Design, Realization, and Verification of a Planar, Tendon-Driven, Variable Stiffness Finger

Allison Goetz
Marquette University

Recommended Citation

Goetz, Allison, "Design, Realization, and Verification of a Planar, Tendon-Driven, Variable Stiffness Finger" (2023). *Master's Theses (2009 -)*. 746.
https://epublications.marquette.edu/theses_open/746

DESIGN, REALIZATION, AND VERIFICATION OF A PLANAR, TENDON-DRIVEN, VARIABLE STIFFNESS FINGER

by

Allison L. Goetz, B.S.

A Thesis submitted to the Faculty of the Graduate School,
Marquette University,
in Partial Fulfillment of the Requirements for
the degree of Master of Science

Milwaukee, Wisconsin
May 2023

ABSTRACT
DESIGN, REALIZATION, AND VERIFICATION OF A PLANAR, TENDON-DRIVEN, VARIABLE STIFFNESS FINGER

Allison L. Goetz, B.S.

Marquette University, 2023

Traditional robotic manipulators are designed for accurate absolute positioning and require a highly structured environment to perform simple operations. They are unable to compensate for small discrepancies in the relative positioning of the robot end-effector with respect to its environment. A task-appropriate compliance allows a robotic manipulator to compensate for small discrepancies in the position or orientation of an object when the object is constrained. A better means of realizing task-appropriate structured compliant behavior is needed to allow a robot to perform dexterous manipulation.

In this thesis, three agonist-antagonist variable stiffness actuators (VSAs) are used to independently control the joint stiffnesses and positions in a 3-DOF finger. The VSAs allow the finger to achieve a large range of controllable 2-dimensional linear elastic behavior at the fingertip. The joint design and actuation strategy enable the independent control of the angular position and stiffness of each joint, without coupling. Each VSA employs two motors in opposition adjusting the length of a cable running through a series of pulleys with a spring-loaded lever. A quasi-static model was developed to evaluate the elastic performance of the design. A set of geometric parameters were optimized to produce the desired quadratic force-deflection behavior for one half of the VSA so that, when the two halves act in opposition, a linear torque-angular deflection relationship is obtained at the joint.

A prototype was assembled and tested to verify the performance of the VSAs and the controllable endpoint compliance. The optimized half-VSA mechanism produced approximately quadratic tension-deflection behavior in the tendons. While the experimental relationships did not closely match the analytical results and differed between joints, the nearly quadratic behavior was achieved in all joints, allowing the antagonistic variable stiffness actuators to generate linear variable stiffness. The controllable linear behavior at the joints was used to adjust the particle planar (2D) compliance at the fingertip. The shape and magnitude of the experimentally obtained fingertip compliance ellipses matched the analytical results well for most configurations.

TABLE OF CONTENTS

CHAPTER

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Traditional robotic manipulators excel in controlled settings and depend on positional accuracy for reliable performance. However, unstructured environments introduce uncertainties in the locations of physical constraints that a stiff, or non-compliant, robot is not equipped to handle. This problem has been identified as "one of the central challenges of robotics" because these uncertainties make controlling contact forces and accurate relative positioning of objects difficult [4]. Conventional robotic manipulators are designed for accurate absolute positioning and require a highly structured environment to perform simple operations. They are unable to compensate for small discrepancies in the relative positioning of the robot end-effector with respect to its environment.

To address the limitations of stiff robots and to facilitate the dexterous manipulation of objects, a task-appropriate structured compliance can be incorporated into the system [5, 6, 7, 8, 9]. A task-appropriate compliance allows a robotic manipulator to compensate for small discrepancies in the position or orientation of an object when the object is constrained. Better means of realizing task-appropriate structured compliant behavior are needed to allow a robot to perform dexterous manipulation. This work focuses on achieving a large range of controllable compliance at the endpoint of a robotic finger in order to enable the realization of a task-appropriate compliance for dexterous manipulation.

Compliance in a robotic manipulator may be attained actively or passively. An overview of each method is given below.

1.1 Active Stiffness Control

Active stiffness (or compliance) involves sensors and feedback control of a stiff robot to simulate the behavior of a conventional spring. Contact forces are measured and positions are modified to obtain a specified relationship between force and displacement at the end effector. This method can be advantageous as it allows compliance adjustment during operation, so the same robot could theoretically perform a wide range of constrained manipulation tasks.

This real-time control of endpoint forces is limited by the speed of the sensors, actuators, and controllers involved in the active control strategy of the robot. Further, dynamic stability becomes a concern with stiffness control. In [10], Hogan emphasizes that “a system that is stable in isolation can become unstable when coupled to an environment that is itself stable.” Even with the implementation of active impedance control, the controlled impedance is not inherently passive so “stability cannot be guaranteed with all environments” [10].

1.2 Passive Compliance

Alternatively, passive compliance is achieved by incorporating elastic elements into the system, allowing the end effector to compensate for variability in the positioning in a constrained task. This method can be implemented in an end-effector mounted compliant wrist or may be added to the robotic joints as in Series Elastic Actuators (SEAs).

End-effector mounted devices incorporate a specific passive stiffness into a robotic gripper mechanism. An RCC device [11] introduces compliance between the robot and gripper and “establishes motion about a remote center of compliance typically at, near, or beyond the functioning end of the operator member” [11]. In other words, the compliance center is projected to a location that leads the held object to its properly mated position.

Series Elastic Actuators (SEAs) add intrinsic passive compliance in robot joints. Typically, a compliant element is placed between a “rigid” actuator and a “rigid” robot link. SEAs improve relative positioning and force control in compliant manipulators because the output force is easily calculated from the known spring constant and displacement information provided by sensors [12]. The arms of the ReThink Robotics Baxter [13] and Sawyer [14] and the arms of the NASA Robonaut 2 [15] are equipped with SEAs to add joint compliance and provide safer robot-human interaction. However, the elasticity provided by SEAs can result in inaccurate positioning in free space and excessive oscillation in positioning when initiating or terminating motion.

Although these inherent (uncontrolled) passive compliance methods have fast dynamic response characteristics and add a level of compliance to an otherwise stiff robotic manipulator, wrist compliance and joint stiffness cannot be altered in real time. Unlike active compliance, these physical properties are constant, determined by the elastic elements used in the device [16]. An improved approach to implement passive stiffness control involves the use of variable stiffness actuators (VSAs), enabling time-varying modulation of joint compliance.

1.3 Variable Stiffness Actuators

Using variable stiffness actuators, a robot manipulator can have the advantages of the fast dynamic response of passive realization and the controllability of active realization by real-time modulation of elastic joints. The interest in, and development of, VSAs can be attributed to their “ability to minimize large forces due to shocks, to safely interact with the user, and their ability to store and release energy in passive elastic elements" [17]. VSAs typically contain two actuators in a series or a parallel arrangement with passive elastic elements, allowing both the position and the stiffness of a joint to be controlled. An overview of several types of VSAs is presented by Vanderborght et al. in [18].

A VSA with a serial structure, shown in Figure 1.1(a), generally uses one actuator for stiffness control and another for position control of the joint. This structure simplifies the independent control of position and stiffness. However, because the two actuators are located at or near the joint, it also results in higher suspended inertia in multi-link serial manipulators, so keeping the system compact while maintaining the desired torque capabilities is difficult. Designs utilizing the serial configuration include the Variable Stiffness Unit (VSU) [19], the variable stiffness joint (VS-Joint) [20], and the Arched Flexure VSA [21].

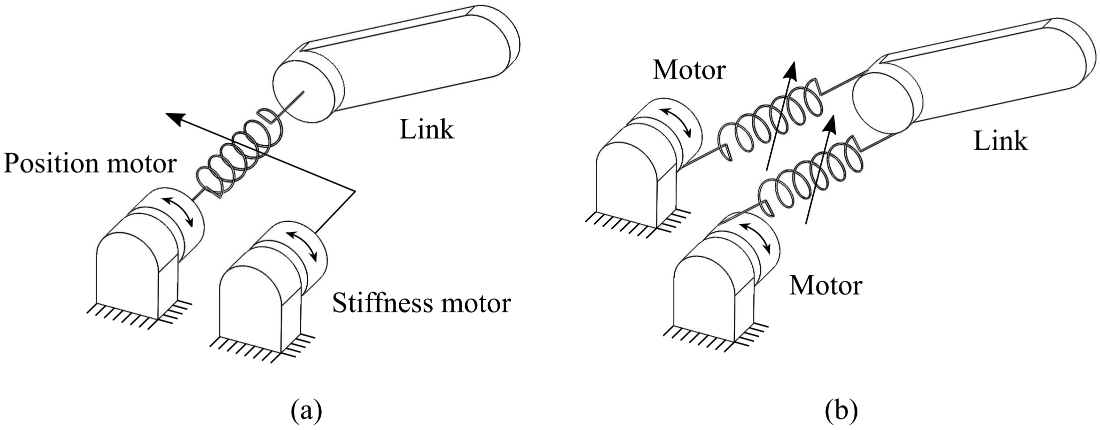

Figure 1.1. Two Variable Stiffness Actuator Structures

A parallel structure VSA, shown in Figure 1.1(b), also uses two motors. Each motor connects to a link through a nonlinear elastic component that together are used to adjust the joint stiffness and position. Because the nonlinear elastic elements can act as transmission components, actuators do not need to be located close to the joint. This configuration allows all actuators to be placed at the base of a multi-link serial manipulator.

1.3.1 Antagonistic, Cable-Driven, Variable Stiffness Actuators

Tendon-driven antagonistic actuators allow the parallel VSA motors to be located away from the robot joints, resulting in lower suspended inertia. This ensures that elastic (rather than inertia or damping) properties dominate the dynamic behavior of the robot. Maintaining low inertia at the joints decreases the torque requirements of other joints in the system. The actuation and implementation methods of several antagonistic, tendon-driven VSAs are described below.

One type of parallel VSA uses opposing McKibben pneumatic artificial muscles (PAMs) for antagonistic actuation. The static mechanical properties of McKibben actuators resemble those of biological muscles [22]. In [23], Trumić et al. selected a pair of McKibben artificial muscles to control a robotic link, citing their high force-to-mass ratio and the simplicity of the mechanism. However, the stiffness range in McKibben PAMs has been shown to be position dependent [22] and the need for pressurized air limits mobility [24].

The German Aerospace Center (DLR) designed an “anthropomorphic hand arm system” using several VSA methods to achieve improved manipulation robustness and dexterity in a robot system [25]. These include the Floating Spring Joint (FSJ) [26] and the bidirectional antagonistic variable stiffness (BAVS) [27] mechanisms. The BAVS mechanism, used in the lower arm and wrist [28], has a parallel VSA configuration in which both actuators generate torque on the link in both directions. While this increases torque capabilities at the link, the design complicates cable routing and requires constant pretensioning of the mechanism.

The tendon-driven Flexible Antagonistic Spring (FAS) mechanism, developed for the hand of the DLR hand-arm system, employs a series of pulleys and a spring-loaded lever for each motor connected in parallel. In Figure 1.2, one half of a complete FAS mechanism is shown. If the cable endpoint (arrowhead) is constrained as the motor pulley winds the cable, the shortened cable acting on the spring pulley causes rotation of the lever about the guide pulley. As the tension spring is elongated, the force in the cable is increased. The arrowhead side of the cable connects to one side of the link near the joint, and an identical mechanism acts on the link on the opposite side of the joint. In [1], it was stated that in order to "obtain the required stiffness characteristics," three geometric parameters and the spring rate were varied to select the physical layout of the mechanism. The desired force-deflection performance, however, was not explicitly specified. The models for the index finger metacarpal joint behavior [1] do show that the force and stiffness values vary nonlinearly with joint deflection for most tendon loads.

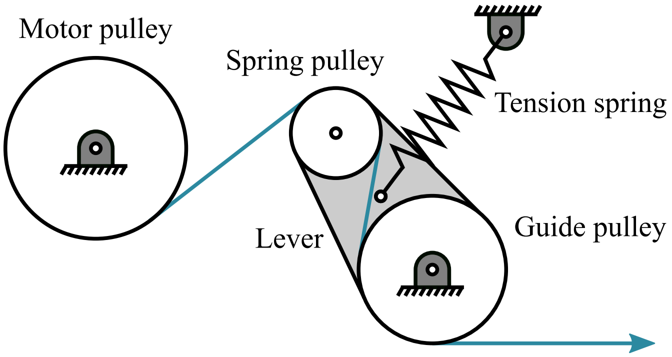

Figure 1.2. DLR Flexible Antagonistic Spring (FAS) VSA Mechanism (Adapted from [1])

The antagonistic, cable driven, variable stiffness actuator described in [2, 29] was explicitly designed to achieve variable stiffness with linear force-deflection characteristics. Similar to the FAS, the antagonistic quadratic spring mechanism illustrated in Figure 1.3 employs a cable running through a system of pulleys with a spring-loaded lever. Two mechanisms of this type connected in parallel perform

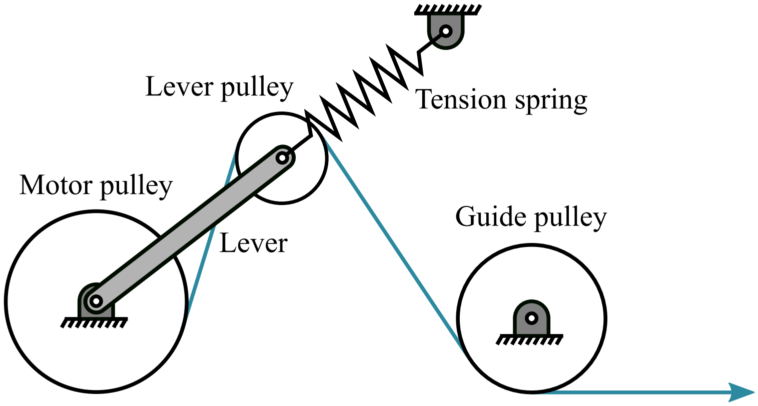

Figure 1.3. Antagonistic Quadratic Spring VSA Design (Adapted from [2])

as two opposing quadratic springs acting on the joint, as described in the following subsection.

