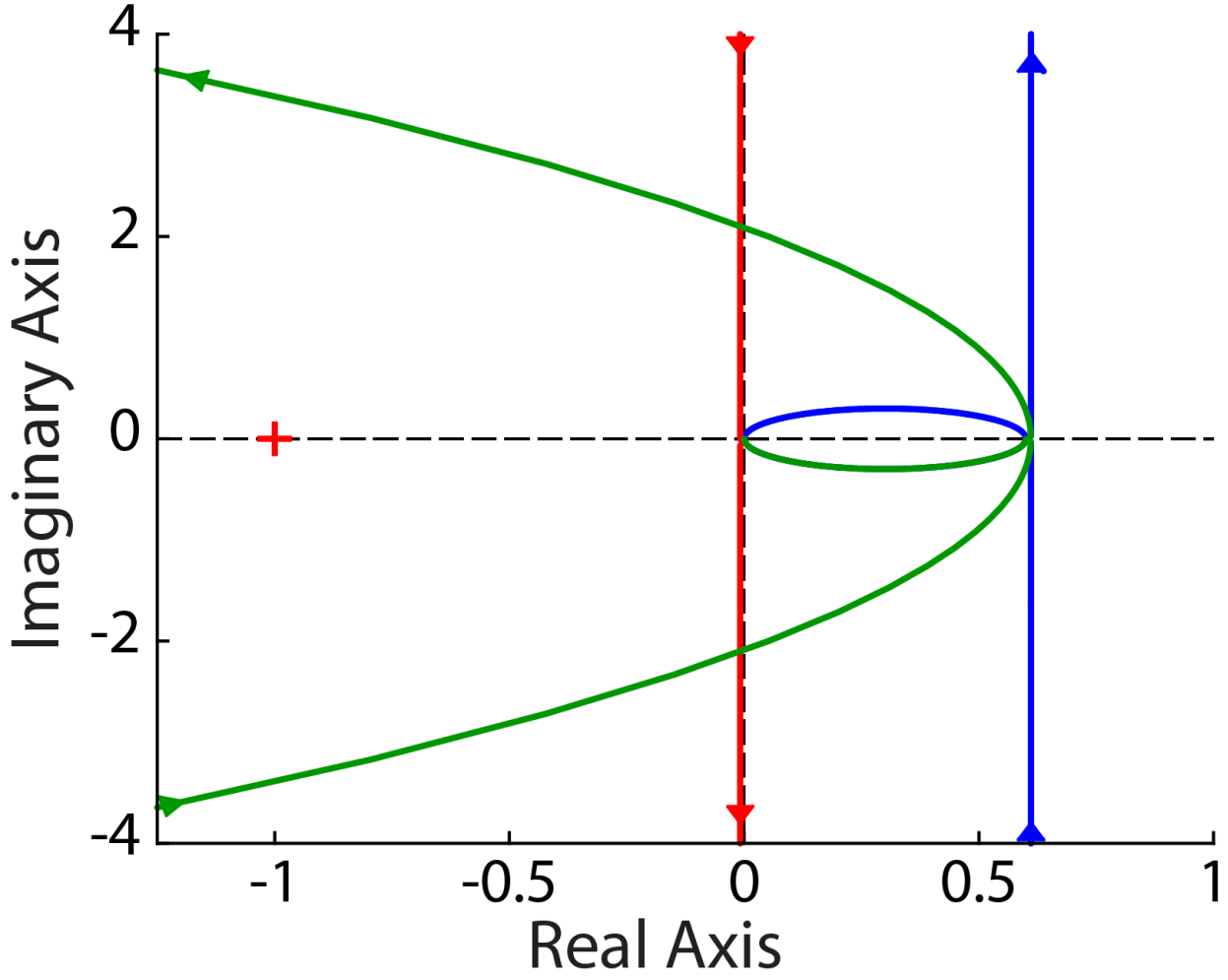

Figure 4.6: For the FT controller, Nyquist plots for the embedded PD (blue) and PID (green) controllers are shown alongside the non-linearity stiffness of the DLO (red). The embedded PD controller is absolutely stable while the embedded PID controller is not.

The post-wrapping task was used for the direct comparison of the FT and FVT controllers and to assess their performance for two different DLO types. The figure-eight task was used to show the versatility of the FVT controller for complex DLO manipulations that can leverage human demonstrations. For both tasks, the proximal contact point p of the DLO was both unknown and dynamic. Although a constant force magnitude was commanded throughout both tasks, the reference force magnitude could easily be defined as time-varying.

The robotic system was comprised of a 7-DOF manipulator, 4-DOF end-effector, and 6-DOF Force/Torque transducer (Barrett WAM, BarretHand; Barrett Technology, MA; Fig. 4.1). The manipulator was configured to mimic the orientation of a human arm with respect to gravity. A custom urethane rubber palm was added to the BarrettHand in order to improve form-closure during grasping. The FT sensor was calibrated using an established least-squares calibration procedure [76] and more than 100 different poses spanning the manipulator workspace. Our computer utilized the Xenomai real-time framework to ensure a communication rate f_s of 500 Hz.

The experimental setup consisted of a 3.8 cm square, rigid post that was cantilevered horizontally by a length of 10.2 cm from a fixed structure directly in front of the robot manipulator (Fig. 4.7). The experiment was performed with two different DLOs having different masses and diameters: a lightweight fishing line and a heavy rope. The monofilament nylon fishing line had a diameter of 0.33 mm and a mass per unit length of 0.08 g/m. The braided nylon rope had a diameter of 3.2 cm and a mass per unit length of 0.7 kg/m. Both DLOs were interweaved with the rigid post structure in order to provide an initial proximal contact point. As seen in Fig. 4.7, the fishing line was tied to a roll of tape for grasp and manipulation while the rope was grasped directly by the end-effector.

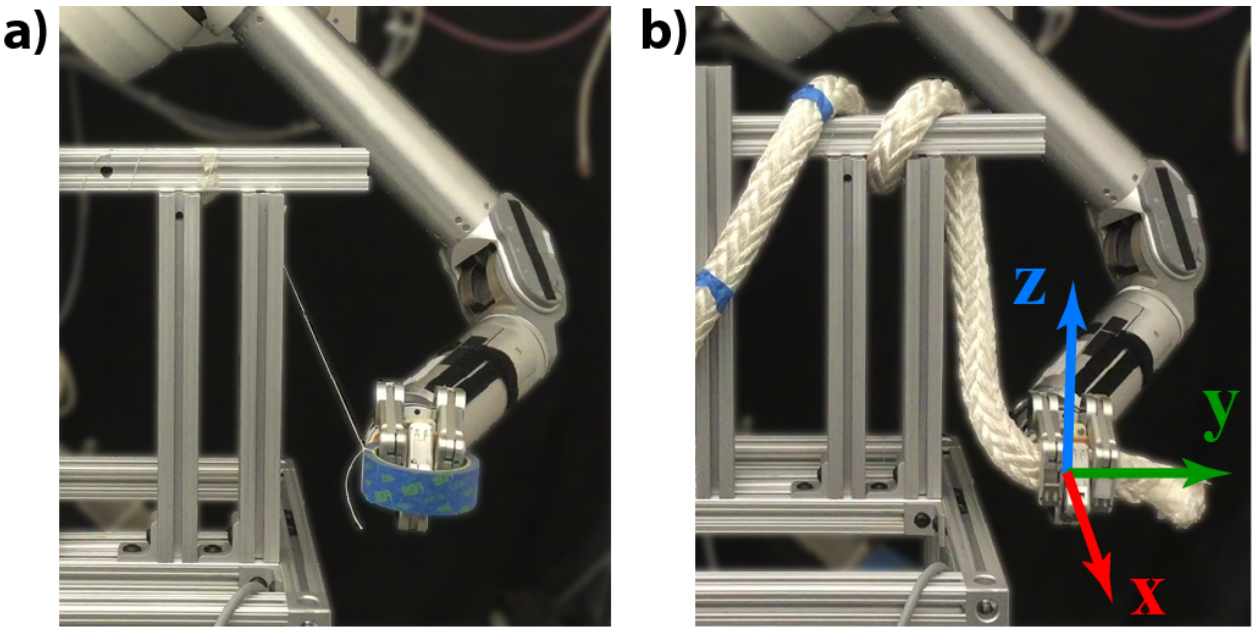

Figure 4.7: Experimental setup for the task of wrapping a DLO around a rigid, cantilevered post using (a) lightweight, monofilament nylon fishing line or a (b) heavy, braided nylon rope.

4.5.1 Circular Wrapping of DLOs Around a Horizontal Post

The first task was the wrapping of each DLO around a horizontal post while regulating tension throughout the motion. This task, trivial for humans, is difficult to implement on a robotic system using analytical or simulation-based methods without having an accurate model of the DLO. In addition, this task involves physical interactions between the DLO and another object, which is difficult to simulate. As a result, the robot controller must regulate tension throughout the manipulation of the DLO in a model-free manner using real-time force feedback alone.

For both controllers, a circular motion was used to create the reference trajectories for the wrapping task. To accomplish the circular motion, the initial configuration of the ma-

nipulator featured a bent elbow, which allowed the manipulator to perform a circular motion of its end-effector without colliding with the cantilevered post. The 3D circular motion was converted to a 3D force vector trajectory for the FT controller and was used directly as the 3D velocity vector trajectory for the FVT controller.

4.5.1.1 FT trajectory design

To obtain the 3D force vector reference trajectory for the FT controller (Fig. 4.2a), we considered a scenario in which the proximal contact point and free length of the DLO were constant. The grasp point of the DLO was moved along a circular path, and a constant tension was maintained. In that case, the DLO would be taut and would trace out the lateral, conical surface of a right cone. The 3D force vector reference trajectory would be comprised of 3D force vectors that lie along the the lateral surface.

Using spherical coordinates, the 3D force vector trajectory was defined using a desired force magnitude f^*, the initial DLO free length L_0 as the slant height of the cone, a desired opening angle α of the cone calculated from the radius of the circular motion of the grasp point g, the initial proximal contact point p as the vertex of the cone, and a desired angular velocity ω around the axis of the cone.