1.3.2 Antagonistic Quadratic Spring Behavior

Geometric parameters of the mechanism layout in [2, 29] were optimized to obtain the desired quadratic force-deflection behavior in each opposing mechanism to achieve linear elastic behavior at the joint. While the DLR FAS mechanism in [1] was optimized for anthropomorphic functionality, achieving linear force-deflection behavior in the joint is advantageous to achieving a specified endpoint compliance.

The nonlinear behavior exhibited by each side of the VSA was optimized to produce a controllable stiffness mechanism with linear force-deflection characteristics when each side is antagonistically connected to the link. Linear force-deflection elastic behavior for a link is obtained by placing two elements with quadratic force-deflection behavior in opposition to one another [3], as illustrated in Figure 1.4.

The forces imposed on the link by the two springs are:

$$F_L = K(x - x_L)^2 \tag{1.1}$$

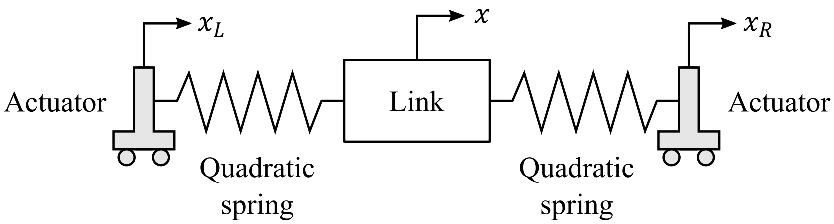

Figure 1.4. Translational Antagonistic Quadratic Spring VSA Configuration (Adapted from [3])

$$F_R = -K(x - x_R)^2 \tag{1.2}$$

The net force acting on the link is:

$$F_L + F_R = K(x^2 - 2xx_L + x_L^2 - x^2 + 2xx_R - x_R^2) \tag{1.3}$$

$$F_L + F_R = 2K(x_R - x_L)\left[x - \frac{1}{2}(x_L + x_R)\right] \tag{1.4}$$

$$F = K_{link}(x - x_{eq}) \tag{1.5}$$

Note that commanded actuator positions (x_R, x_L) determine the link equilibrium position (x_{eq}) and the link stiffness (K_{link}). The link equilibrium position is proportional to the sum of commanded spring positions $(x_R + x_L)$, while the link stiffness is proportional to the difference of commanded spring positions $(x_R - x_L)$. The net force acting on the body $F = F_L + F_R$ varies linearly with the deflection of the body from its equilibrium position $(x - x_{eq})$. The control inputs $(x_R - x_L)$ and $(x_R + x_L)$ can be manipulated to produce any joint position and any joint stiffness desired if Equations 1.1 and 1.2 are valid for a large range of control inputs.

The maximum joint stiffness (approx. 0.7 N/mm) of the opposing quadratic spring VSA in [2, 29] was roughly 4 times larger than the minimum joint stiffness (approx. 0.2 N/mm) within the optimized linear segment of the modeled stiffness-deflection curve.

1.3.3 Effective Power Transmission in Cable-Driven Serial Mechanisms

The cable-driven, antagonistic VSA mechanisms presented in [1] and [2] are suitable for integration in a compliant hand, as the actuators can be placed in a housing located away from the fingers, with joint position and stiffness adjusted simultaneously via antagonistic actuation.

The tendon routing of a robotic finger impacts the control method and capabilities of the manipulator. Coupling between the motion of several joints will result when a single tendon exerts torque on more than one joint. The cable routing must minimize the relationship between joint motion and tendon forces so that: 1) the adjustment of the stiffness of a distal joint does not affect the configuration of the finger, and 2) the adjustment of joint position does not affect the stiffness or position of more distal joints. Prior cable routing methods used in robotic fingers are briefly reviewed below.

Underactuated robotic fingers contain coupled joints, in which one actuator moves two or more joints. This can decrease the number of required actuators, but does not allow the independent control of joint position or stiffness. By contrast, each actuator in a fully actuated and fully decoupled robotic finger moves only one joint.

The DLR developed two underactuated hands, 1) the WHISG (a Wearable Hand to Investigate Passive Stiffness in Grasping) hand and 2) the CLASH (Compliant Low cost Antagonistic Servo Hand) 3F hand [30].

The 4-DOF main finger (thumb) of the WHISG hand has a 2-DOF base joint and two coupled 1-DOF joints. The tendons in the finger are illustrated in Figure 1.5. The double green arrowheads indicate that cables act on either side of the finger, controlling the two degrees of freedom of the base joint. The other tendons are located centrally along the length of the finger and affect motion at all three joints. The blue tendon closes the finger and the purple tendon opens the finger.

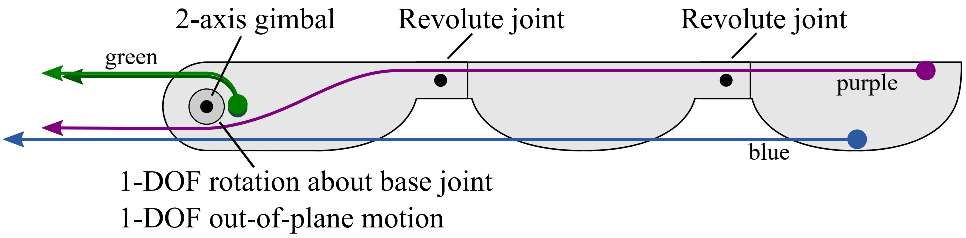

Figure 1.5. Tendon Routing of the WHISG Finger

The CLASH 3F hand was developed with a 3-DOF thumb. Like the WHISG thumb, it contains a 2-DOF base joint, but has only one subsequent 1-DOF revolute joint. The tendon routing of the CLASH 3F finger is illustrated in Figure 1.6. The green tendon affects only one degree of freedom at the base joint, while the blue tendon affects motion at both joints. The purple double arrowhead denotes two tendons on opposite sides of the finger acting on the two degrees of freedom at the base joint, as well as the 1-DOF distal joint.

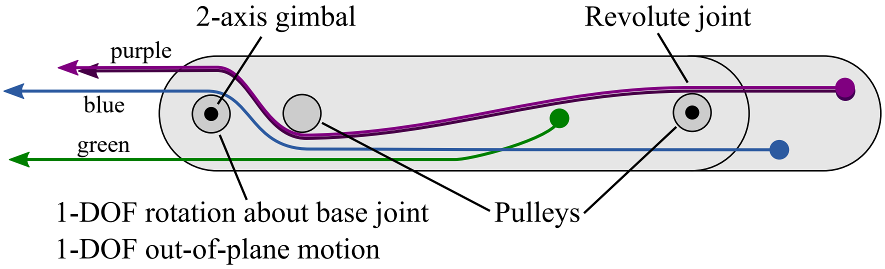

Figure 1.6. Tendon Routing of the CLASH 3F Finger

Another cable-driven dexterous manipulator, the underactuated Shadow Dexterous Hand [31] uses antagonistic McKibben pneumatic muscles in the forearm for the actuation of the wrist and hand [32]. Each base joint has 2 independently-actuated degrees of freedom. However, like the WHISG finger in Figure 1.5, the actuation of the two 1-DOF joints in each finger is coupled.

The joint coupling demonstrated in the underactuated fingers of the WHISG hand, the CLASH 3F hand, and the Shadow Dexterous Hand complicates the control of each joint stiffness and therefore the control of the endpoint stiffness. The power transmission in a fully actuated hand, developed by the DLR, is described below.

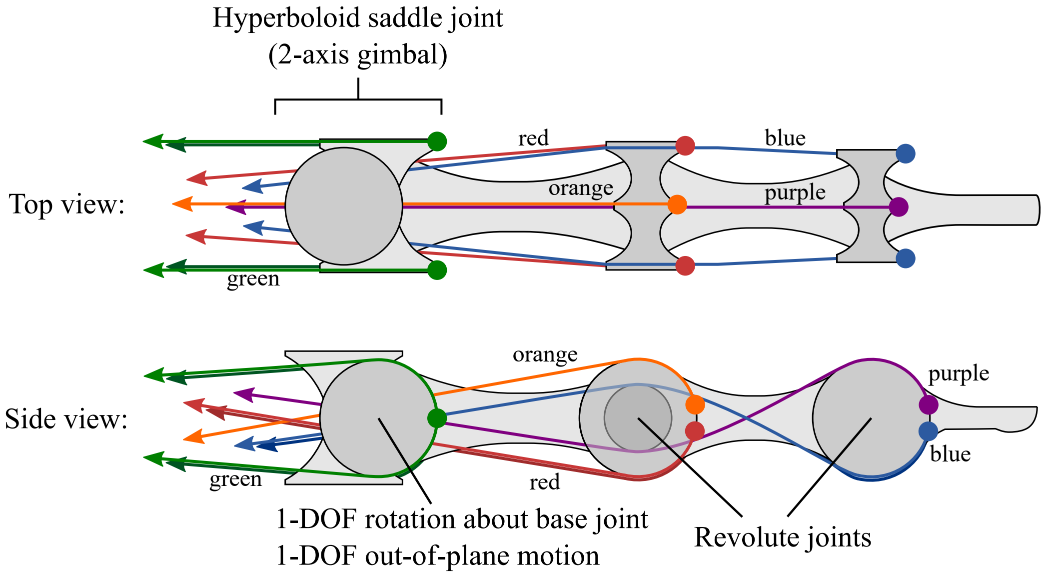

Figure 1.7. Tendon Routing of a Finger of the DLR Hand

The fully actuated hand of the DLR hand-arm system is controlled by FAS mechanisms housed in the forearm [24]. A 2-DOF hyperboloid saddle joint reduces actuation coupling at the base of each finger. Subsequent joints are 1-DOF revolute joints. The double arrowheads denote tendons acting on opposite sides of the finger, while tendons shown with a single arrow are located centrally between each side of the finger. At each 1-DOF joint, there are two flexors and one extensor to “create a force triangle and increase the joint stability” [33]. These are visible in the top view in Figure 1.7 where the tendons responsible for flexion and extension of the distal joint are shown in blue and purple, respectively. The flexor and extensor tendons of the middle joint are shown in red and orange, respectively. All tendons are routed internally at the base joint, reducing the coupling between the motion of the base joint and the motion of successive joints. However, the tendons responsible for actuation of the distal joint do not appear to pass through the center of the middle joint, but are guided over an internal pulley. The green tendons control the two degrees of freedom at the base joint.

1.4 Approach Overview

Recall that a better means of realizing task-appropriate structured compliance is needed to enable the dexterous manipulation of a held object and allow the manipulator to compensate for discrepancies in the position or orientation of the object. Three individually controlled joints in series allow the realization of an arbitrary point planar compliance matrix at the fingertip [34]. In order to provide a held object with a desired planar compliance, the stiffness of each joint in a hand must be easily and independently controllable and allow a wide range of stiffness values. The point planar compliance matrix at the fingertip, however, is limited by the stiffness range of the joints and the configuration of the finger.

The nonlinear elastic behavior produced by the DLR FAS mechanism [1] is not ideal in an application where linear stiffness in the joints is desired. If the elastic behavior at the joint is linear, then for a small deflection from an equilibrium position, the compliance matrix is determined by the finger configuration and the finger joint compliances, as described below.

If the coordinate frame used to describe the compliance is placed at the fingertip, the finger configuration can be described by the location $\mathbf{r}_i$ of each joint relative to the endpoint coordinate frame, given by

$$\mathbf{r}_i = \begin{bmatrix} x_i \\ y_i \end{bmatrix}. \tag{1.6}$$

The corresponding joint twist $\mathbf{t}_i$ for each joint is given by

$$\mathbf{t}_i = \begin{bmatrix} y_i \\ -x_i \end{bmatrix}. \tag{1.7}$$

The 2 × 2 positive semidefinite compliance matrix $\mathbf{C}$ at the endpoint of a 3-joint finger is

$$\mathbf{C} = c_1\mathbf{t}_1\mathbf{t}_1^T + c_2\mathbf{t}_2\mathbf{t}_2^T + c_3\mathbf{t}_3\mathbf{t}_3^T \tag{1.8}$$

where c_i is the joint compliance. A desired compliance matrix $\mathbf{C}$ at the fingertip can be realized by determining the joint location $\mathbf{r}_i$ and joint compliance c_i for each joint following the synthesis procedure in [34].