The vector set $\mathbf{S} = \{\underline{s}_1, \underline{s}_2, ..., \underline{s}_N\}$ was defined for a series of timesteps $t \in \{0, \Delta t, 2\Delta t, \cdots, T\}$, where N is the total number of timesteps, Δt is the robot control publishing time, and T is the total runtime respectively. The vector set establishing the cone's lateral surface in (4.12) was defined using the polar angle $\theta(t)$ and azimuth angle $\phi(t)$ in (4.13) along with the

cone parameters previously described.

$$\underline{s}(t) = \begin{bmatrix} L_0 \sin(\theta(t)) \\ L_0 \cos(\alpha) \\ L_0 \sin(\phi(t)) \end{bmatrix} \tag{4.12}$$

where

$$\begin{bmatrix} \theta(t) \\ \phi(t) \end{bmatrix} = \begin{bmatrix} \alpha \sin(\omega t) \\ -\alpha \cos(\omega t) \end{bmatrix} \tag{4.13}$$

Each individual vector from (4.12) comprising the vector set **S** was normalized to yield a unit vector that pointed away from the proximal contact point p.

$$\hat{\underline{s}} = \frac{\underline{s}}{\|\underline{s}\|_2} \tag{4.14}$$

Each unit vector from (4.14) was converted to $\hat{\underline{f}}^*$ by reversing the direction of $\hat{\underline{s}}$ such that each desired force vector would point away from an origin at g and towards p.

$$\hat{\underline{f}}^* = -\hat{\underline{s}} \tag{4.15}$$

The final 3D force vector reference trajectory f^* was determined as follows.

$$\underline{f}^* = f^* \hat{\underline{f}}^* \tag{4.16}$$

The 3D force vectors comprising the reference trajectory were decomposed into three orthogonal components according to a Cartesian coordinate system whose origin was coincident with the grasp point (Fig. 4.7b). Finally, each of the three reference trajectories of orthogonal force components were used as inputs to a traditional PD motion controller.

Table 4.1: For each controller and DLO, we report the mean±standard deviation for the error in force magnitude and offset angle of the 3D force vector, as well as the 95% rise time t_r for the force magnitude.

Controller	DLO	Error in force mag.* (N)	Offset angle of 3D force vector (°)	t_r for force mag. (s)
FT	Fishing line	0.09 ± 0.20	3.95 ± 2.25	0.36
	Rope	0.14 ± 0.14	3.36 ± 2.17	0.42
FVT	Fishing line	0.01 ± 0.30	-	0.62
	Rope	-0.22 ± 0.31	-	0.68

*The reference force magnitude was 12.5 N for the task of wrapping a DLO around a horizontal post.

4.5.1.2 FVT trajectory design

To obtain the 3D velocity vector reference trajectory for the FVT controller (Fig. 4.2b), we considered the same scenario as for the FT controller. However, the reference trajectory for the FVT controller was defined in the Cartesian velocity space. The FVT reference trajectory was based on the velocity profile for movement of the grasp point g along the circumference of the base of the right cone.

For a given sampling period Δt, a set of 3D velocity vectors was defined with a desired angular velocity ω. The end-effector velocity magnitude $\dot{x}^*$ was derived based on the user-defined ω and the circumference of the base of the cone.

$$\underline{\dot{x}}^*(t) = \dot{x}^* \begin{bmatrix} \cos(\omega t) \\ 0 \\ \sin(\omega t) \end{bmatrix} \tag{4.17}$$

where $t \in \{0, \Delta t, 2\Delta t, \cdots, T\}$.

For both the FT and FVT controllers, the reference force magnitude f^* was 12.5 N and the angular velocity ω around the axis of the cone was $8°/s$. We set Δt to 2 ms and T to 30 s. For safety purposes, a 20 cm/s limit was set for the maximum end-effector velocity. The reference end-effector velocity magnitude $\dot{x}^*$ was set to 4 cm/s. The PD gains for both controllers were tuned from initial values calculated according to the Ziegler-Nichols tuning method.

4.5.1.3 Performance of FT and FVT controllers

The FT controller tracked the desired force trajectory for all three orthogonal components of 3D force (defined in Fig. 4.7b) with low errors having a mean ± standard deviation of -0.03 ± 0.5 N and -0.1 ± 0.5 N in the y-direction for the fishing line and rope, respectively. For the x-component of force, the error was 0.2 ± 0.8 N and 0.1 ± 0.7 N for the fishing line and rope, respectively. Although the z-direction had the highest error of -0.3 ± 0.8 N and -0.3 ± 0.7 N for the fishing line and the rope, respectively, the FT controller was still able to regulate the overall force magnitude to within a 2% margin of the desired value. The low overall error in force magnitude for the FT controller is shown in Table 4.1.

One can also assess the performance of the FT controller by evaluating the angular offset of the 3D force vector from the reference direction. A dot product was used to report the angular offset between the actual and reference 3D force vectors (Fig. 4.8, blue). The FT controller regulated the direction of the 3D force vector to within $1.6° - 8.5°$ for the fishing line and to within $0.9° - 7.3°$ for the nylon rope.

A slow frequency oscillation was observed in the angular offset for both the lightweight fishing line and heavy nylon rope (Fig. 4.8). This phenomenon is attributed to the cyclic change in manipulability as the manipulator performed the commanded circular trajectory. To assess the effects of manipulator configuration on controller performance, a singular value decomposition was performed on the manipulator's geometric Jacobian. The inverse of the condition number of the Jacobian is commonly used to assess how close the manipulator is to approaching a singular configuration [77]. We overlaid the time history of the inverse of the condition number (red) onto that of the angular offset (blue) in Fig. 4.8. The slow oscillatory change in the inverse of the condition number corresponds with the slow

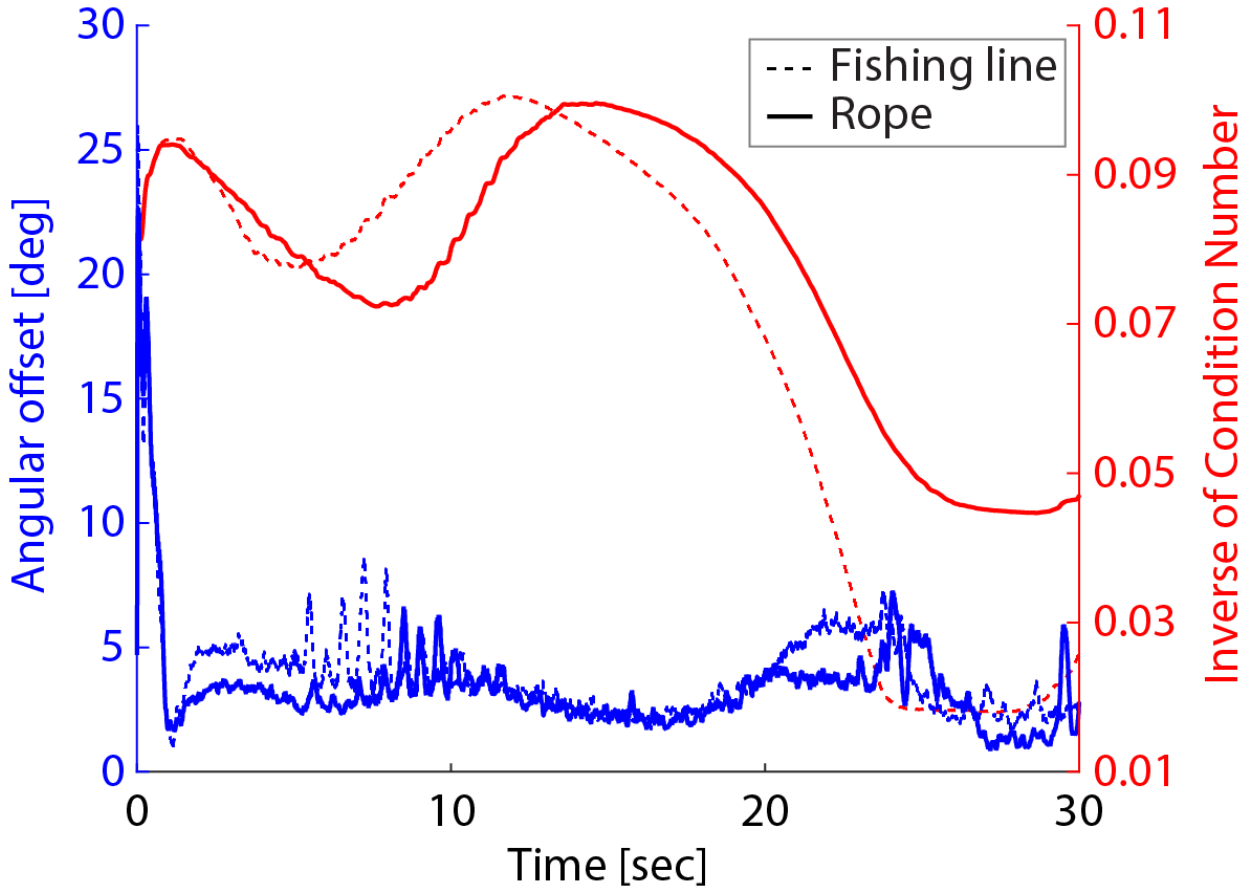

Figure 4.8: The angular offset of the the 3D force vector from the reference direction is shown in blue when using the FT controller for the task of wrapping fishing line (dotted) and rope (solid) around a horizontal post. The inverse of the condition number of the geometric Jacobian during the FT controller implementation is shown in red.

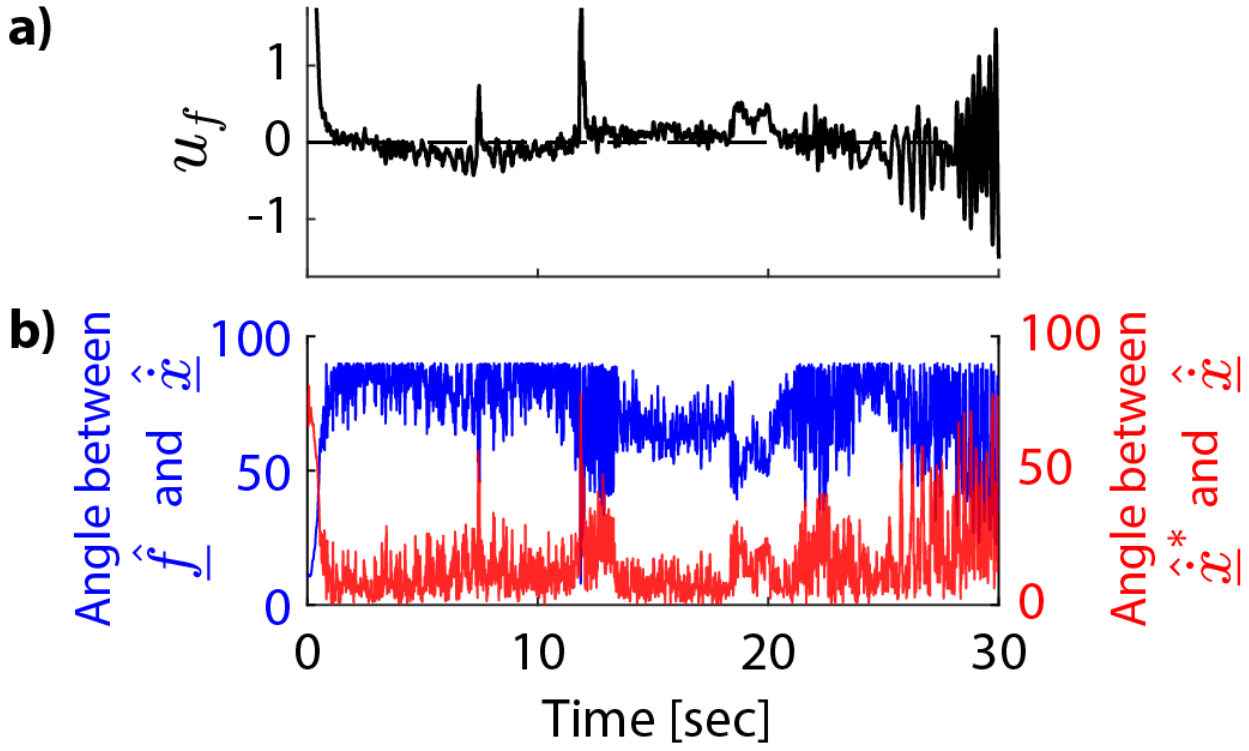

Figure 4.9: (a) Output of the embedded PD force controller u_f. (b) The angle between the actual force unit vector $\underline{\hat{f}}$ and commanded velocity unit vector $\underline{\hat{\dot{x}}}$ is shown in blue. The angle between the desired movement direction $\underline{\hat{\dot{x}}}^*$ and the commanded movement direction $\underline{\hat{\dot{x}}}$ is shown in red.

oscillatory change in angular offset. Decreases in the inverse of the condition number and manipulability correspond with increases in high frequency oscillations in the angular offset.

To assess the performance of the FVT controller, we analyzed the output of the embedded PD force controller. Results are shown for the manipulation of the monofilament fishing line (Fig. 4.9); trends are similar for the manipulation of the braided rope. When the error in force magnitude e_f was near zero, the PD force controller output u_f was near zero, as expected (Fig. 4.9a). The FVT controller prioritized tracking of the reference 3D velocity trajectory $\underline{\dot{x}}_d$ and moved the end-effector along the desired feedforward velocity direction. As

a result, the angle between the desired movement direction $\hat{\underline{x}}^*$ and commanded movement direction $\hat{\underline{x}}$ was near-0° (Fig. 4.9b, red). Also, the end-effector was commanded to move perpendicularly relative to the actual force vector, as shown by the near-90° angle between the actual force unit vector $\hat{\underline{f}}$ and commanded velocity unit vector $\hat{\underline{x}}$ (Fig. 4.9b, blue).

When the error in force magnitude increased, the PD force controller output u_f increased. The FVT controller then prioritized the correction of the force magnitude over the tracking of the reference 3D velocity trajectory. Accordingly, the angle between the force and velocity vector decreased away from 90°, and the angle between the desired and commanded movement directions increased away from 0°. Thus, the FVT controller prioritizes DLO force magnitude over the tracking of the reference 3D velocity trajectory, by design of the embedded PD force controller. This capability is useful for tasks in which DLO tension is more important than the precise tracking of a human-demonstrated movement.

One can also assess the performance of the FVT controller by evaluating the force magnitude throughout the wrapping task (Fig. 4.10). The FVT controller regulated force magnitude throughout the wrapping task with low errors having a mean ± standard deviation of 0.01 ± 0.30 N and -0.22 ± 0.31 N for the fishing line and rope, respectively (Table 4.1). Upon closer inspection of experimental videos, it was observed that the transient drops in force for the fishing line were caused by abrupt movements of the proximal contact point as the smooth fishing line slipped against the square, rigid horizontal post.

As with the angular offset for the FT controller (Fig. 4.8, blue), a slow frequency oscillation was observed in the force response for the FVT controller (Fig. 4.10, blue) that was related to fluctuations in the inverse of the condition number and manipulability. Oscillations in the force response increased in frequency and magnitude near the end of the task, when manipulability degraded (Fig. 4.10, red).

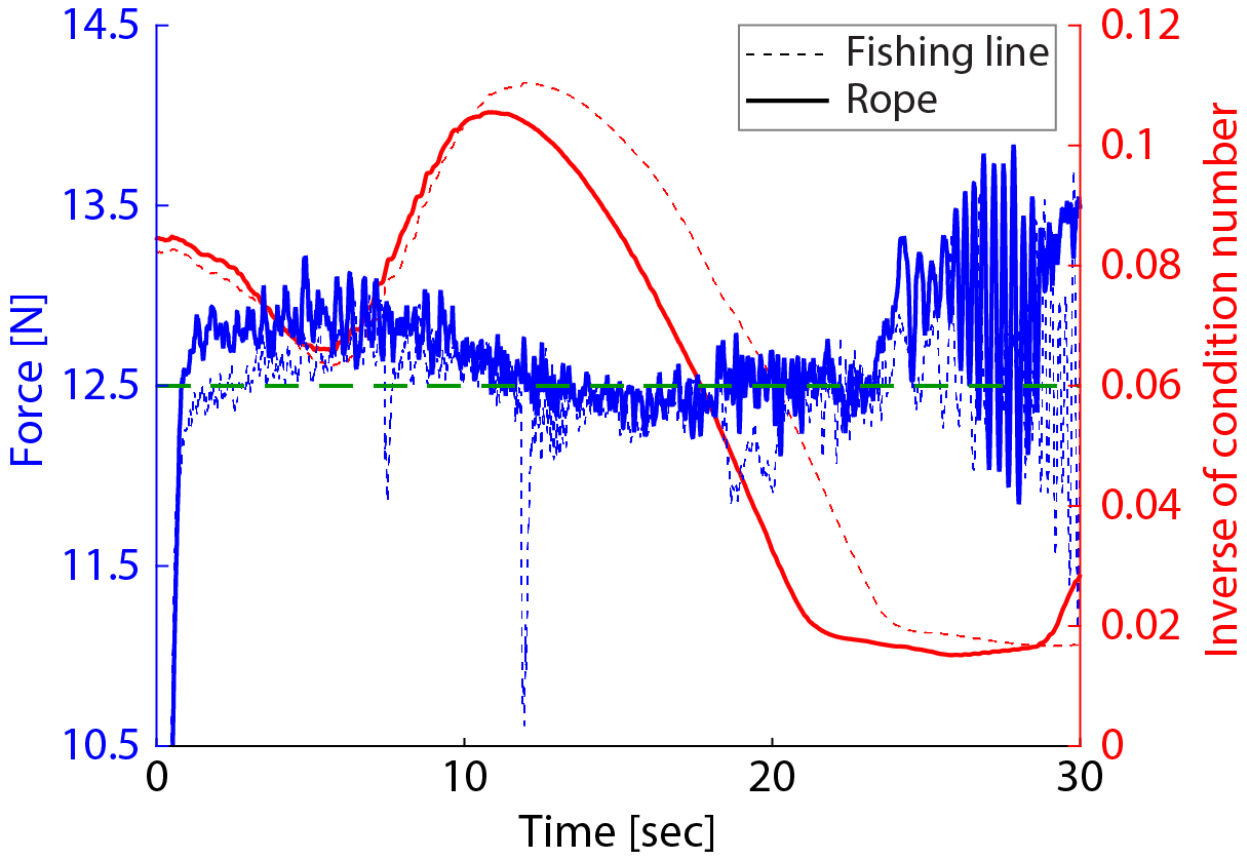

Figure 4.10: (a) The force response is shown in blue when using the FVT controller for the task of wrapping fishing line (dotted) and rope (solid) around a horizontal post. The desired force magnitude is shown by the green dashed line. (b) The inverse of the condition number of the geometric Jacobian during the FVT controller implementation is shown in red.

The performance of the FT and FVT controllers are directly compared in Fig. 4.11 and Table 4.1. The error in force magnitude averaged across both DLO types, and expressed as mean ± standard deviation, was 0.1 N ± 0.2 N for the FT controller and 0.1 N ± 0.3 N for the FVT controller. Both controllers regulated the DLO force magnitude well (mean error was $< 1\%$ of the reference force magnitude), although the force response produced by the FVT controller was slightly noisier than that produced by the FT controller (Table 4.1).

Fig. 4.11 shows the step response behavior for both controllers. The FT controller had a faster 95% rise time t_r than the FVT controller for both DLO types, although the rise times for all cases were less than 0.7 sec. The rise times were faster for the lightweight fishing line than the heavy rope (Fig. 4.11, Table 4.1). The rise times were 16.5% and 8.8% greater for the FT and FVT controllers, respectively, for the heavy rope as compared to those for the lightweight fishing line. This could be attributed to the lower mass and inertia of the fishing line as compared to the rope. Another contributing factor could be that the monofilament fishing line is stiffer than the braided rope. The Young's modulus of the monofilament nylon fishing line is 2.7 GPa [78]. For the rope, the Young's modulus was derived from the experimental axial stiffness value for fiber ropes [79]. The Young's modulus of the braided nylon rope was computed as 150.2 MPa based on the experimental axial stiffness and the cross-sectional area of the rope. While bending stiffness can affect the force response, the force component related to axial stiffness is likely to dominate that related to bending stiffness. Torsional stiffness was assumed to be zero as there was no deliberate twisting of either DLO during the wrapping task.