The mechanism in [2] improved the design of [24] by providing linear elastic behavior at the joint over a known range of cable deflection. However, the range of stiffness values was quite limited. In the revised design presented here, the range of joint linear stiffness values is increased to increase the manipulation capabilities of a compliant finger, and ultimately a compliant hand.

Actuation coupling was implemented in the underactuated fingers of the WHISG, CLASH 3F, and Shadow hands [30, 32]. While actuation coupling between joints mimics human tendon routing, it complicates the control of the endpoint Cartesian stiffness matrix. Current variable stiffness finger designs [24, 30, 31] incorporate 2 degrees of freedom at the base of the finger to simulate the motion of a human finger. The 2-DOF base joint of a spatial finger complicates routing and is unnecessary for a finger designed for integration onto a planar hand.

To address the limitations of previous VSA and finger designs, an increased controllable range of endpoint stiffness with minimal actuation coupling at the joints is needed. To do this, an antagonistic VSA mechanism is optimized to produce improved quadratic force-deflection behavior, which results in a larger range of linear elastic behavior in each joint. Over the optimized cable displacement range, the stiffness of the joint varies linearly to allow the relatively straightforward control of joint compliance. Additional geometric parameters are included as design variables in the optimization of the VSA to expand the effective design space to achieve a larger range of linear stiffness values.

The tendon anchoring and cable routing are designed to maintain stability and avoid coupling of the joints. Undesirable coupling between subsequent joints is reduced by routing cables through the center of more proximal joints. Central tendon routing provides a direct relationship between the actuator positions and the position and stiffness of the associated joints. The proposed finger design, described in the following subsections, allows independent control of joints in series and is desirable so that changing the position or stiffness of one joint does not impact the position or stiffness of subsequent joints in the finger.

1.4.1 Design Objectives

The primary objective of this work is to design a 3-DOF planar finger that achieves a large range of controllable 2-dimensional linear elastic behavior at the fingertip. To do this: 1) a large range of controllable linear elastic behavior at each joint is needed and 2) the actuation and transmission must decouple the elastic and kinematic behavior for each joint.

The 3-DOF finger is intended for future use on a 3-finger planar hand with variable stiffness joints for constrained manipulation tasks. A finger resembling the size and range of motion of an anthropomorphic finger maintains the compactness and mobility appropriate for the development of a robotic hand.

1.4.2 Design Overview

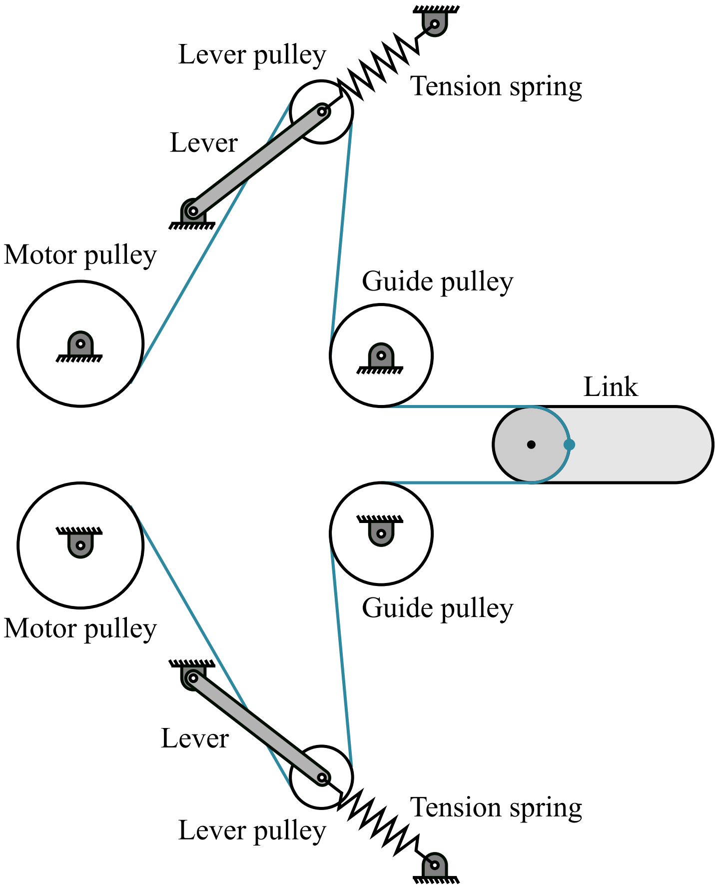

Figure 1.8. Modified Antagonistic Quadratic Spring VSA Design

Recall that three individually controlled joints in series are needed for the realization of an arbitrary point planar compliance at the fingertip, as described in Section 1.4. Here, three agonist-antagonist variable stiffness actuators (VSAs) are used to control the joints of a 3-DOF finger. The joint design and actuation strategy allow the angular position and stiffness of each joint to be independently controlled, without coupling. Each VSA (illustrated in Fig. 1.8) employs two motors in opposition adjusting the length of a cable running through a series of pulleys with spring-loaded levers. The two motor pulleys wind the cable both to modulate the position and the stiffness of the link. If both motor pulleys wind the cable in opposite directions to decrease the length of cable equally on both halves of the mechanism, the levers are moved and the two conventional tension springs are elongated. This increases the force in the cable and the effective stiffness of the joint without changing the joint position. If the motor pulleys rotate the same amount in the same direction, such that the length of cable within the system does not change, only the joint angle (i.e., the equilibrium point) changes, not the joint stiffness.

A set of geometric and elastic parameters are optimized to produce a quadratic force-deflection curve for one half of the VSA. The lever mechanism in [2, 29] has been modified to achieve a larger range of linear stiffness behavior over a specified range of cable deflection. Using additional geometric parameters in the optimization enlarges the design space to allow the generation of mechanism designs with better results, i.e., a greater linear stiffness range.

To achieve the design objective of kinematic and elastic decoupling, the tendons are routed internally through the center of the joints as shown in Figure 1.9.

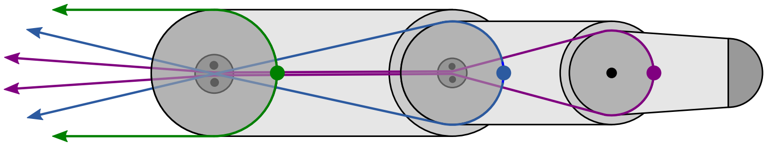

Figure 1.9. Conceptual Finger Design with Central Tendon Routing

1.5 Thesis Overview

This thesis presents the design and performance verification of a 3-DOF finger with each joint antagonistically actuated with variable stiffness actuators. The optimized VSA yields variable linear stiffness behavior with an analytical stiffness ratio (highest to lowest stiffness) of approximately 8. Chapter 2 describes the overall finger design and system level functions. Chapter 3 details the design and performance of the variable stiffness actuators used in the finger. Chapter 4 presents the testing procedures and results that verify system performance of the finger.

Chapter 5 summarizes the significance of the results and possibilities for future work.

CHAPTER 2

THREE DEGREE OF FREEDOM FINGER DESIGN

In this chapter, the design of a 3-DOF finger to achieve a controllable end-point particle planar stiffness matrix is described. Specifically, the selection of the finger geometry, joint actuation, power transmission, and control method are presented. The modified antagonistic quadratic VSA design is described in Chapter 3.

The finger is designed to be strong and lightweight in order to reduce the suspended inertia while supporting the forces necessary to achieve the desired end-point elastic behavior. Tendons actuating distal links are routed through the center of more proximal joints to minimize actuation coupling between subsequent joints and allow the independent control of the position and stiffness at each joint. Further, differential mechanisms were designed to load the finger symmetrically, minimizing out-of-plane moments in the system.

An overview of the system design is provided in Section 2.1. The design of specific features of the finger and variable stiffness actuation are detailed in Section 2.2. The selection and design of transmission features are presented in Section 2.3. The control method is described in Section 2.4.

2.1 Design Overview

Agonist-antagonist VSAs have an actuation technique similar to human skeletal muscles. They are only able to exert tension on a joint, so several muscles contract or relax simultaneously to achieve the joint position and stiffness required for a given task. Here, two quadratic spring actuators are connected antagonistically to each robotic link to produce linear elastic behavior at the joint. The use of tendons for power transmission allows the effective joint inertia to be minimized, and the routing method minimizes actuation coupling between subsequent joints. The control system reads encoder positions and controls stepper motors to achieve a commanded position and stiffness at each joint to produce a desired endpoint compliance matrix.

2.2 Finger Structure

The three sections of a human finger (not including the thumb) are the proximal phalanx (PP), medial phalanx (MP), and distal phalanx (DP). The base joint where the finger connects to the hand is the metacarpophalangeal joint (MCPJ). The proximal interphalangeal joint (PIPJ) connects the PP and MP and the distal interphalangeal joint (DIPJ) connects the MP and PP. These finger sections and joints are illustrated in Figure 2.1.

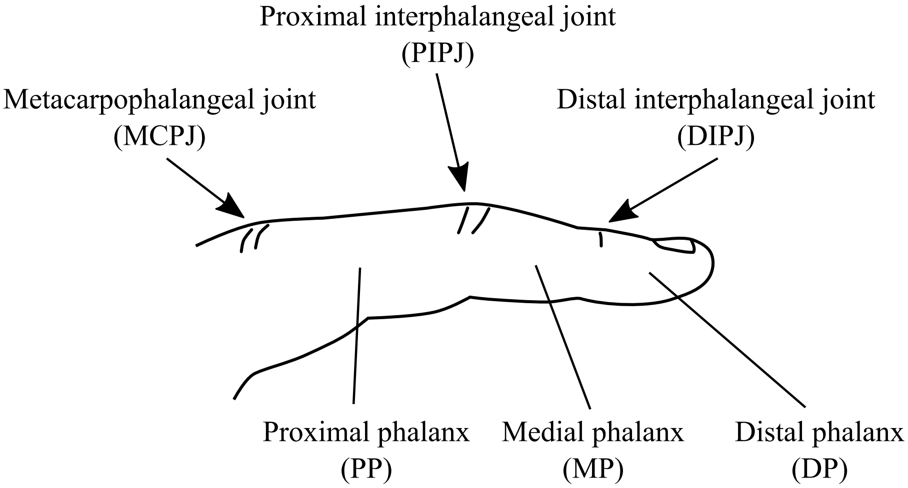

Figure 2.1. Human Finger Sections and Joints

In order to maintain the approximate size of an anthropomorphic finger while allowing space for the cable routing, the finger links are designed to be approximately one standard deviation larger than the sections of an average human third finger [35] as indicated in Table 2.1. The sections and joints corresponding to a human finger are identified in the CAD model of the variable stiffness finger in Figure 2.2.

Table 2.1. Average Third Finger Section Lengths and Selected Design Values

Finger Section	**Average Human**	**Design**
Proximal phalanx	44.63 ± 3.81 mm	48.50 mm
Medial phalanx	26.33 ± 3.00 mm	29.50 mm
Distal phalanx	17.40 ± 1.85 mm	19.50 mm

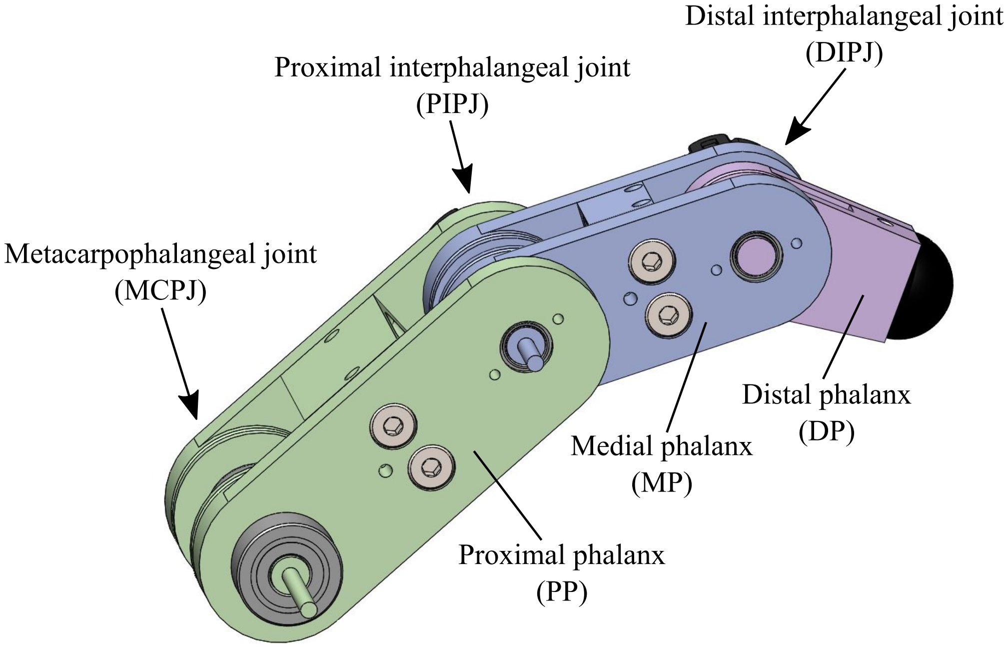

Figure 2.2. CAD Model of Finger

The links are designed to accommodate a range of motion of at least 90° clockwise and 90° counterclockwise at each joint. The custom link components are machined from 7075 aluminum for high strength and low weight. A threaded hole is provided at the tip of the distal link and a hemispherical nitrile rubber bumper is attached to increase friction and enable the manipulation of objects with a 3-finger hand.

Each link is composed of several components as illustrated in Figure 2.3 in which the first link (PP) is isolated. This design ensures that the pulleys at which the tendons are terminated are fully machinable, allows the tendons to pass through each link, and reduces the overall weight of the finger. On the PP and MP segments, the blocks connecting the two sides of each link contain channels through which the cables pass to the next joints. The finger is designed such that bearings in the housing support the PP, bearings in the PP support the MP, and bearings in the MP support the DP. The shaft diameters on either side of each link at the joint are stepped down to accommodate the selected encoders.

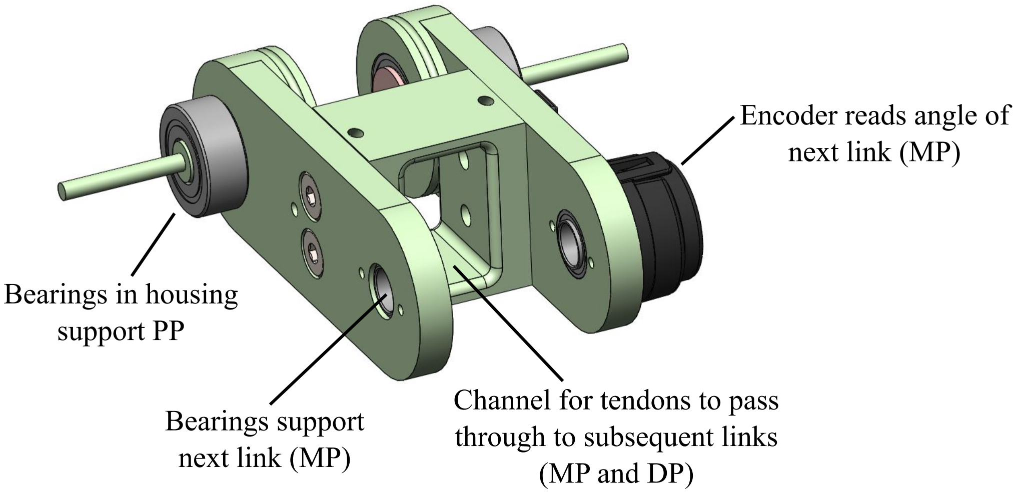

Figure 2.3. CAD Model of First Link (Proximal Phalanx)

2.3 Transmission

The power transmission between the VSAs and the joints must withstand the required tendon forces, maintain the stability of the finger, and enable independent joint actuation.

In this thesis, the joints are designed such that the cable passes through the center of each joint, allowing subsequent joints to rotate independently from previous joints. Two dowel pins between coupled, freely rotating disks constrain the ca-

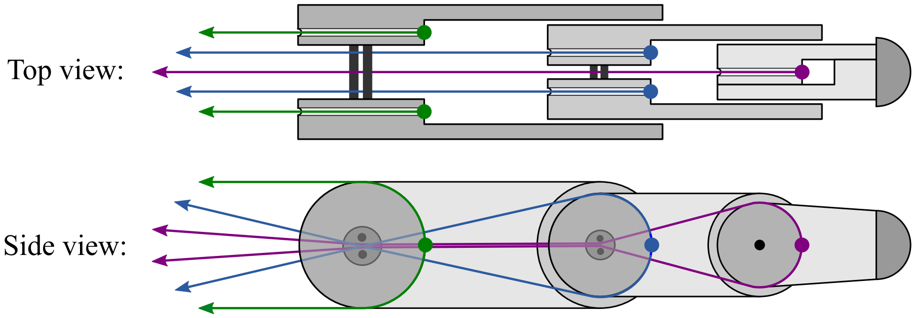

Figure 2.4. Top and Side View of Finger Tendon Routing

bles to the center of the more proximal joints, as the space between the pins is only slightly larger than the diameter of each tendon. This design is illustrated in the top view of the finger in Figure 2.4 and demonstrated in the annotated CAD model in Figure 2.5. A photo of a joint is provided in Appendix A. This routing method minimizes the radius of curvature of the tendon as it passes through a joint, providing the finger with nearly point-to-point cable routing. In other words, as the joint configuration changes, the length of the tendon in the finger remains virtually constant, thereby minimizing actuation coupling between joints.

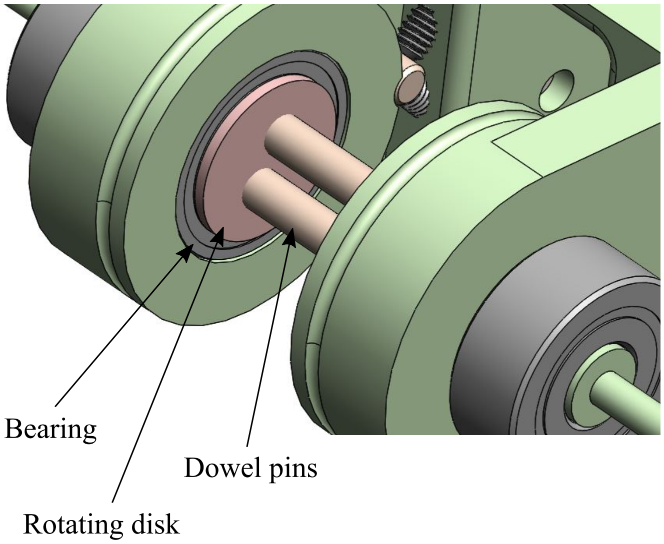

Figure 2.5. Joint Center Tendon Routing Method

To ensure that the force of the cable remains centered along the finger and to minimize undesirable out-of-plane moments in the system, the distal link is only controlled by one central pulley, but the medial and proximal links each have two pulleys to load the joint symmetrically while allowing the cables of more distal joints to pass between them. This layout is illustrated in the top view of the finger in Figure 2.4. The differential mechanisms enabling this tendon routing are identified and explained in the following subsection. A short dowel pin and a pair of set screws act as a clamp to secure the cable to the pulleys at each link. Tendon anchoring techniques are described in Appendix A.

2.3.1 Differential Mechanism Design

Two central tendons control the DIPJ (one each for flexion and extension), while four tendons from inline differential mechanisms control the PIPJ and MCPJ, respectively. Differential mechanisms transmit tendon forces from the VSA mechanisms to symmetrically load the MCPJ and PIPJ using the method illustrated in Figure 2.6. This symmetrical loading ensures that forces remain centered along the length of the finger and do not introduce unwanted moments in the system.

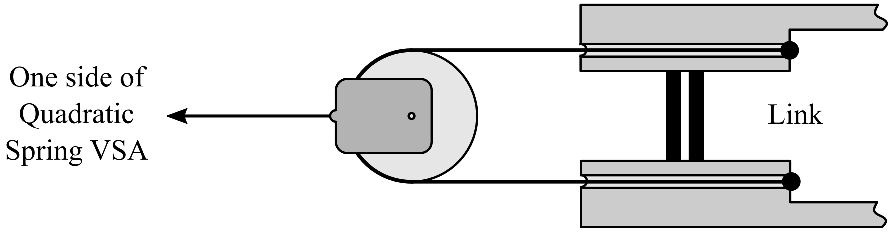

Figure 2.6. Differential Mechanism Layout (Top View)

2.3.2 Tendon Material

PowerPro® Super 8 Slick V2™ fishing line with an 80 lb break strength was selected for actuation of the joints. The line is made with braided ultra-high-molecular-weight polyethylene Spectra® fiber and is manufactured using a resin coating process that permeates the whole line rather than coating only the surface. This manufacturing process reduces the likelihood that the coating will wear off and add friction in the system.

The maximum tension in the cable was set at 100 N (22.5 lb) based on the specified 80 lb break strength and experiments performed with the selected cable (fishing line). This tension value will accommodate a 4-lb load at the end of the fully extended finger.

2.4 Position and Stiffness Control

A National Instruments CompactRIO controller (cRIO-9063) with a 32-Channel Digital I/O Module (NI 9403) is used to control the stepper motors and read quadrature rotary encoder values to generate the desired joint positions and stiffnesses.

Programming was done in NI LabVIEW 2020. The angles of the joints and levers are measured by nine encoders, three for each VSA joint. The encoder locations for each VSA joint are indicated in Figure 2.7. The encoder values are used to update joint and lever angles during testing. The relationship between lever angles and joint stiffness is experimentally determined for each joint. Then, with a desired angle and stiffness command for each joint, the six motors in the three VSA joints are moved to achieve the desired finger configuration and joint stiffnesses. The position and stiffness of each joint are also used to calculate the analytical values of the compliance matrix at the endpoint for testing, using Equations 1.6-1.8. The methods used to determine the desired encoder angles are detailed in Chapter 3.

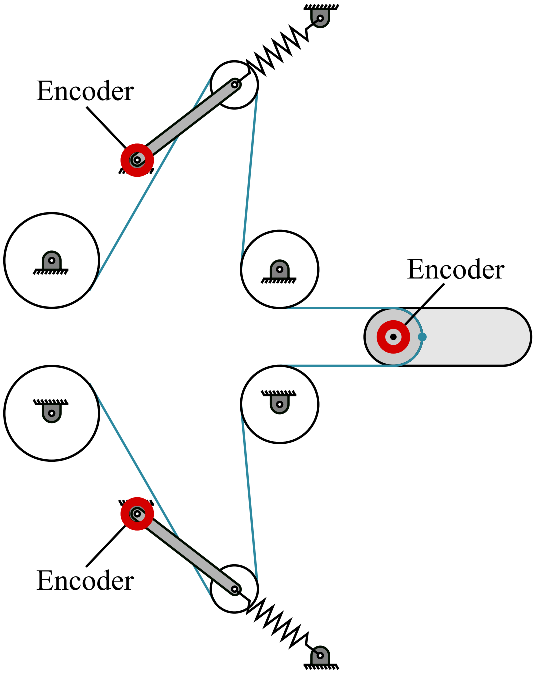

Figure 2.7. Encoder Locations for One Variable Stiffness Joint

2.5 Summary

The finger is designed to maintain structural integrity when loaded by tendon and external forces. The joints are designed such that there is minimal actuation coupling between subsequent joints, allowing the joint positions and stiffnesses to be controlled independently and produce a larger range of stiffness values. This kinematic and elastic decoupling allows the mechanism to produce a large range of endpoint compliance matrices. The design and performance of the antagonistic quadratic spring variable stiffness actuators used in the finger are described in Chapter 3.

CHAPTER 3

ANTAGONISTIC QUADRATIC SPRING MECHANISM DESIGN

The geometric layout of each half of the VSA was determined through the optimization of geometric parameters to produce quadratic force-deflection behavior in each opposing mechanism. The design was optimized to achieve a large range of controllable linear stiffness in each joint, characterized by a high stiffness ratio (ratio of highest to lowest stiffness).

The previously developed antagonistic quadratic spring VSA [2] reaches a maximum stiffness ratio of approximately four. In this thesis work, additional geometric parameters are used to expand the effective design space and achieve a higher stiffness ratio.

This chapter describes the parametric optimization of the mechanism layout to achieve the desired force-deflection relationship. In Section 3.1, the functional design of the mechanism is described and a static analysis of the forces in the system is presented. In Section 3.2, the modeling strategy and optimization methodology are reviewed and the optimal results are presented and discussed. Verification testing of the VSA is presented in Section 3.4.

3.1 Functional Design

To analyze the stiffness-deflection curve for different combinations of design parameter values, the force-deflection curve must first be generated. This analysis involves using the mechanism geometry and spring specifications to calculate the tension in the cable as the motor pulley winds the cable. The change in cable length within the mechanism as the motor pulley rotates and the lever angle changes is illustrated in Figure 3.1.

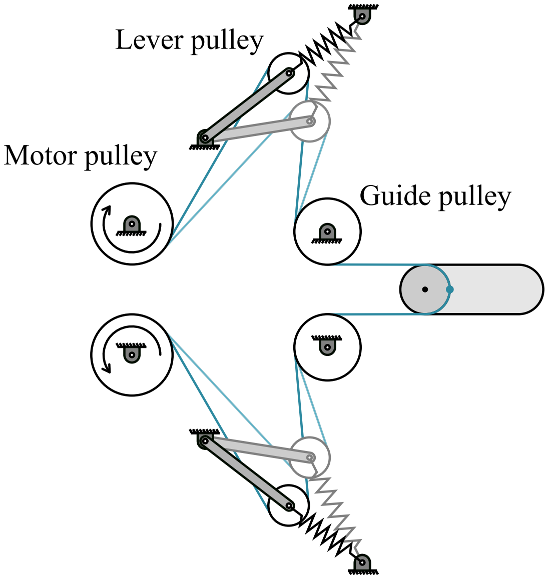

Figure 3.1. Cable Deflection as Lever Angle Changes with Motor Pulley Rotation

In order to achieve a large range of controllable linear stiffness at each joint, each of the two halves of the VSA is a mechanism designed to produce quadratic force-deflection behavior (as described in Section 1.3.1) in the tendons connected to each link. The layout of the mechanism, presented in Figure 3.2, consists of three pulleys through which a tendon is routed. The fixed motor shaft winds the tendon, which shortens the total length of the tendon within the mechanism. This winding of the tendon causes the lever pulley to move along a circular path (with a radius equal to the length of the lever), extending the conventional tension spring and increasing the force in the tendon.