While Fig. 4.11 and Table 4.1 suggest that the FT controller is superior to the FVT controller in terms of performance, the FT controller is difficult to implement. The reference trajectory for the FT controller must be defined in 3D force space while the references for

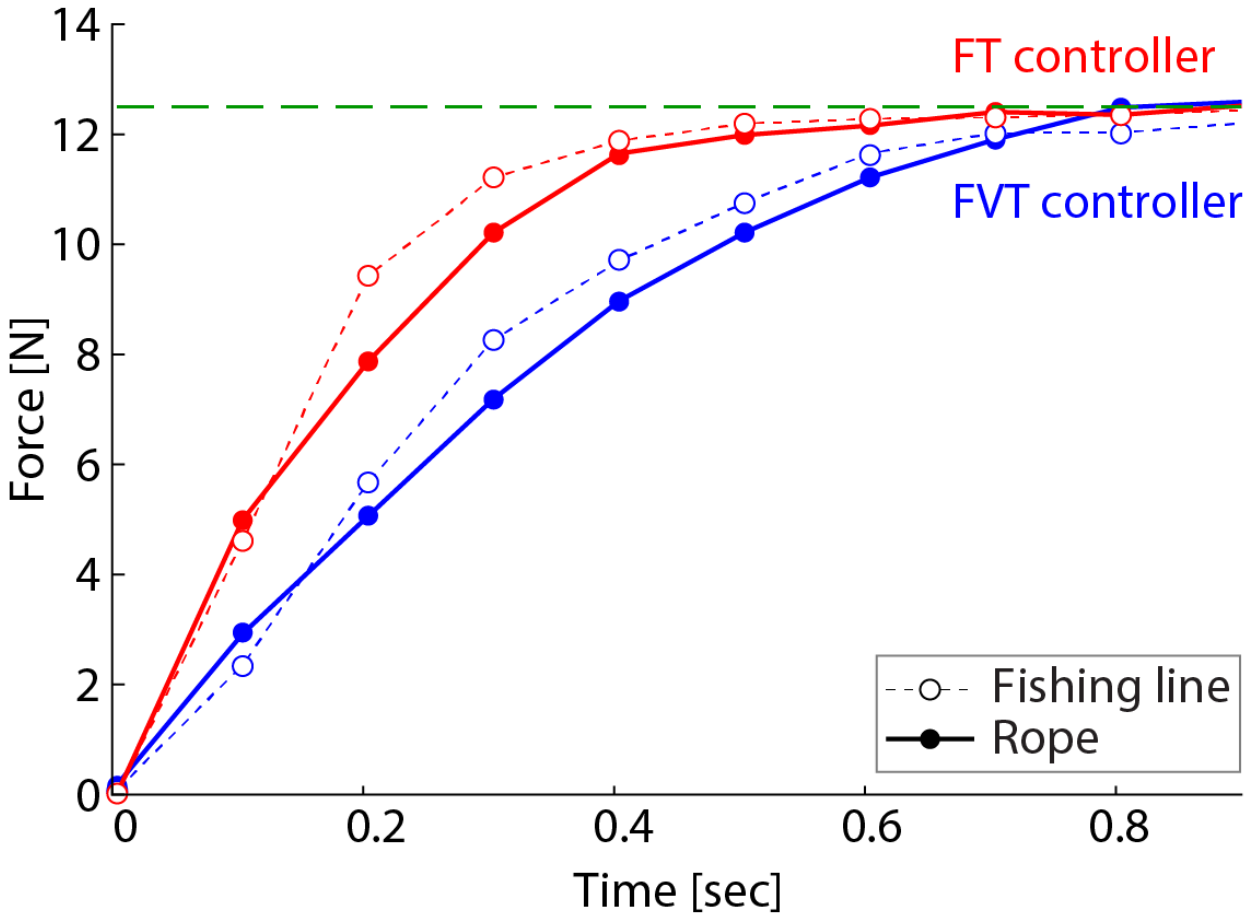

Figure 4.11: The step response for force is shown for the FT controller (red) and FVT controller (blue) for the task of wrapping fishing line (dotted line, open dots) and rope (solid line, solid dots) around a horizontal post. The desired force magnitude is shown by the green dashed line.

the FVT controller are comprised of an intuitive 3D Cartesian space motion trajectory and a force magnitude trajectory. As demonstrated in Section IV-B, the FVT controller has a distinct advantage over the FT controller for more complex tasks, such as performing a figure-eight motion when wrapping a DLO around a boat cleat.

4.5.2 Figure-eight Wrapping of Rope Around a Boat Cleat

Wrapping of DLOs using circular motions of the end-effector can be defined relatively easily in 3D force vector space. Although the length of the DLO decreases as the wrapping task progresses, the proximal contact point for the DLO does not change drastically. However, the proximal contact point can move in complex ways when a task requires multiple anchor points. For example, there are numerous anchor points to consider when routing bundles of DLOs through multiple clamp locations within an aircraft fuselage [80] or routing a single DLO through a cable harness [55].

When the proximal contact point moves in complex ways, it can be difficult to define reference 3D force vector trajectories for the FT controller. In such cases, the FVT controller has an advantage because one only needs to provide as references a force magnitude trajectory and 3D velocity vector trajectory. The 3D velocity vector trajectory can be created using a human demonstration. Simply put, it is easier and more intuitive to define a 3D velocity vector trajectory for the FVT controller than a 3D force vector trajectory for the FT controller for tasks that require complex manipulations of the DLO.

To demonstrate the effectiveness of the FVT controller, we performed a figure-eight wrap of a rope around a boat cleat. First a human demonstration was recorded. Algorithm 1 was then used to produce the smooth reference 3D velocity vector trajectory to be used in the feedforward portion of the FVT controller. Based on preliminary tests, the subsampling

timestep t_{SUB} was set to 0.5 s with a moving average filter window size n of 72.

Unlike the prior demonstrations of wrapping DLOs around a horizontal post, the boat cleat wrapping task was achieved using the multiphase process shown in Fig. 4.12. The robot began the task with a firm grasp of the rope, which dangled loosely below the bottom horn of the boat cleat. In Phase 1, the 1st stage of FVT control was activated, after exceeding a tension threshold of 9 N, in order to regulate tension as the rope was hooked around the bottom horn and then wrapped around the top horn of the boat cleat. The temporal scaling factor γ was set to 2.5 such that the feedforward velocity trajectory would be 2.5x the duration of the human-demonstrated motion trajectory. In Phase 2, the end-effector loosened its grasp on the DLO, slid down the DLO to gain a longer free length L of rope, and regrasped the rope. This step was necessary because the length of rope needed to complete the figure-eight movement was longer than the outstretched length of the manipulator. In Phase 3, the DLO was wrapped a second time around the bottom horn of the boat cleat. Since the FVT controller was not implemented in Phases 2 or 3, the temporal scaling factor was set to 1.0 for the feedforward velocity trajectories. In the final Phase 4, the 2nd stage of FVT control was activated in order to tighten the rope and complete the figure-eight wrapping task. The temporal scaling factor was set to 1.5 for this final feedforward velocity trajectory.

For the final demonstration of the figure-eight wrapping task shown in Supplemental Video 1, we used an end-effector speed of 16 cm/s, which yielded a good balance between overall time to task completion and performance of the FVT controller. Fig. 4.12 shows the force magnitude as a function of time throughout the multiphase task. All four phases of the task are shown in order to highlight the force regulation results during Phases 1 and 4, when the 1st and 2nd stages of the FVT controller were activated, respectively. The reference

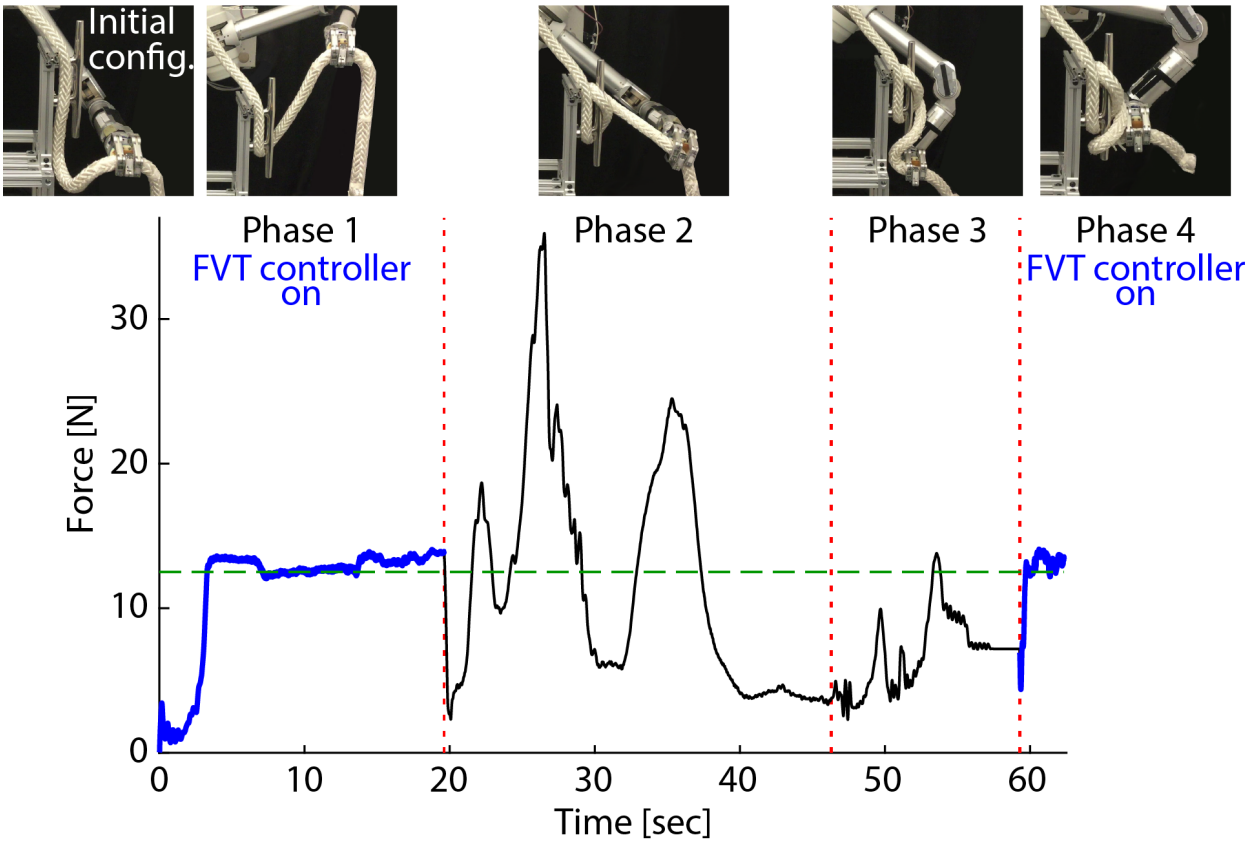

Figure 4.12: The force response is shown in blue for the FVT controller during the multiphase task of wrapping rope around a boat cleat. The FVT controller was activated during Phases 1 and 4 only. The desired force magnitude is shown by the green dashed line. Distinct phases are separated by red dotted lines.

force magnitude f^* was 12.5 N. The error in force magnitude, expressed as mean $\pm$ standard deviation, was -0.6 ± 0.5 N for the 1st stage of FVT control and 0.7 ± 0.5 N for the 2nd stage of FVT control.

For the FVT controller, the error in force magnitude was slightly larger for the task of wrapping the rope around a boat cleat (Fig. 4.12) as compared to wrapping the rope around a horizontal post (Fig. 4.10). However, the figure-eight wrapping task required a complex motion that was driven by a single, smoothed human demonstration. One could improve the performance of the FVT controller by using a slower end-effector speed, tuning the PD gains of the embedded force PD controller, and ensuring that the DLO is taut during the human demonstration. An FT controller might be capable of outperforming the FVT controller for such a complex manipulation task as figure-eight wrapping, but one would need a priori knowledge of the complex movement of the proximal contact point and a reference 3D force vector trajectory.

4.6 Conclusion

We developed and implemented model-free controllers that can be used to augment existing motion-based robot controllers for tasks involving the tension-driven manipulation of DLOs. Our controllers used PD force control to achieve closed loop stability despite the unknown stiffness properties of a DLO. On a real robot system, we demonstrated the FT and FVT controllers for two different wrapping tasks with DLOs. These controllers provide a foundation for developing more advanced tension-driven manipulation strategies for DLOs.

4.6.1 Summary of Contributions

This work introduces two novel controllers for the tension-driven manipulation of DLOs in 3D space: a force trajectory (FT) controller and a force-velocity trajectory (FVT) controller. Both controllers are model-free, making them easier to implement in scenarios where the properties of the DLO are unknown. Through the circle criterion, we show that our embedded PD force controller design can be tuned for stability without knowledge of the elastic properties of the DLO. Additionally, the feedforward velocity component of the FVT controller can leverage human demonstrations for complex manipulation tasks.

In contrast to prior works that control the geometric or topological state of the DLO using computer vision, our controllers prioritize the tensile state of the DLO using force transducers. The FT and FVT controllers were demonstrated for the circular wrapping of two types of DLOs (monofilament nylon fishing line, braided nylon rope) around a horizontal post. The FVT controller was demonstrated for a more complex task of wrapping of the braided rope around a boat cleat.

While the controllers were demonstrated on a single robotic system, the modular design of the controllers enable their implementation on any other robotic system that uses traditional motion control and is capable of measuring end-effector forces. We have published an open-source ROS package to facilitate implementation of the FT and FVT controllers on other robotic systems at https://github.com/BiomechatronicsLab/dlo-tension-control.

While the goals of FVT control are similar to those of hybrid force/motion control, it is important to differentiate the two approaches. Hybrid force/motion controllers regulate end-effector contact force and velocity by decoupling the two regimes and accounting for natural and artificial constraints. A hybrid force/motion controller tracks force by moving the end-

effector in directions constrained by contact and controls velocity by moving in unimpeded directions. Force and velocity are tracked in parallel using embedded PD and PI controllers, respectively. In contrast, FVT control uses a PD force control loop and feedforward velocity term to determine the end-effector velocity vector output of the controller. The FVT controller is placed in series with and upstream of a traditional PD motion controller.

With hybrid force/motion control, the satisfaction of both force and velocity setpoints is not guaranteed because the dynamic coupling between the manipulator and environment are not considered during implementation [81]. With FVT control, feedback control is used for regulating force magnitude only and the feedforward velocity trajectory is used as a guide. Perfect tracking of the velocity trajectory is not guaranteed, as force regulation takes precedence by design.

4.6.2 Limitations and Future Work

While the FT and FVT controllers are able to regulate tension during the manipulation of DLOs, there is room for improvement. For example, the controllers do not account for singular manipulator configurations. A manipulability analysis could be conducted a priori in order to constrain the controller outputs and avoid singular configurations.

The current robotic system does not account for obstacles in the environment that might impede the motion of the DLO or the manipulator itself. An a priori model of the environment and/or computer vision could be used to integrate obstacle avoidance capabilities. Furthermore, if computer vision could be used to track the 3D location of a dynamic proximal contact point, one could more easily specify the 3D force vector trajectory inputs to an FT controller.

Computer vision could also be used to track the geometric or topological state of the DLO in parallel with the tensile state provided by a force/torque transducer. Our demonstration of the wrapping of a rope around a boat cleat benefitted from a human demonstration. If a robot encountered a novel post or cleat in an unstructured field environment, computer vision would be critical for planning changes in the DLO's geometric state as well as its tensile state.

We present novel FT and FVT control structures that can serve as a foundation for additional fine-tuning depending on one's application. For example, one could incorporate weighting terms into the FVT controller structure to encode the importance of tracking force magnitude relative to following a velocity trajectory. Reinforcement learning could also be used to autonomously tune controller gains while considering disturbances, sensor noise, and uncertainty in DLO characteristics [82]. By enabling the direct control of the tensile state of a DLO, our FT and FVT controllers expand the capabilities for robotic systems to perform complex manipulation tasks with deformable linear objects.

CHAPTER 5

Summary and Conclusion

The work presented in this dissertation has provided new methodologies for object manipulation through artificial tactile sensing and proprioception. While computer vision remains the most common modality for sensory feedback in robotics, the work in this paper addresses gaps in object manipulation research that can only be executed with a sense of touch. Through the use of machine learning, we utilized tactile data to perceive tactile directionality during the grasp of a handheld object. Novel controllers were developed to enable tension-driven control during deformable linear object manipulation through the use of a force transducer. These methodologies can be used to improve object manipulation.

5.1 Contributions

Development of perceptual capabilities for tactile directionality: Using the skin deformation features of a multimodal tactile sensor, we can perceive the direction of perturbations being applied to a handheld object. Through both discrete (HMM) and continuous (CNN) models, we can estimate a tactile state for the finger-object interaction that includes directional context for the perturbation. In addition, we introduce the use of confidence intervals through our CNN implementation as a measure of uncertainty. The uncertainty estimate can be used by the robot or a remote teleoperator to make decisions based on

confidence in the point estimate and the risk associated with the manipulation task.

Real-time implementation of tactile directionality perception on a real robot: A CNN regression model is implemented in real-time on robot hardware for a pick-and-place task and for unpredictable perturbations by an external human agent. We show that the perception model works with objects having different shapes and sizes than the object used for model training. Perception of tactile directionality can be used in decision-making algorithms to enable robotic systems to make context-appropriate actions.

Novel controllers for use in model-free, tension-driven manipulation of deformable linear objects: To tackle the challenge of manipulating dynamic and complex deformable linear objects having unknown physical properties, we developed two novel controllers to enable tension-driven manipulation in a model-free manner. The FT controller regulates the direction and magnitude of the measured DLO tension through a defined trajectory of 3D force vectors. The FVT controller regulates the magnitude of tension through a desired force magnitude and feedforward 3D motion profile. These controllers have been published online in the form of an open-source ROS package for use by the robotics community: https://github.com/BiomechatronicsLab/dlo-tension-control.

Implementation of novel controllers on a real robot: We show that two novel controllers for model-free, tension-driven manipulation can be implemented in real-time on robot hardware. In particular, we implement these controllers for two utilitarian tasks: the circular wrapping of a DLO around a horizontal post and the figure-eight wrapping of a rope around a boat cleat. Both controllers are able to control the tensile state of the DLO, as desired.

5.2 Future Work

5.2.1 Improve generalizability and training efficiency for tactile directionality perception

The work presented in Chapters 2 and 3 provides methodologies for perceiving tactile directionality of external perturbations being applied to an object held in parallel grasp. As seen in Figure 3.1, the tactile sensor was fixed at a single contact angle and area of contact on the deformable fingerpad. This covers only a small portion of possible grasps when it comes to object manipulation. The ability to perceive tactile directionality regardless of the contact location on the fingertip would be especially useful.

However, to enable this level of generalizability, a vast amount of training data is needed for multiple contact points. Unlike data from images and videos that are relatively simple to collect, tactile data from physical interactions are limited due to the need for experiments on real robot systems. Such physical experiments can be difficult to conduct and time-consuming, and can induce wear on the hardware.

One solution is to develop accurate models of tactile sensors that can be used to simulate innumerable finger-object contacts and the associated tactile sensor readings. Unfortunately, the creation of such accurate models of tactile sensors and contact mechanics has remained a long-standing problem in the field of robotics. Recently, a research group has developed a finite element, neural network-based model of a multimodal, deformable tactile sensor [83]. Another solution is to develop a generative model that can be used to simulate tactile images for lateral perturbations against the tactile sensor. This can significantly reduce the amount of real tactile sensor data needed to build a more generalizable tactile directionality perception network.

5.2.2 Complement the approach to tension-driven manipulation of DLOs with additional sensors

Tactile sensors can be added to the fingers and palm of the end-effector to improve the implementation of tension-driven control. The distribution of forces on sensorized fingertips can be monitored to estimate the direction of the taut DLO. A sensorized palm in combination with the fingers can be used to estimate the DLO grip quality, enabling active loosening and tightening of the grasp throughout the task. With known finger configurations, a 3D tension vector can be estimated without a wrist force transducer. While finger configuration was fixed in Chapter 4, tactile sensors could be used to regrasp a DLO in a manner that distributes the tension reaction forces across all digits. Localized tactile sensing on the end-effector can enable new methods of estimating and regulating DLO tension.

Computer vision could be used to complement the controllers in Chapter 4 for the tension-driven manipulation of DLOs. While computer vision is initially used to locate and grasp the DLO, computer vision could also be used during the regulation of the tensile state of the DLO to provide additional information about the geometric state of the DLO. With visual feedback, one might also be able to track the dynamic movement of the proximal contact point and the grasp point along the length of the DLO. With the proximal contact point and grasp point tracked in real-time, the system can actively monitor the free length of the DLO or if DLO starts sliding from the desired wrapping location.

Obstacle avoidance can also be automated through the incorporation of computer vision. Large arm movements are often needed during the tension-driven manipulation of DLOs. This creates a large volume in which the manipulator or the DLO might undesirably collide with the environment. With visual feedback, the location and size of obstacles can be

provided to the robotic system. Through programs like MoveIt!, an alternative joint path can be found through inverse kinematics that would prevent the manipulator or the DLO from colliding with an obstacle.

5.2.3 Automate gain and parameter tuning for the tension-driven manipulation of DLOs

Chapter 4 presents novel controllers for manipulating deformable linear objects. While these controllers are able to regulate the tensile state of the DLO, the operator must provide a reference motion and manually tune the temporal scale of the feedforward motion profile. Both the FT and FVT controllers also require gain tuning of the embedded PD controllers.