The tension in the cable is calculated using a quasi-static analysis. The forces on the lever pulley that generate moments about the base of the lever are illustrated in Figure 3.3. Bearings are used throughout the system to minimize friction and the compliance of the cable is neglected. As such, because friction is negligible and inertia is not relevant in a static analysis, the tension T is constant along the length of the tendon

$$T = \|T_1\| = \|T_2\|. \tag{3.1}$$

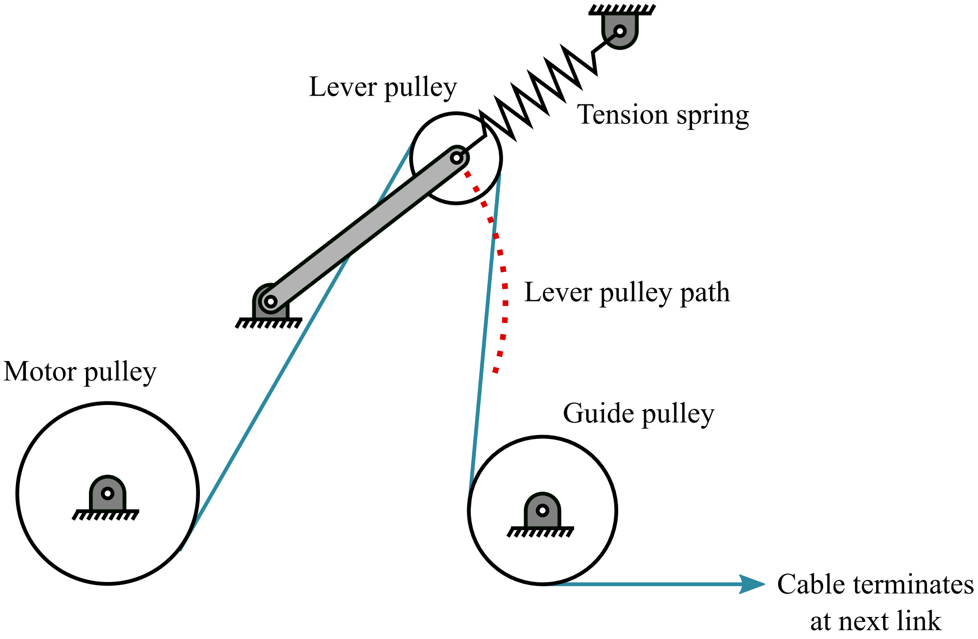

Figure 3.2. Quadratic Spring Mechanism Design

The known force from the spring F_s, based on the spring constant and mechanism geometry, determines the tension in the cable T. The sum of moments about point O in Figure 3.3 is

$$\mathbf{M_O} = ((\mathbf{r}_1 \times \hat{\mathbf{T}}_1) + (\mathbf{r}_2 \times \hat{\mathbf{T}}_2))T + (\mathbf{r}_S \times \mathbf{F}_S) = \mathbf{0}. \tag{3.2}$$

Then, the relationship between the spring force and cable tension to be used in the mechanism analysis is

$$T = \frac{-\hat{\mathbf{z}}^T(\mathbf{r}_S \times \mathbf{F}_S)}{\hat{\mathbf{z}}^T((\mathbf{r}_1 \times \hat{\mathbf{T}}_1) + (\mathbf{r}_2 \times \hat{\mathbf{T}}_2))}. \tag{3.3}$$

The stiffness of the mechanism for each value of cable displacement is calculated as the derivative of cable tension with respect to the cable deflection (dT/dL).

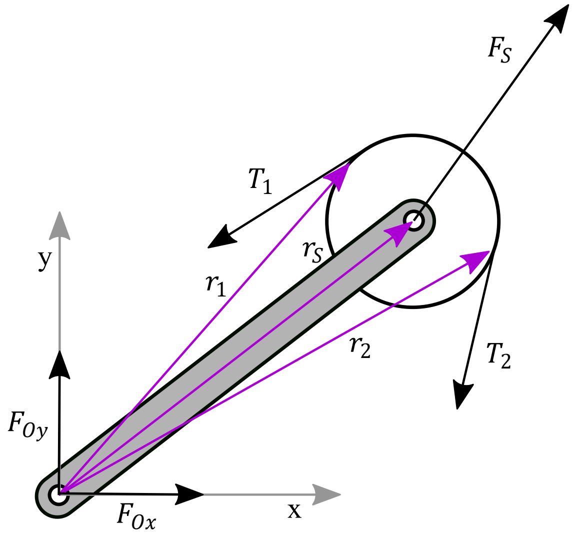

Figure 3.3. Forces Acting on the Lever Pulley

3.2 Parametric Modeling Strategy

Geometric parameters affecting the behavior of the system were optimized to determine the mechanism layout producing the desired quadratic force-deflection behavior. This optimization involves determining a cable deflection range over which the mechanism achieves a wide range of linear stiffness behavior. Altering the mechanism geometry changes the output force-deflection curve of a tendon attached to the link.

3.2.1 Mechanism Design Parameters

A set of parameters defining the physical layout of the VSA were optimized to generate a desirable quadratic force-deflection curve. When selecting an off-the-shelf spring, elastic parameters such as free length, spring constant, and initial spring tension cannot be selected independently. As such, these parameters were not included in the optimization. Instead, hundreds of commercially-available conventional tension springs were evaluated for possible use. Since spring selection determines spring free length, spring constant, and initial spring tension, geometric optimization was performed for each spring in a finite set of promising springs.

The ten geometric and elastic design parameters are shown in Figure 3.4. Each of the optimization parameters are defined in Table 3.1.

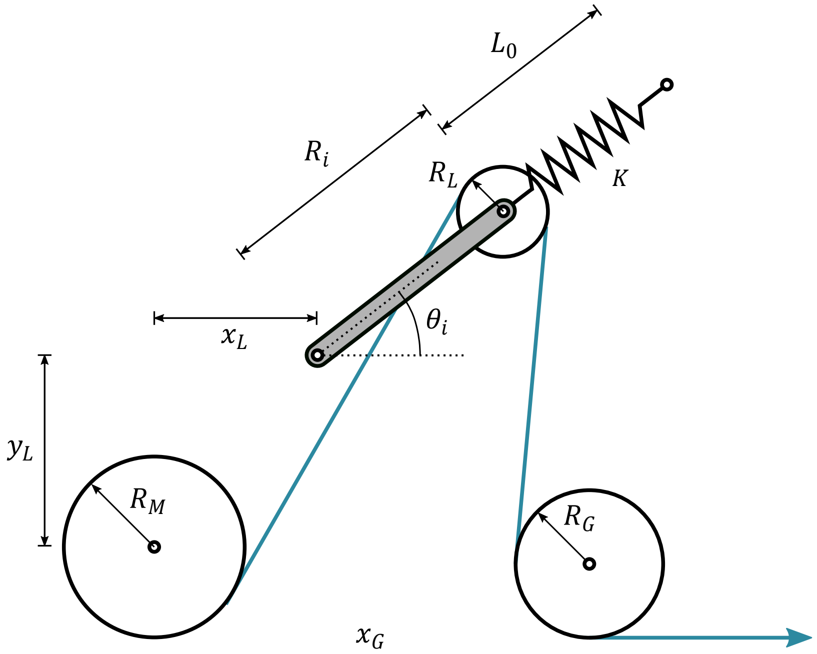

Figure 3.4. Mechanism Design Parameters

Table 3.1. Design Parameters, Descriptions, and Range of Values Considered

Parameter	**Description**	**Lower Bound**	**Upper Bound**
R_i (mm)	distance from the lever base to the center of the lever pulley	15	50
x_G (mm)	distance from the center of the motor pulley to the center of the guide pulley in the x-direction	30	70
x_L (mm)	distance from the center of the motor pulley to the lever base in the x-direction	-25	25
y_L (mm)	distance from the center of the motor pulley to the lever base in the y-direction	0	25
θ_i (°)	initial angle from the x-axis to the line connecting the lever base to the center of the lever pulley	30	60
R_M (mm)	radius of the motor pulley	3	12
R_L (mm)	radius of the lever pulley	3	12
R_G (mm)	radius of the guide pulley	3	12
L_0 (mm)	initial spring length	discrete values	
K (N/mm)	spring constant	discrete values	

3.2.2 Spring Selection

Conventional extension springs require an initial tension force to separate adjacent coils. This concept is illustrated in Figure 3.5. So, the stiffness at each joint, while theoretically zero for an ideal linear spring, will not display zero stiffness at the initial mechanism position with an off-the-shelf extension spring.

As previously stated, many off-the-shelf springs were investigated as part of the optimization of the lever mechanism. Rather than optimizing each spring parameter individually, a full evaluation was performed using the specifications from

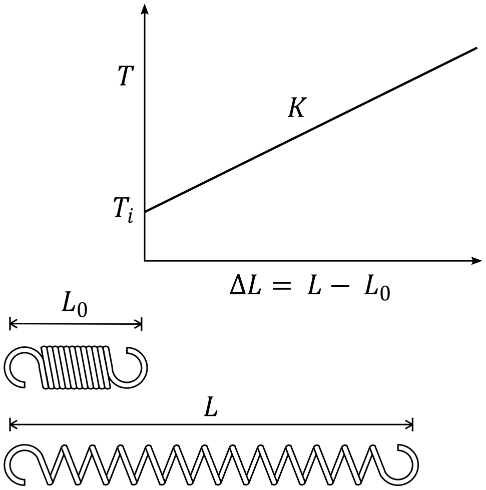

Figure 3.5. Initial Tension in Extension Springs

each of these existing springs. This evaluation method ensures that the optimal mechanism layout does not require an expensive, custom extension spring.

A large catalog of springs was considered. Spring performance criteria in spring selection include: 1) high allowable strain, 2) low initial tension value, and 3) short length (so the mechanism remains relatively compact). Springs with lower spring rates typically have a higher allowable strain, so it is beneficial to use multiple low-stiffness springs rather than one stiff spring. A parallel connection of two springs enables the lever pulley to remain centrally located within the housing. The maximum force the mechanism will exert on the spring should not exceed the maximum allowable load specified for the selected spring.

Table 3.2. Set of Considered Springs and Specifications

Specification	**Spring 1**	**Spring 2**	**Spring 3**	**Spring 4**
Rate (N/mm)	1.49	1.19	2.10	1.21
Length (mm)	31.75	38.10	31.75	31.75
Initial Tension (N)	2.65	2.65	3.53	3.14
Maximum Deflection (mm)	17.01	21.59	14.22	25.40

Four springs selected based on these criteria were then evaluated in greater detail. The specifications for these four springs are listed in Table 3.2. Since two springs connected in parallel will be used in the physical realization of the design, the specified spring rate and initial tension are doubled in the numerical evaluation.

3.2.3 Elastic Mechanism Performance

The mechanism layout was optimized to achieve the desired quadratic force-deflection behavior in each half of the VSA by analyzing the behavior of the derivative of the force-deflection curve (stiffness-deflection curve). A quadratic force-deflection curve or a quadratic with a linear component produces a linear derivative, i.e., a controllable linear stiffness in the mechanism. In order to assess the performance of a given mechanism layout, a single objective function value is used to evaluate the elastic mechanism performance, specifically how well a given parametric design produces a large range of stiffnesses over a large range of cable deflection.

Evaluating the change in the length of the cable within the system involves determining the length of cable in contact with the pulleys (segments AB, CD, and EF in Figure 3.6) and the length of cable between the pulleys (segments BC and DE in Figure 3.6). A modified version of the MATLAB function file crosstancirc.m developed in [36] was used in this calculation and is included in Appendix E.7. The tension in the cable is calculated at each evaluated lever angle using the mechanism geometry, spring rate, and spring deflection.