The temporal scale and PD gain tuning can be optimized through the use of gain scheduling, for example. The temporal scale and gains can be changed dynamically based on information such as force error or remaining rope length. Many gain scheduling techniques exist such as fuzzy gain scheduling, which varies the values based on an operating range and a defined set of fuzzy logic rules [84]. The temporal scale of the feedforward motion profile can be modified according to the distance between the end-effector and the nominal human demonstration trajectory. The gains of the embedded PD controller can be modified based on error in the tension magnitude.

APPENDIX A

Hidden Markov Models

HMMs are used as a black-box density model to estimate the likelihood of a given time-series sequence. Through the use of discrete-time and non-observable discrete-states, HMMs use Markov chains to determine the joint distribution for a time-series of data $\mathbf{x}_{1:T}$ and its hidden latent states $\mathbf{z}_{1:T}$ where T is the number of total number of timesteps. Through use of these joint distributions, the observation probability $p(\mathbf{x}_{1:T})$ is estimated and used to optimize the HMM parameters θ.

For continuous non-discrete observations, the observation model probability $p(\mathbf{x}_t|z_t)$, where t represents one timestep, is defined using a conditional Gaussian.

$$p(\mathbf{x}_t|z_t = k, \theta) = \mathcal{N}(\mathbf{x}_t|\mu_k, \Sigma_k) \tag{A.1}$$

Using the observation models calculated at each time-step, the joint distribution $p(\mathbf{x}_{1:T}, \mathbf{z}_{1:T})$ can be determined for the time-series.

$$p(\mathbf{x}_{1:T}, \mathbf{z}_{1:T}) = p(\mathbf{z}_{1:T})p(\mathbf{x}_{1:T}|\mathbf{z}_{1:T}) = \left[p(z_1)\prod_{t=2}^{T} p(z_t|z_{t-1})\right]\left[\prod_{t=1}^{T} p(\mathbf{x}_t|z_t)\right] \tag{A.2}$$

where $p(z_1)$ is derived from initial state distribution π and $p(z_t|z_{t-1})$ can be derived from state transition matrix A. $p(\mathbf{x}_{1:T}, \mathbf{z}_{1:T})$ can then be used to determine the probability of the sequence $p(\mathbf{x}_{1:T})$ by summing the all the joint distributions among all the possible hidden

state sequences.

$$p(\mathbf{x}_{1:T}) = \sum_{z_{1:T}} p(\mathbf{x}_{1:T}, \mathbf{z}_{1:T}) \tag{A.3}$$

(A.3) is used to optimize the parameters for the HMM.

Optimizing the parameters for each HMM is accomplished through a supervised learning process. Each model is composed of parameters $\theta = (\pi, A, B)$ where π is the initial state distribution containing $p(z_1)$, A is the state transition matrix containing values for $p(z_t|z_{t-1})$, and B contains the distributions for the emission probabilities $p(\mathbf{x}_t|z_t)$ as defined in (A.1). The number of hidden states K are defined by the user either through optimization techniques, such as cross-validation or maximizing a objective function. The goal of the training is to optimize the parameters in such a manner that maximizes the Maximum Likelihood Estimate (MLE)

$$\ell(\mathbf{x}_{1:T}|\theta) = \log p(\mathbf{x}_{1:T}|\theta) = \log \sum_{z_{1:T}} p(\mathbf{x}_{1:T}, \mathbf{z}_{1:T}|\theta) \tag{A.4}$$

where $\ell(\mathbf{x}_{1:T}|\theta)$ is the log of the sum of the observation probabilities over all the latent states.

The process of optimizing these parameters is known as the Baum-Welch or the Expectation-Maximization (EM) algorithm, a gradient-based method for maximizing the MLE through parameter optimization [85], summarized in Table A. To start the learning, the probabilities in π and A are randomized. Emission probabilities in B is initialized through the conditional Gaussian in (A.1) with the μ and Σ statistics computed based on a randomized sampling of the training data along all hidden states. The algorithm is composed of two steps: E-step (Expectation), and M-step (Maximization). The E-step calculates the expected log-likelihood of the data based on the current θ. The M-step maximizes the log-likelihood from the E-step by modifying parameters π, A, and B based on the expected statistics.

The E-step computes the expected log-likelihood value based on the current θ and suf-

Algorithm 2: Baum-Welch Parameter Optimization

Input: Initial State Transition Matrix $\mathbf{A}_0$, Initial Emission Probabilities $\mathbf{B}_0$, Initial State Distribution $\underline{\pi}$, Time Sequence Dataset $\mathbf{X} = \{\underline{x}_0, ..., \underline{x}_N\}$

Output: $\mathbf{\Theta} = \{\mathbf{A}, \mathbf{B}, \underline{\pi}\}$

1 **for** n = 0 to N **do**

2 Calculate iteratively forward variable α_t for all hidden states (A.6)

3 Calculate observation probability through summation of α_t for all hidden states (A.7)

4 Calculate backwards variable β_t for each hidden state through a backwards iterative process (A.9)

5 Calculate γ at each each time step for all hidden states (A.11

6 Calculate ξ at each time-step for all possible hidden state transitions (A.13)

7 Update parameters $\underline{\pi}$ (A.14), $\mathbf{A}$ (A.15), $\mathbf{B}$ (A.16-A.17)

8 **end for**

9 **return** $\Theta = \{\mathbf{A}, \mathbf{B}, \underline{\pi}\}$

ficient statistics. The forwards-backwards algorithm is used to compute these sufficient statistics through a two-step process. The forward part of this algorithm involves defining a forward variable α_t

$$\alpha_t(i) = p(\mathbf{x}_{1:t}, z_i|\theta),\ 1 \leq i \leq K \tag{A.5}$$

which represents the joint probability of a partial observation sequence at time t and being in the hidden state z_i with the given model parameters θ. α_1 is initialized through current model parameters π and B. Through an iterative process, α_t is calculated at each time-step in the observation sequence $\mathbf{x}_{1:T}$.

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{K} \alpha_t(i)a_{ij}\right] b_j(x_{t+1}),\ 1 \leq t \leq T-1,\ 1 \leq j \leq K; \tag{A.6}$$

where a_{ij} is a component of A and b_j is the emission probability of hidden state z_j. After the alpha terms are computed for each hidden state, they are summed to provide the probability of observation sequence $\mathbf{x}_{1:T}$.

$$p(x_{1:T}|\theta) = \sum_{i=1}^{K} \alpha_T(i); \tag{A.7}$$

which also can be converted into a log-probability. In a similar iterative process, a backwards variable $\beta_t(i)$ is defined as the probability of the partial observation from $t+1$ to the end T given the hidden state z_t and the current model parameters.

$$\beta_t(i) = p(\mathbf{x}_{t+1:T}|z_t, \theta),\ 1 \leq i \leq K \tag{A.8}$$

β_T is initialized at 1 for all the hidden states. The iteration process is performed in reverse to determine all the backwards β_t values.

$$\beta_t(i) = \sum_{j=1}^{K} a_{ij}b(\mathbf{x}_{t+1})\beta_{t+1}(j),\ 1 \leq j \leq K \tag{A.9}$$

Using the statistics calculated in the E-step, the M-step maximizes the probability expectation by incriminating model parameters in θ. Two new variables $\gamma_t(i)$ and $\xi_t(i,j)$ are

defined to optimize the parameters π, A, and B. $\gamma_t(j)$ is defined as the probability of being in state z_j at time-step t based on the observation sequence and current model parameters.

$$\gamma_t(i) = p(z_i|\mathbf{x}_{1:T}, \theta) \tag{A.10}$$

Using (A.5) and (A.8), γ_t can be calculated at each time-step for each hidden state z_i.

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{p(\mathbf{x}_{1:T}|\theta)} = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{K} \alpha_t(i)\beta_t(i)} \tag{A.11}$$

$\xi_t(i,j)$ is defined as the probability of being in state z_i at time t and being in state z_j at time $t+1$ given the observation sequence and the current parameters.

$$\xi_t(i,j) = p(z_t = z_i, z_{t+1} = z_j|\mathbf{x}_{1:T}, \theta) \tag{A.12}$$

ξ can also be computed using the model parameters as well as the forward and backward variables.

$$\xi_t(i,j) = \frac{\alpha_t(i)a_{i,j}b_j(x_{t+1})\beta_{t+1}(j)}{p(\mathbf{x}_{1:T}|\theta)} = \frac{\alpha_t(i)a_{i,j}b_j(x_{t+1})\beta_{t+1})(j)}{\sum_{i=1}^{K}\sum_{j=1}^{K} \alpha_t(i)a_{i,j}b_j(x_{t+1})\beta_{t+1}(j)} \tag{A.13}$$

With all the γ and ξ terms computed for all time-steps and hiddens states, the model parameters can be updated. The initial probability distribution π is updated using the computed γ.

$$\bar{\pi}_i = \gamma_1(i) \tag{A.14}$$

where $\bar{\pi}_i$ is the initial probability for state z_i. The state transition matrix is updated using both γ and ξ.

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \tag{A.15}$$

The mean and variance of Gaussian emission probability B is also updated for each observation type using γ.

$$\bar{\mu}_{\mathbf{i}} = \frac{\sum_{t=1}^{T} \gamma_t(i) \cdot \mathbf{x}_{1:t}}{\sum_{t=1}^{T} \gamma_t(i)} \tag{A.16}$$

$$\bar{\mathbf{\Sigma}}_{\mathbf{i}} = \frac{\sum_{t=1}^{T} \gamma_t(i) \cdot (\mathbf{x}_{1:t} - \mu_{\mathbf{i}})(\mathbf{x}_{1:t} - \mu_{\mathbf{i}})^T}{\sum_{t=1}^{T} \gamma_t(i)} \tag{A.17}$$

REFERENCES

[1] B. Sundaralingam, A. S. Lambert, A. Handa, B. Boots, T. Hermans, S. Birchfield, N. Ratliff, and D. Fox, "Robust Learning of Tactile Force Estimation through Robot Interaction," in *2019 International Conference on Robotics and Automation (ICRA)*, May 2019, pp. 9035–9042, iSSN: 2577-087X.

[2] W. Wang and K. Siau, "Artificial Intelligence, Machine Learning, Automation, Robotics, Future of Work and Future of Humanity: A Review and Research Agenda," *Journal of Database Management*, vol. 30, no. 1, pp. 61–79, Jan. 2019. [Online]. Available: http://services.igi-global.com/resolvedoi/resolve.aspx?doi=10.4018/JDM.2019010104

[3] G. Du, K. Wang, S. Lian, and K. Zhao, "Vision-based Robotic Grasping From Object Localization, Object Pose Estimation to Grasp Estimation for Parallel Grippers: A Review," *Artificial Intelligence Review*, vol. 54, no. 3, pp. 1677–1734, Mar. 2021, arXiv: 1905.06658. [Online]. Available: http://arxiv.org/abs/1905.06658

[4] O. Kroemer, S. Niekum, and G. Konidaris, "A Review of Robot Learning for Manipulation: Challenges, Representations, and Algorithms," *arXiv:1907.03146 [cs]*, Nov. 2020, arXiv: 1907.03146. [Online]. Available: http://arxiv.org/abs/1907.03146

[5] A. Billard and D. Kragic, "Trends and challenges in robot manipulation," *Science*, vol. 364, no. 6446, p. eaat8414, Jun. 2019. [Online]. Available: https://www.sciencemag.org/lookup/doi/10.1126/science.aat8414

[6] R. Dahiya, G. Metta, M. Valle, and G. Sandini, "Tactile Sensing – From Humans to Humanoids," *IEEE Transactions on Robotics*, vol. PP, no. 99, pp. 20, 1, Nov. 2010. [Online]. Available: http://dx.doi.org/10.1109/TRO.2009.2033627

[7] K. Gutierrez and V. J. Santos, "Online perception of tactile directionality using Hidden Markov Models," in *Proc. Robotics: Science and Systems Workshop on "Tactile Sensing for Manipulation: Hardware, Modeling, and Learning"*, Boston, MA, Jul. 2017.

[8] K. Gutierrez and V. Santos, "Perception of Tactile Directionality via Artificial Fingerpad Deformation and Convolutional Neural Networks," *IEEE Transactions on Haptics*, vol. 13, no. 4, pp. 831–839, Oct. 2020. [Online]. Available: https://ieeexplore.ieee.org/document/9007491/

[9] K. P. Murphy, *Machine learning: a probabilistic perspective*. MIT press, 2012.

[10] B. T. Gleeson, S. K. Horschel, and W. R. Provancher, "Perception of Direction for Applied Tangential Skin Displacement: Effects of Speed, Displacement, and Repetition," *IEEE Transactions on Haptics*, vol. 3, no. 3, pp. 177–188, Jul. 2010.

[11] L. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[12] R. S. Johansson and J. R. Flanagan, "Coding and Use of Tactile Signals from the Fingertips in Object Manipulation Tasks," *Nature Reviews Neuroscience*, 2009.

[13] I. Birznieks, *Tactile Sensory Control of Dexterous Manipulation in Humans*. Umeå University, 2003.

[14] L. J. Berryman, J. M. Yau, and S. S. Hsiao, "Representation of Object Size in the Somatosensory System," *Journal of Neurophysiology*, vol. 96, no. 1, pp. 27–39, Jul. 2006. [Online]. Available: http://jn.physiology.org/content/96/1/27

[15] D. Burke, S. C. Gandevia, and G. Macefield, "Responses to passive movement of receptors in joint, skin and muscle of the human hand." *The Journal of Physiology*, vol. 402, no. 1, pp. 347–361, Aug. 1988. [Online]. Available: http://doi.wiley.com/10.1113/jphysiol.1988.sp017208

[16] V. G. Macefield, C. Hager-Ross, and R. S. Johansson, "Control of grip force during restraint of an object held between finger and thumb: responses of cutaneous afferents from the digits," *Experimental Brain Research*, vol. 108, no. 1, pp. 155–171, 1996.

[17] C. Hager-Ross and R. S. Johansson, "Nondigital afferent input in reactive control of fingertip forces during precision grip," *Exp Brain Res*, vol. 110, no. 1, pp. 131–141, 1996.

[18] V. G. Macefield and R. S. Johansson, "Control of grip force during restraint of an object held between finger and thumb: responses of muscle and joint afferents from the digits," *Experimental Brain Research*, vol. 108, no. 1, pp. 172–184, Feb. 1996. [Online]. Available: http://www.ncbi.nlm.nih.gov/pubmed/8721165

[19] A. Saxena, J. Driemeyer, and A. Y. Ng, "Robotic Grasping of Novel Objects using Vision," *International Journal of Robotics Research*, vol. 27, no. 2, pp. 157–173, Feb. 2008.

[20] J. M. Romano, K. Hsiao, G. Niemeyer, S. Chitta, and K. J. Kuchenbecker, "Human-Inspired Robotic Grasp Control With Tactile Sensing," *IEEE Transactions on Robotics*, vol. PP, no. 99, pp. 1–13, Aug. 2011.

[21] Z. Su, K. Hausman, Y. Chebotar, A. Molchanov, G. E. Loeb, G. S. Sukhatme, and S. Schaal, "Force estimation and slip detection/classification for grip control using a biomimetic tactile sensor," in *Humanoid Robots (Humanoids), 2015 IEEE-RAS 15th International Conference on*. IEEE, 2015, pp. 297–303.

[22] N. Wettels and G. Loeb, "Haptic feature extraction from a biomimetic tactile sensor: Force, contact location and curvature," in *2011 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Phuket, Thailand, Dec. 2011, pp. 2471 –2478.

[23] J. Li, S. Dong, and E. Adelson, "Slip Detection with Combined Tactile and Visual Information," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, May 2018, pp. 7772–7777.

[24] M. A. Abd, I. J. Gonzalez, T. C. Colestock, B. A. Kent, and E. D. Engeberg, "Direction of Slip Detection for Adaptive Grasp Force Control with a Dexterous Robotic Hand," in *2018 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, Jul. 2018, pp. 21–27.

[25] I. A. Sucan and S. Chitta, ""MoveIt!"," Oct. 2015. [Online]. Available: http://moveit.ros.org

[26] C. D. Mah and F. A. Mussa-Ivaldi, "Evidence for a specific internal representation of motion-force relationships during object manipulation," *Biological Cybernetics*, vol. 88, no. 1, pp. 60–72, Jan. 2003. [Online]. Available: http://link.springer.com/10.1007/s00422-002-0347-9

[27] G. Placencia, M. Rahimi, and B. Khoshnevis, "Sensing directionality in tangential haptic stimulation," in *Engineering Psychology and Cognitive Ergonomics*. Berlin, Heidelberg: Springer, 2009, pp. 253–261.

[28] C.-H. Lin, J. Fishel, and G. E. Loeb, "Estimating Point of Contact, Force and Torque in a Biomimetic Tactile Sensor with Deformable Skin," 2013. [Online]. Available: http://www.syntouchllc.com/Technology/_publications/201_Lin_Analytical.pdf

[29] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in PyTorch," in *NIPS-W*, 2017.

[30] Dixee Kimball, C. Huang, and K. Yang, "Generative Adversarial Networks for Multiclass Image Generation," *Stanford Computer Science*, 2016. [Online]. Available: https://web.stanford.edu/class/cs221//r2017estricted/p-final/dkimball/final.pdf

[31] S. Ioffe and C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," *arXiv:1502.03167 [cs]*, Feb. 2015, arXiv: 1502.03167. [Online]. Available: http://arxiv.org/abs/1502.03167

[32] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *International Conference for Learning Representations*, San Diego, CA, USA, 2015, pp. 1–13. [Online]. Available: https://dblp.org/db/conf/iclr/iclr2015.html

[33] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On Calibration of Modern Neural Networks," *arXiv:1706.04599 [cs]*, Jun. 2017, arXiv: 1706.04599. [Online]. Available: http://arxiv.org/abs/1706.04599

[34] "Random Errors and Prediction | STAT 462," 2018. [Online]. Available: https://newonlinecourses.science.psu.edu/stat462/node/254/

[35] D. A. Nix and A. S. Weigend, "Estimating the mean and variance of the target probability distribution," in *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)*, vol. 1, Jun. 1994, pp. 55–60 vol.1.

[36] A. Khosravi, S. Nahavandi, D. Creighton, and A. F. Atiya, "Comprehensive Review of Neural Network-Based Prediction Intervals and New Advances," *IEEE Transactions on Neural Networks*, vol. 22, no. 9, pp. 1341–1356, Sep. 2011.

[37] K. Drewing, M. Fritschi, R. Zopf, M. O. Ernst, and M. Buss, "First evaluation of a novel tactile display exerting shear force via lateral displacement," *ACM Transactions on Applied Perception (TAP)*, vol. 2, no. 2, pp. 118–131, 2005. [Online]. Available: http://dl.acm.org/citation.cfm?id=1060586

[38] M. P. Vitello, M. O. Ernst, and M. Fritschi, "An instance of tactile suppression: Active exploration impairs tactile sensitivity for the direction of lateral movement," in *Proceedings of EuroHaptics*, 2006, pp. 351–355.

[39] D. V. Keyson and A. J. M. Houtsma, "Directional sensitivity to a tactile point stimulus moving across the fingerpad," *Perception & Psychophysics*, vol. 57, no. 5, pp. 738–744, Jul. 1995. [Online]. Available: http://www.springerlink.com/index/10.3758/BF03213278

[40] T. Kim and Y.-L. Park, "A Soft Three-Axis Load Cell Using Liquid-Filled Three-Dimensional Microchannels in a Highly Deformable Elastomer," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 881–887, Apr. 2018. [Online]. Available: http://ieeexplore.ieee.org/document/8255581/

[41] W. Yuan, S. Dong, and E. Adelson, "GelSight: High-Resolution Robot Tactile Sensors for Estimating Geometry and Force," *Sensors*, vol. 17, no. 12, 2762, pp. 1–21, Nov. 2017. [Online]. Available: http://www.mdpi.com/1424-8220/17/12/2762

[42] M. Polic, I. Krajacic, N. Lepora, and M. Orsag, "Convolutional autoencoder for feature extraction in tactile sensing," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3671–3678, 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8758942/

[43] W. Yuan, M. A. Srinivasan, and E. H. Adelson, "Estimating object hardness with a GelSight touch sensor," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2016, pp. 208–215.