The tension in the cable is calculated beginning at the initial lever position (when G, H, and I are collinear) and the evaluation is terminated when one of the following conditions is satisfied: 1) the maximum allowable spring strain is exceeded, 2) the maximum cable tension is exceeded, or 3) other features in the mechanism geometry interfere with further rotation of the lever. The stiffness-deflection

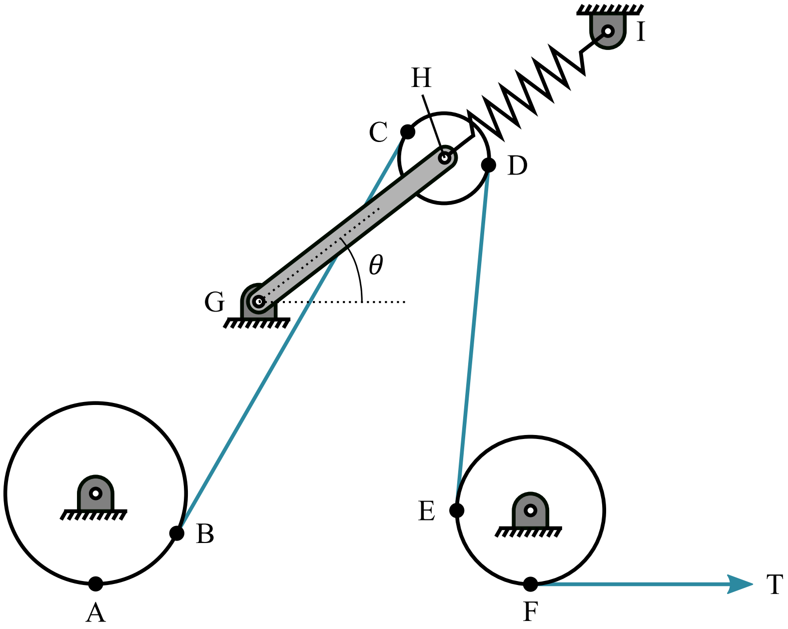

Figure 3.6. Mechanism Points used in Evaluation

curve is then generated, where the stiffness of the mechanism is analyzed as the derivative of cable tension with respect to cable deflection (dT/dL).

In order to determine the quality of a given design, i.e., identify whether the design produces a large range of linear stiffness-deflection behavior, multiple linear regression lines are fit to the simulation data. A bisection method, described below, is used to create a line fit beginning at the initial stiffness K_{min}. The tendon deflection x_i, at which the evaluation terminates, is iteratively selected to ensure that the stiffness curve remains within 10% of the value obtained if the relationship were linear within the range being considered ($0 \leq x \leq x_i$ for each iteration i). The maximum mechanism stiffness K_{max} occurs at tendon deflection x_{lim}, the tendon deflection limit for the final bisection iteration.

A bisection method, outlined in Figure 3.7, iteratively evaluates linear regression results to identify the point at which the stiffness over a range of cable deflection is no longer within 10% of a fit line beginning at K_{min}. The loop begins

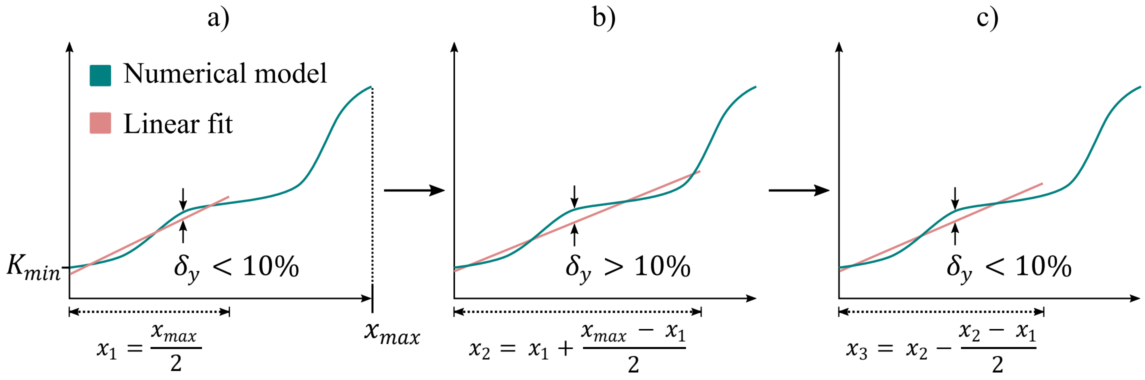

Figure 3.7. Example of the First 3 Iterations of an Iterative Bisection Method

by fitting a regression line to the first half of the stiffness curve (Fig. 3.7(a)). If at any point the numerical model result deviates more than 10% from the line fit value, the evaluation range is halved. If the maximum deviation in the evaluation range is less than 10% of the line fit value, the evaluation range increases by half of the current range (Fig. 3.7(b)). This is repeated 10 times, which was determined to achieve sufficient precision to compare results. An example layout of the results of a numerical evaluation is provided in Figure 3.8, where K_{max} and x_{lim} are found using the iterative bisection method. Within the optimization, the initial evaluation range is decreased (e.g., the lower bound is increased) for this analysis to allow the linear regression to capture a greater portion of the stiffness curve. If the analysis is performed beginning from the stiffness at zero deflection K_0, the low initial slope of the analytical stiffness curve results in a high ($> 10\%$) deviation from the regression line at the y-intercept. The initial stiffness K_{min} is instead calculated at a low, nonzero cable deflection value where $K_{min} = 1.1K_0$. It can also be seen that the mechanism does not have zero stiffness at zero cable deflection, and this relationship between the initial spring tension and the initial mechanism stiffness is described in Appendix B.

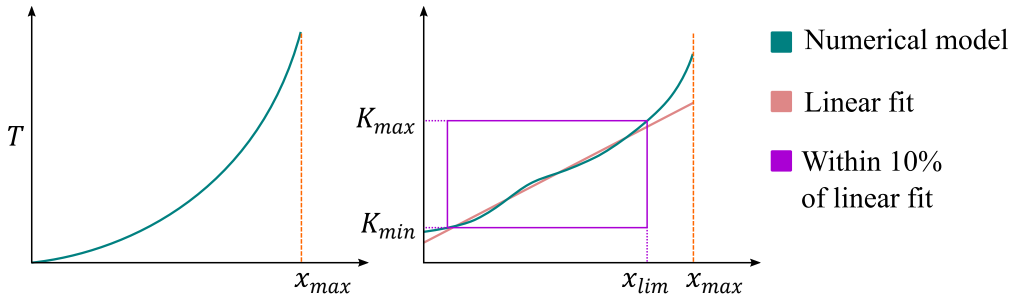

Figure 3.8. Example of Numerical Evaluation Results

3.2.4 Mechanism Optimization

In MATLAB, a genetic algorithm optimization was used to optimize the eight continuously variable geometric parameters to determine the optimal design using each spring within the upper and lower bounds for each parameter. Thus, the optimization problem is given as

$$\begin{aligned} \text{minimize} \quad & f(\boldsymbol{\theta}) \\ \text{subject to} \quad & \boldsymbol{\theta}^{\mathbf{l}} \leq \boldsymbol{\theta} \leq \boldsymbol{\theta}^{\mathbf{u}} \end{aligned}$$

where $\boldsymbol{\theta}$ is the optimization parameter vector. $\boldsymbol{\theta}^{\mathbf{l}}$ and $\boldsymbol{\theta}^{\mathbf{u}}$ are the lower and upper bounds of the parameters, respectively, as listed in Table 3.3. During the optimization, the objective function value is minimized in order to identify a mechanism layout capable of achieving a large stiffness range, given by the ratio of highest to lowest stiffness K_{max}/K_{min}, over a large range of linear stiffness-deflection behavior x_{lim}. Because MATLAB minimizes the objective function, these components become K_{min}/K_{max} and $1/x_{lim}$ in order to maximize the inverse values. Then, the maximum stiffness K_{max} is multiplied again to increase the overall mechanism stiffness magnitude as the optimization, without squaring K_{max}, resulted in low joint stiffness values. These low stiffness values would make testing difficult and reduce the allowable load at the endpoint. The objective function used to determine the optimal mechanism configuration is given by

$$f = \left(\frac{1}{K_{max}}\right)\left(\frac{K_{min}}{K_{max}}\right)\left(\frac{1}{x_{lim}}\right) = \frac{K_{min}}{K_{max}^2 x_{lim}} \tag{3.4}$$

where x_{max} is the longest cable deflection range, K_{max} is the maximum stiffness, and K_{min} is the minimum stiffness. A genetic algorithm technique was chosen over a gradient search method due to the highly nonlinear behavior of the design space.

A genetic algorithm seeks the minimum of a specified objective function by a natural selection process. The genetic algorithm first generates a random population of initial design parameter values. The objective function value is evaluated for each individual within the population. A new population is then generated from the current population. This generation involves retaining individuals with good objective function values (elite), then combining portions of parameter descriptions from two individuals (crossover), and making random changes to individual points (mutation). The process repeats until a termination condition is reached, such as maximum number of generations exceeded or no improvement in the performance of the best individual in several generations.

A single genetic algorithm optimization run yields a mechanism parametric design with the lowest objective function value. The optimization was performed several times for each of the four springs in order to select a spring. Then, given the selected spring, the optimization was run several times with a larger population to determine the optimal mechanism layout.

3.2.5 Mechanism Optimization Results

Ultimately, Spring 4 (listed in Table 3.2) was selected, as it consistently produced the best objective function values. This spring enabled a large range of linear stiffness-deflection behavior in the mechanism while maintaining a maximum cable tension value below the specified limit. The optimized mechanism layout is illustrated in Figure 3.9. The lever, lever pulley, and spring are shown in the initial configuration (lowest stiffness) and the final configuration (maximum stiffness). The optimized parameter values and the actual values used in the prototype are listed in Table 3.3. Some dimensions in the prototype were altered from optimized values for simplification and to allow the use of off-the-shelf pulleys. An evaluation was performed for the selected values to ensure the parameter adjustments were acceptable. The objective function value increased by 3% for the mechanism layout with the actual parameter values, so this was determined to be an acceptable modification.

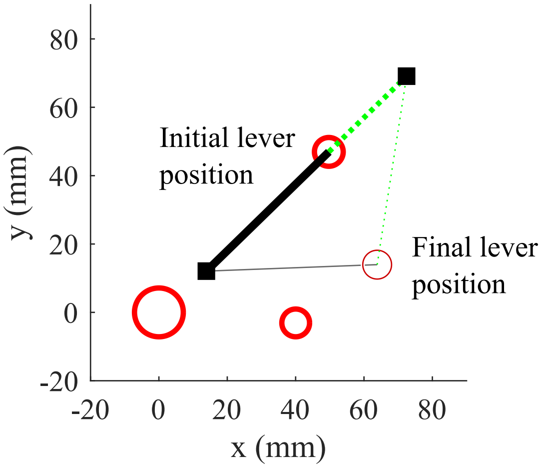

Figure 3.9. Optimized Mechanism Layout shown at Initial Lever Position (minimum stiffness) and Final Lever Position (maximum stiffness)

The optimized force-deflection curve and stiffness-deflection curve are presented in Figure 3.10. The tendon deflection value at which the linear spring reaches

Table 3.3. Optimized Lever Mechanism Parameters

Mechanism Parameter	**Optimization Value**	**Actual Value**
R_i (mm)	50.00	50.00
x_G (mm)	40.02	40.00
x_L (mm)	13.89	13.89
y_L (mm)	12.07	12.07
θ_i (°)	44.24	44.24
R_M (mm)	7.14	7.14
R_L (mm)	4.21	3.97
R_G (mm)	4.06	3.97

its maximum allowable strain is indicated by the vertical dashed line on each graph. The box over the stiffness-deflection curve indicates the region where the calculated stiffness remains within 10% of the linear regression fit line for the given mechanism geometry.

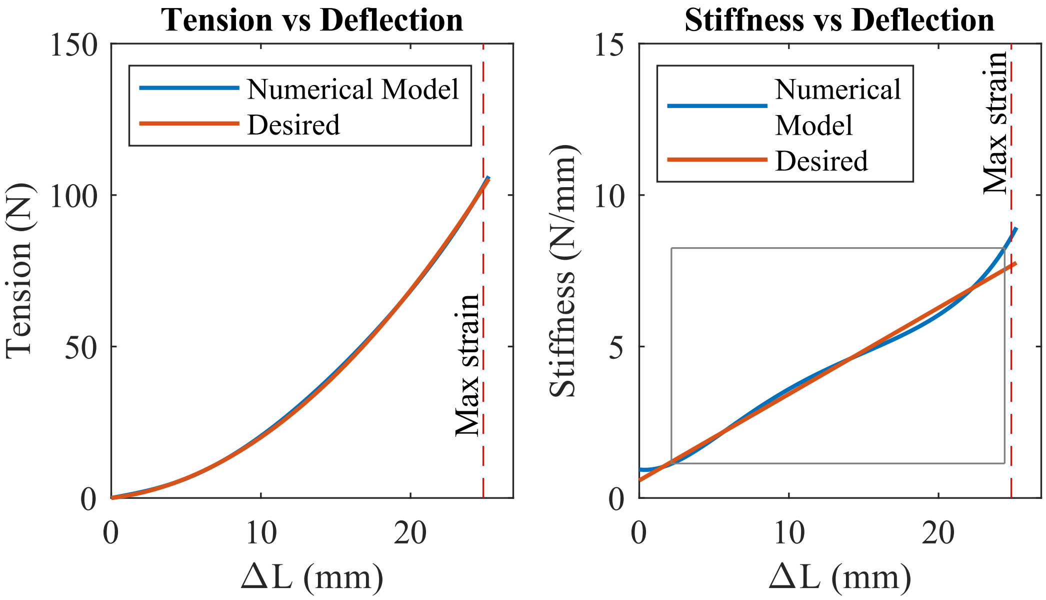

Figure 3.10. Optimized Force-Deflection and Stiffness-Deflection Curves

3.3 Prototype Development

Physical design features of the VSA mechanism and finger are described in Appendix A. The tendon anchoring and tendon routing methods are detailed in Sections A.1 and A.2. Photos of the VSA mechanism and the full prototype are included in Section A.3. The motor selection and design of the motor shaft is described in the following subsection.

3.3.1 Motor Selection and Motor Shaft Design

Six StepperOnline stepper motors with integrated 5.18:1 planetary gearboxes and StepperOnline DM542T digital stepper drivers were selected for actuation of the variable stiffness mechanisms. Although servo motors provide energy savings at low torque and have closed-loop position control, they are more expensive and unnecessary for this application in which the actuators are simply winding a cable. The lever and joint positions are required to control and evaluate the performance of the mechanism, while the motor positions do not provide any necessary information.

The motor shaft attachment for winding the cable was designed separately from the motor pulley of the VSA mechanism, i.e., the motor pulleys (shown with rotation arrows in Figure 3.1) are not concentric with the motor shafts winding the cable. This layout ensured that the diameter of the shaft on the selected motor would not limit the allowable force in the tendon or the range of motor pulley radii during optimization. The motor shaft design is presented in Appendix A, Figure A.3. A photo of a single mechanism is provided in Figure A.7 in which it is clear that the cable winding pulley is not concentric with the VSA motor pulley.

3.4 Quadratic Nonlinear Spring Mechanism Testing

The VSAs and finger were assembled and tested to determine experimental consistency with the expected performance based on numerical evaluation results. The experimental tension-deflection behavior for half of the VSA (a single-sided mechanism) at each joint is presented first. Results of tests performed on the full antagonistic VSA yield the relationship between joint stiffness and lever deflection. These results are used in the following chapter to evaluate the performance of the finger endpoint compliance.

3.4.1 Apparatus

Nine 14-bit micro optical quadrature encoders, located at the levers and joints as shown in Figure 2.7, are used to monitor lever and joint positions throughout testing. Weight hangers are used to apply known forces on the tendons and links. NI LabVIEW is used to display and record lever angles, joint angles, and calculated joint stiffness values.

A homing routine to establish initial encoder positions and a routine for moving the motors to achieve commanded joint positions and stiffnesses were also produced in LabVIEW. Flowcharts which outline these routines and their associated subroutines are included in Appendix C.

3.4.2 Single-Sided Mechanism Testing

First, half of each VSA was tested to produce experimental tension-deflection curves for each joint. These results are then compared to the results obtained from the quasistatic numerical model described in Section 3.2. The tension-deflection behavior is expected to resemble a quadratic function with a linear element.

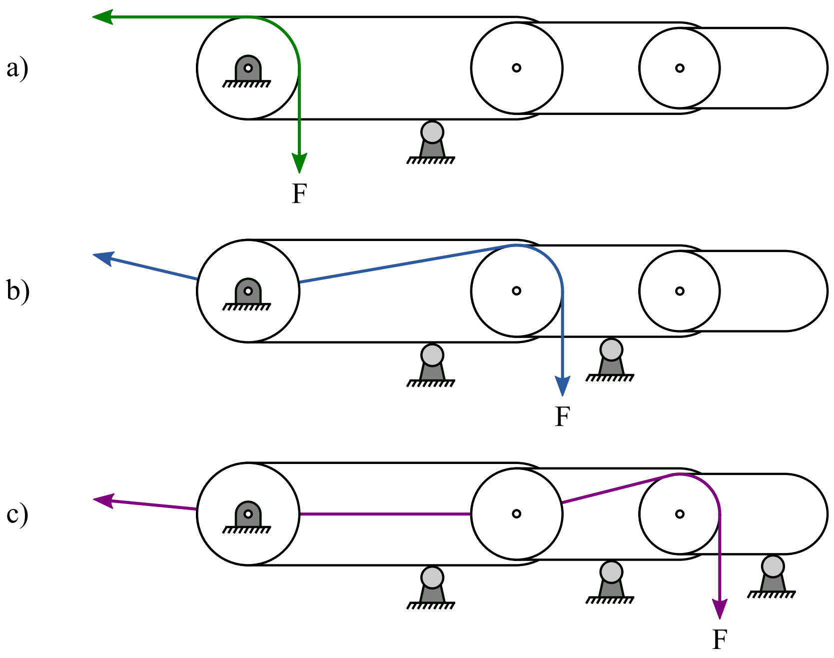

Figure 3.11. Single-Sided VSA Testing Method for a) the MCP joint, b) the PIP joint, and c) the DIP joint

Single-Sided Mechanism Testing Procedure

For the single mechanism testing of each joint, the links were constrained to isolate each joint behavior and to allow the cable to slide freely over the joint pulley as illustrated in Figure 3.11. The top half of the VSA for each joint was tested to determine a force-deflection relationship while the bottom half was left disconnected. The motor positions were fixed and known forces (using weight hangers) were applied directly to the tendon. The lever deflection associated with each tendon load was recorded, up to approximately 90 N of cable tension. Three trials were performed for each joint.

Single-Sided Mechanism Results

The average cable tension vs. lever deflection relationship is shown in Figure 3.12. The numerical model (MATLAB) results are included as well. Due to gravity and manufacturing tolerances, the lever and spring were not exactly collinear with zero cable tension, so the minimum stiffness case is given at 5° lever angles.

The experimental curves appear stiffer than the numerical model, i.e., there is less lever deflection for a given cable tension than expected from the numerical model. The tension-deflection relationships are different between the joints, despite each joint using the same VSA mechanism. These discrepancies may be due to friction in the mechanisms and/or cable routing.

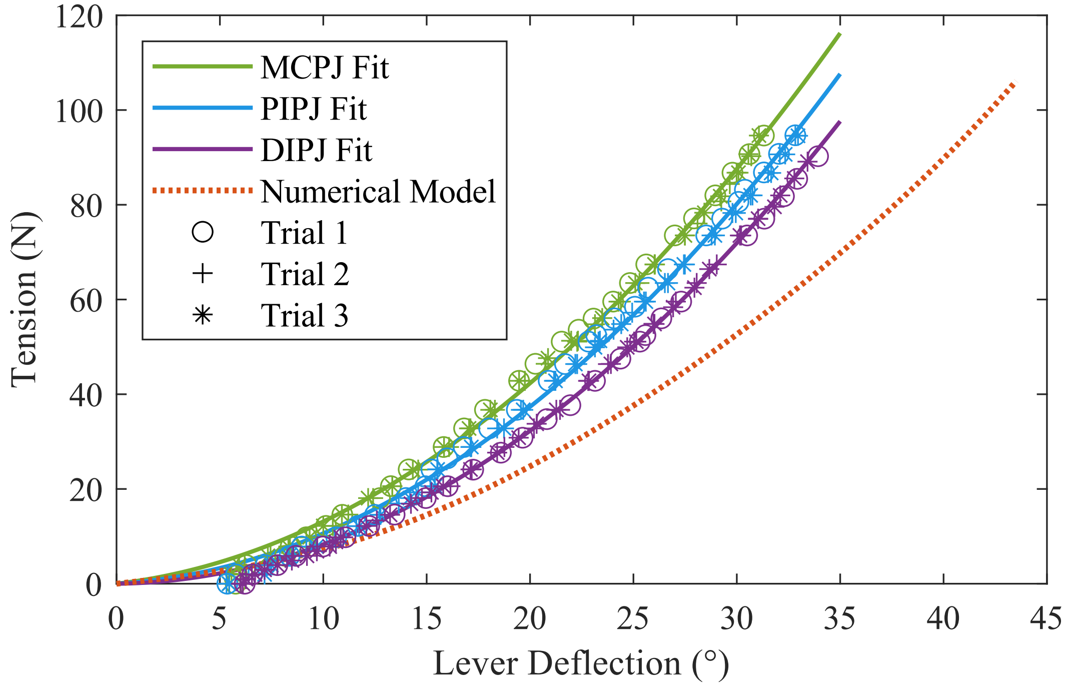

Figure 3.12. Experimental Tension-Deflection Curve for Each VSA and Expected Results Obtained from the Numerical Model

3.4.3 Antagonistic Compliant Actuator Testing

The joints were then tested to evaluate the stiffness-deflection performance of each antagonistic VSA. The relationship between joint stiffness and lever angle is used to achieve commanded joint stiffness values for endpoint testing in the following chapter.

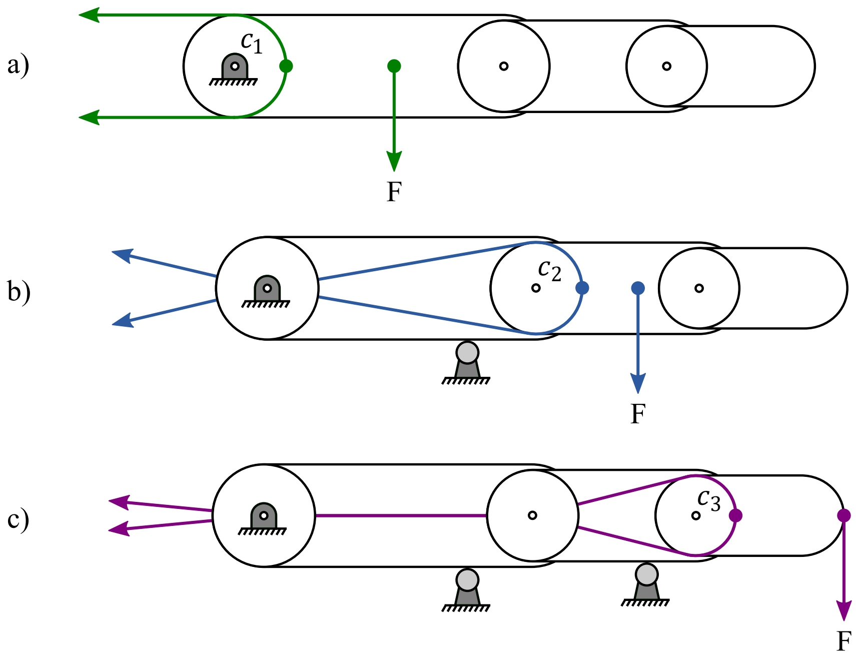

Figure 3.13. Antagonistic VSA Testing Method for a) the MCP joint, b) the PIP joint, and c) the DIP joint

Antagonistic Compliant Actuator Testing Procedure

For the antagonistic testing of each joint, the more proximal links are fixed such that only the joint being tested can change its position. Weights were used to apply different forces at a known distance from the joint as shown in Figure 3.13 to produce a torque-deflection relationship for the joint. This testing was performed at different VSA lever angles, i.e., different joint stiffness values. Three trials were performed for each selected lever angle (5°, 10°, 15°, 20°, 25°, and 30°) at each joint.

The slope of the linear portion of the joint torque-deflection curve at a specified lever angle provides the joint stiffness associated with that lever angle.

Antagonistic Compliant Actuator Results

The average joint stiffness vs. lever angle relationship for each joint is plotted in Figure 3.14 and regression lines are fit to the data. The experimental maximum to minimum stiffness ratio (30° to 5° lever angles) for each joint is 6.05, 5.18, and 5.93 for the MCP joint, PIP joint, and DIP joint, respectively.

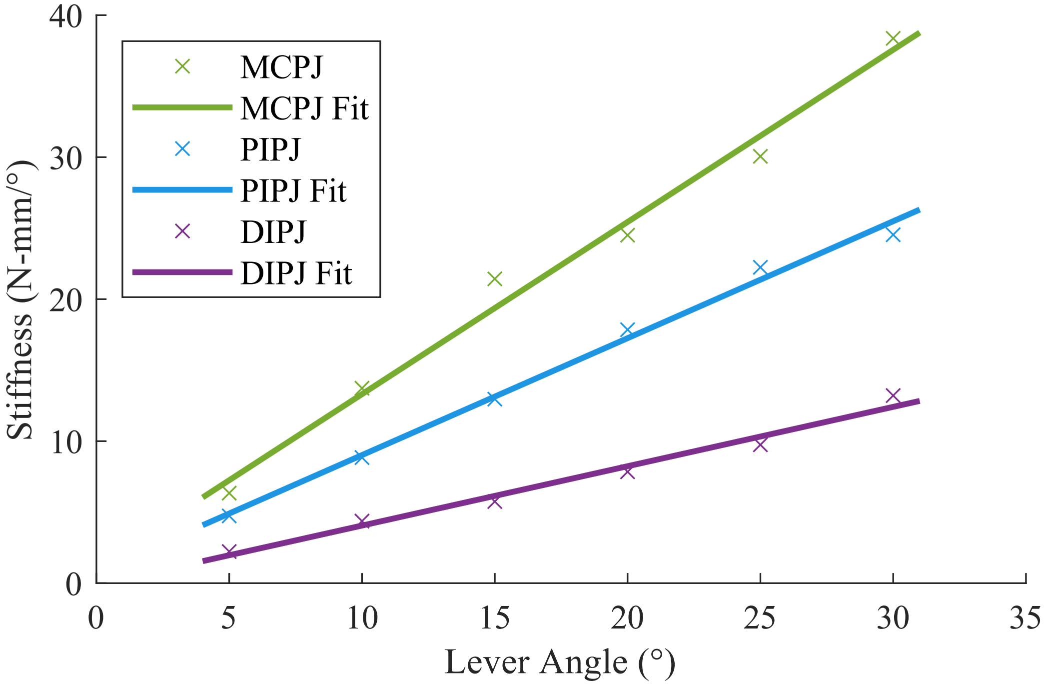

Figure 3.14. Joint Stiffness vs. Lever Angle based on Average Torque-Deflection Linear Regression Slope for Each Joint

The linear regression equations generated for joint stiffness K $(N * mm/^{\circ})$ vs. VSA lever angle θ $(^{\circ})$ at each joint i are

$$K_1 = 1.213 * \theta_1 + 1.187, \tag{3.5}$$

$$K_2 = 0.823 * \theta_2 + 0.788, \tag{3.6}$$

$$K_3 = 0.418 * \theta_3 - 0.122, \tag{3.7}$$

where

$$\theta_i = \frac{\theta_{upper} + \theta_{lower}}{2},\ i = 1,\, 2,\, 3. \tag{3.8}$$

The effective lever angle θ_i for each VSA is calculated as the average of the upper and lower mechanism lever angles. The calculations of joint stiffness K_i from VSA lever angle θ_i for each joint are used within the motor control program in LabVIEW for endpoint testing.