[44] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[45] B. Ward-Cherrier, N. Pestell, L. Cramphorn, B. Winstone, M. E. Giannaccini, J. Rossiter, and N. F. Lepora, "The TacTip Family: Soft Optical Tactile Sensors with 3D-Printed Biomimetic Morphologies," *Soft Robotics*, vol. 5, no. 2, pp. 216–227, Apr. 2018. [Online]. Available: https://www.liebertpub.com/doi/10.1089/soro.2017.0052

[46] R. L. Klatzky and S. J. Lederman, "Tactile roughness perception with a rigid link interposed between skin and surface," *Perception & Psychophysics*, vol. 61, no. 4, pp. 591–607, Jan. 1999. [Online]. Available: http://www.springerlink.com/index/10.3758/BF03205532

[47] R. L. Klatzky, S. J. Lederman, C. Hamilton, M. Grindley, and R. H. Swendsen, "Feeling textures through a probe: Effects of probe and surface geometry and exploratory factors," *Perception & Psychophysics*, vol. 65, no. 4, pp. 613–631, May 2003. [Online]. Available: http://www.springerlink.com/index/10.3758/BF03194587

[48] H. Culbertson, J. Unwin, and K. J. Kuchenbecker, "Modeling and Rendering Realistic Textures from Unconstrained Tool-Surface Interactions," *IEEE Transactions on Haptics*, vol. 7, no. 3, pp. 381–393, Jul. 2014.

[49] K. J. Kuchenbecker, R. C. Parajon, and M. P. Maggio, "Evaluation of a Vibrotactile Simulator for Dental Caries Detection," *Simulation in Healthcare*, vol. 12, no. 3, p. 148, Jun. 2017.

[50] N. Zamani and H. Culbertson, "Effects of Dental Glove Thickness on Tactile Perception Through a Tool," in *2019 IEEE World Haptics Conference (WHC)*, Tokyo, Japan, Jul. 2019, pp. 187–192.

[51] Y. Chebotar, O. Kroemer, and J. Peters, "Learning robot tactile sensing for object manipulation," in *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*. IEEE, 2014, pp. 3368–3375. [Online]. Available: http://ieeexplore.ieee.org/abstract/document/6943031/

[52] N. Harischandra and V. Dürr, "A forward model for an active tactile sensor using Echo State Networks," in *2012 IEEE International Symposium on Robotic and Sensors Environments Proceedings*, Nov. 2012, pp. 103–108.

[53] L. K. Pynn and J. F. DeSouza, "The function of efference copy signals: Implications for symptoms of schizophrenia," *Vision Research*, vol. 76, pp. 124–133, Jan. 2013. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0042698912003616

[54] F. F. and P. Payeur, "Dexterous Robotic Manipulation of Deformable Objects with Multi-Sensory Feedback - a Review," in *Robot Manipulators Trends and Development*, A. Jimenez and B. M. Al Hadithi, Eds. InTech, Mar. 2010.

[55] J. Zhu, B. Navarro, R. Passama, P. Fraisse, A. Crosnier, and A. Cherubini, "Robotic Manipulation Planning for Shaping Deformable Linear Objects With Environmental Contacts," *IEEE Robotics and Automation Letters*, vol. 5, no. 1, pp. 16–23, Jan. 2020, conference Name: IEEE Robotics and Automation Letters.

[56] A. Nair, D. Chen, P. Agrawal, P. Isola, P. Abbeel, J. Malik, and S. Levine, "Combining self-supervised learning and imitation for vision-based rope manipulation," in *2017 IEEE International Conference on Robotics and Automation (ICRA)*, May 2017, pp. 2146–2153.

[57] H. Han, G. Paul, and T. Matsubara, "Model-based reinforcement learning approach for deformable linear object manipulation," in *2017 13th IEEE Conference on Automation Science and Engineering (CASE)*, Aug. 2017, pp. 750–755, iSSN: 2161-8089.

[58] M. Yan, Y. Zhu, N. Jin, and J. Bohg, "Self-Supervised Learning of State Estimation for Manipulating Deformable Linear Objects," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2372–2379, Apr. 2020, conference Name: IEEE Robotics and Automation Letters.

[59] M. Meyer, G. Debunne, M. Desbrun, and A. H. Barr, "Interactive animation of cloth-like objects in virtual reality," *The Journal of Visualization and Computer Animation*, vol. 12, no. 1, pp. 1–12, 2001, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/vis.244. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/vis.244

[60] S. Natsupakpong and M. Cenk Çavuşoğlu, "Determination of elasticity parameters in lumped element (mass-spring) models of deformable objects," *Graphical Models*, vol. 72, no. 6, pp. 61–73, Nov. 2010. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S1524070310000147

[61] J. Sanchez, C. M. Mateo, J. A. Corrales, B. Bouzgarrou, and Y. Mezouar, "Online Shape Estimation based on Tactile Sensing and Deformation Modeling for Robot Manipulation," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct. 2018, pp. 504–511, iSSN: 2153-0866.

[62] I. Leizea, H. Álvarez, I. Aguinaga, and D. Borro, "Real-time deformation, registration and tracking of solids based on physical simulation," in *2014 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, Sep. 2014, pp. 165–170.

[63] J. Berkley, G. Turkiyyah, D. Berg, M. Ganter, and S. Weghorst, "Real-time finite element modeling for surgery simulation: an application to virtual suturing," *IEEE*

Transactions on Visualization and Computer Graphics, vol. 10, no. 3, pp. 314–325, May 2004, conference Name: IEEE Transactions on Visualization and Computer Graphics.

[64] B. Lloyd, G. Szekely, and M. Harders, "Identification of Spring Parameters for Deformable Object Simulation," *IEEE Transactions on Visualization and Computer Graphics*, vol. 13, no. 5, pp. 1081–1094, Sep. 2007, conference Name: IEEE Transactions on Visualization and Computer Graphics.

[65] Y. Wang, D. McConachie, and D. Berenson, "Tracking Partially-Occluded Deformable Objects while Enforcing Geometric Constraints," *arXiv:2011.00627 [cs]*, Nov. 2020, arXiv: 2011.00627.

[66] Y. Song, K. Yang, X. Jiang, and Y. Liu, "Vision Based Topological State Recognition for Deformable Linear Object Untangling Conducted in Unknown Background," in *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, Dec. 2019, pp. 790–795, iSSN: null.

[67] T. Bretl and Z. McCarthy, "Quasi-static manipulation of a Kirchhoff elastic rod based on a geometric analysis of equilibrium configurations," *Intl Journal of Robotics Research*, vol. 33, no. 1, pp. 48–68, 2014.

[68] D. Katic and M. Vukobratovic, "A neural network-based classification of environment dynamics models for compliant control of manipulation robots," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 28, no. 1, pp. 58–69, Feb. 1998, conference Name: IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics).

[69] A. X. Lee, H. Lu, A. Gupta, S. Levine, and P. Abbeel, "Learning force-based manipulation of deformable objects from multiple demonstrations," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2015, pp. 177–184.

[70] S. Yue and D. Henrich, "Manipulating deformable linear objects: sensor-based fast manipulation during vibration," in *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No.02CH37292)*, vol. 3, May 2002, pp. 2467–2472 vol.3.

[71] F. Ding, J. Huang, Y. Wang, T. Matsuno, and T. Fukuda, "Vibration damping in manipulation of deformable linear objects using sliding mode control," *Advanced Robotics*, vol. 28, no. 3, pp. 157–172, Feb. 2014. [Online]. Available: http://www.tandfonline.com/doi/abs/10.1080/01691864.2013.861769

[72] G. Duchemin, P. Maillet, P. Poignet, E. Dombre, and F. Pierrot, "A hybrid position/force control approach for identification of deformation models of skin and underlying tissues," *IEEE Transactions on Biomedical Engineering*, vol. 52, no. 2, pp. 160–170, Feb. 2005, conference Name: IEEE Transactions on Biomedical Engineering.

[73] M. K. Soltani, S. Khanmohammadi, F. Ghalichi, and F. Janabi-Sharifi, "A soft robotics nonlinear hybrid position/force control for tendon driven catheters," *International Journal of Control, Automation and Systems*, vol. 15, no. 1, pp. 54–63, Feb. 2017. [Online]. Available: https://doi.org/10.1007/s12555-016-0461-4

[74] J. Acker and D. Henrich, "Manipulating deformable linear objects: characteristic features for vision-based detection of contact state transitions," in *Proceedings of the IEEE International Symposium onAssembly and Task Planning, 2003.* Besancon, France: IEEE, 2003, pp. 204–209. [Online]. Available: http://ieeexplore.ieee.org/document/1217212/

[75] R. J. Anderson and M. W. Spong, "Hybrid impedance control of robotic manipulators," *IEEE Journal on Robotics and Automation*, vol. 4, no. 5, pp. 549–556, 1988.

[76] F. B. Carlson, "On the Calibration of Force/Torque Sensors in Robotics," *arXiv:1904.06158 [cs]*, Apr. 2019, arXiv: 1904.06158. [Online]. Available: http://arxiv.org/abs/1904.06158

[77] B. Siciliano and O. Khatib, Eds., *Springer Handbook of Robotics*. Berlin Heidelberg: Springer-Verlag, 2008. [Online]. Available: http://springerlink.com/content/r6m219/

[78] "Young's Modulus: Tensile Elasticity Units, Factors & Material Table." [Online]. Available: https://omnexus.specialchem.com/polymer-properties/properties/young-modulus

[79] "Rope/wire: Axial and bending stiffness." [Online]. Available: https://www.orcina.com/webhelp/OrcaFlex

[80] A. J. Shah and J. A. Shah, "Towards manipulation planning for multiple interlinked deformable linear objects," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*. Stockholm, Sweden: IEEE, May 2016, pp. 3908–3915. [Online]. Available: http://ieeexplore.ieee.org/document/7487580/

[81] C. Ott, R. Mukherjee, and Y. Nakamura, "Unified impedance and admittance control," in *Robotics and Automation (ICRA), 2010 IEEE International Conference on*. IEEE, 2010, pp. 554–561.

[82] W. J. Shipman and L. C. Coetzee, "Reinforcement learning and deep neural networks for pi controller tuning," *IFAC Symposium on Control, Optimization and Automation in Mining, Mineral and Metal Processing*, vol. 52, no. 14, pp. 111–116, 2019.

[83] Y. Narang, K. Van Wyk, A. Mousavian, and D. Fox, "Interpreting and Predicting Tactile Signals via a Physics-Based and Data-Driven Framework," in *Robotics: Science and Systems XVI*. Robotics: Science and Systems Foundation, Jul. 2020. [Online]. Available: http://www.roboticsproceedings.org/rss16/p084.pdf

[84] A. Prayitno, V. Indrawati, and I. Immanuel Trusulaw, "Fuzzy Gain Scheduling PID Control for Position of the AR.Drone," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 8, no. 4, p. 1939, Aug. 2018. [Online]. Available: http://ijece.iaescore.com/index.php/IJECE/article/view/7290

[85] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occurring in the statistical analysis of probabilistic functions of markov chains," *The annals of mathematical statistics*, vol. 41, no. 1, pp. 164–171, 1970.