3.5 Summary

The layout of each half of the VSA was optimized to produce quadratic force-deflection behavior in the tendons connected to each link. Eight geometric parameters affecting the behavior of the system were varied to determine a cable deflection range over which the mechanism achieves a wide range of linear stiffness behavior. The VSAs were assembled using the optimized mechanism layout and testing was performed to evaluate the tension-deflection and stiffness-deflection behavior at each joint.

The tension-deflection curves appear precise between the three trials performed at each joint and align well with polynomial fit lines consisting of quadratic and linear components. They are not, however, a good match to the analytical tension-deflection curve generated by the MATLAB numerical model. The mechanisms exhibit less lever deflection than expected at higher tension values, possibly due to friction within the mechanisms or cable routing before reaching the joints. The off-the-shelf springs used in the mechanisms had a specified ±10% tolerance for the spring rate, which may contribute to some discrepancy between the experimental and analytical model for joint torque-deflection behavior.

The ratio of high to low joint stiffness was expected to be approximately 8. Experimentally, the MCP and DIP joints had stiffness ratios of approximately 6, with the PIP joint closer to 5. If testing at 0° lever angles was feasible, the stiffness ratio for each joint would likely increase as the minimum joint stiffness would be lower than the results obtained at 5° lever angles.

Joint stiffness appears to vary linearly with lever deflection, though at different rates for each joint despite having the same design. The slight differences in cable routing between the VSA and finger for each joint likely contributes to this discrepancy. In order to ensure the tendon remained centrally located within the housing and finger, the MCPJ VSA and PIPJ VSA are vertically offset. These tendons pass over low-friction sleeve bearings prior to reaching the finger, and this may introduce more friction in the MCP and PIP joints than the DIP joint, which did not require this cable routing. Further, the tendon material was assumed to have infinite stiffness in the numerical model of the mechanism. However, long stretches of cable may allow the compliance of the tendon material to affect the joint deflection and, as a result, the stiffness measured at the joint. For example, the distance from the DIPJ VSA to the DIP joint is more than twice the distance from the MCPJ VSA to the MCP joint. The relationships presented in the previous section (Equations 3.5-3.7) are used to modulate the stiffness of each joint using the VSA lever angles for endpoint testing in the following chapter.

CHAPTER 4

SYSTEM LEVEL PERFORMANCE EVALUATION

The complete VSA system is shown in Figure 4.1. A polycarbonate housing supports the elastic components of the three VSAs and the base joint of the finger. The geometric layout of each VSA was determined from the optimization outlined in Chapter 3. A 3D printed part was attached to the fingertip to simplify endpoint testing.

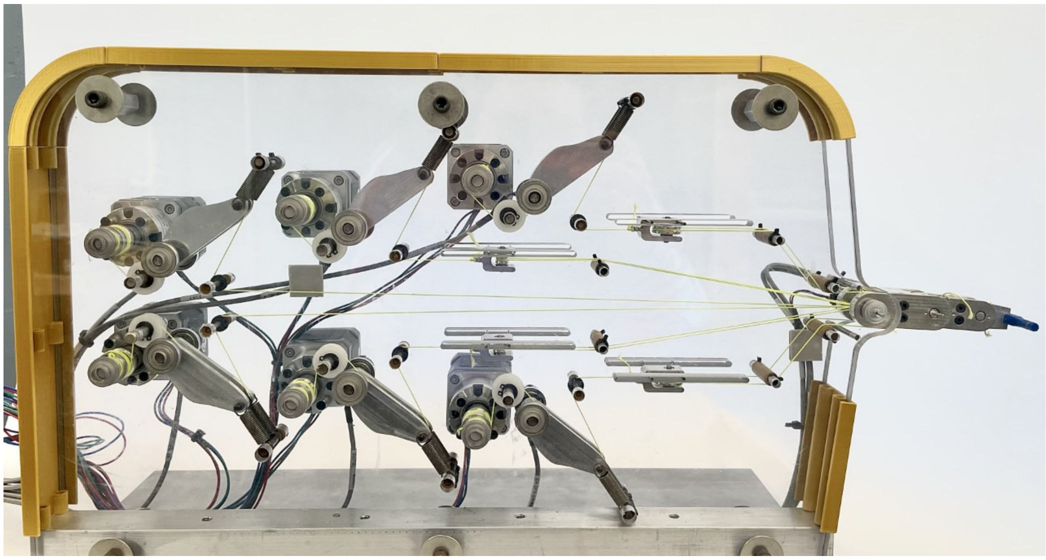

Figure 4.1. Tendon-Driven, Variable Stiffness Finger Prototype

Several finger configurations with different endpoint compliance matrices were tested to assess the performance of the variable stiffness finger. The synthesis procedure presented in [34] was used to determine necessary joint compliance values for each configuration, given the desired joint positions and endpoint compliance matrix. The relationship between lever angle and joint stiffness obtained from the experiments described in Chapter 3 is used to modulate joint stiffness values for endpoint testing. The expected endpoint deflection was calculated for specified force vectors applied to a given configuration as described below. Joint stiffness values are assumed to be constant and link weight is neglected.

4.1 Analytical Fingertip Deflection Model

The load applied at the fingertip generates torque at each joint. The analytical model for expected endpoint deflection calculates the joint torques at a given loading condition and finger configuration. The selected joint stiffness values are used to determine the deflection at each joint in order to generate an expected fingertip deflection.

The antagonistic mechanism testing performed in Chapter 3 revealed that the torque-deflection curves did not necessarily display linear behavior beginning at zero torque. In most cases, the joints exhibited linear torque-deflection behavior after a period of nonlinearity, during which the effective joint torque did not exceed the torque due to friction within the mechanisms and joints. This friction resulted in small joint deflection values for the initial portion of the joint torque-deflection curves, as shown in the joint torque-deflection results in Appendix D. In order to develop a more accurate model to compare with experimental fingertip deflection results, a constant friction torque was included in the calculated torque value for each joint.

The relationship between joint stiffness and friction torque was generated as a second-order polynomial fit of the y-intercepts of the linear torque vs. deflection fit lines for each joint. In other words, the initial vertical offset of the linear fit lines for a given joint in Appendix D, Figures D.1-D.6 were used to determine the initial torque (due to friction) for that joint at a given lever angle, i.e., a given joint stiffness. This relationship is presented in Figure 4.2. The torque due to friction generally increases as the joint stiffness increases, and this effect is greatest in the MCP

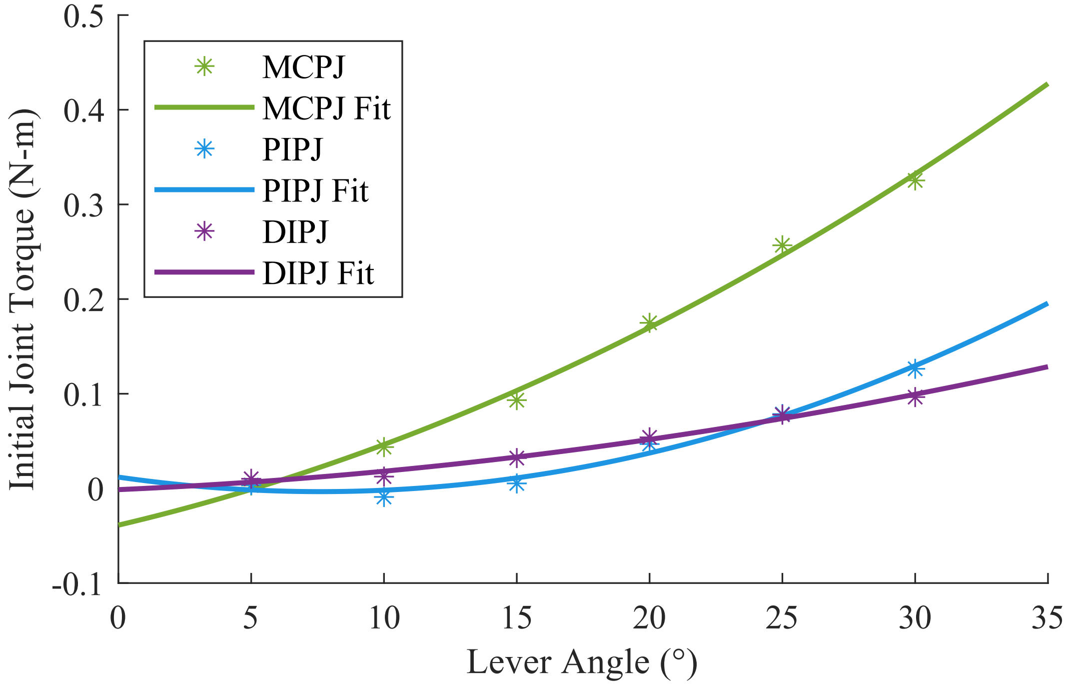

Figure 4.2. Initial Torque (Fit Line Y-Intercepts from Figures D.1-D.6) at Each Lever Angle for Each Joint

joint. Validation tests were performed to achieve a more accurate friction model. The procedure and results from this testing are included in Appendix D, Section D.2. For each joint at each configuration, the higher of the two corresponding friction torque estimates is used to adjust the calculated joint torque.

Theoretical endpoint deflection for each load at each finger configuration was calculated by determining joint torques and using known joint compliances. First, joint torques are calculated by

$$\boldsymbol{\tau} = \mathbf{J}^{\mathbf{T}}\mathbf{f} - \boldsymbol{\tau_f} \tag{4.1}$$

where $\boldsymbol{\tau}$ is the joint torque vector, $\mathbf{J}$ is the 3x3 endpoint Jacobian matrix, $\mathbf{f}$ is the 3x1 force vector applied at the endpoint, and $\boldsymbol{\tau_f}$ is the joint torque vector due to friction. The joint torque that must be applied to allow joint motion generally increases with the stiffness at the joint, as shown by the initial torque values from the antagonistic actuator testing.

Then, the angular deflection at each joint due to the calculated torque values is

$$\mathbf{\Delta q} = \mathbf{c_q}\boldsymbol{\tau} \tag{4.2}$$

where $\mathbf{\Delta q}$ is a vector of joint deflection values and $\mathbf{c_q}$ is the diagonal matrix of joint compliances. Analytical Cartesian endpoint deflection is then obtained using the calculated joint angle deflection for each load at each finger configuration. While joint stiffnesses are assumed to be constant for each configuration, the expected endpoint deflection is calculated incrementally using Equations 4.1 and 4.2 to model the endpoint behavior as weight is applied. For example, the initial finger configuration would yield a Jacobian matrix $\mathbf{J_1}$. A load (e.g., 100g) applied to the endpoint would then result in joint deflections $\mathbf{\Delta q_1}$. The new finger configuration, accounting for these joint deflection values, would yield $\mathbf{J_2}$ at which a higher load (e.g., 200g) will be applied, resulting in $\mathbf{\Delta q_2}$, and so on.

4.2 Finger Endpoint Testing

Four joint configurations were selected at which a combination of possible joint stiffnesses could generate a desired endpoint compliance matrix. These configurations were achieved by commanding the necessary joint stiffnesses and angles in LabVIEW and running the joint configuration routine illustrated in Section C.2 of Appendix C, Figures C.4-C.6.

4.2.1 Apparatus

Aluminum extrusions and a movable pulley were used to create a fixture, illustrated in Figure 4.3, allowing the fingertip to be loaded at angles between -120° and 120° in 30° increments. Weight hangers were used to provide a known load and the joint angles provided by the encoders were used to determine Cartesian endpoint deflection. A photo of this test setup is included in Appendix A, Figure A.8.

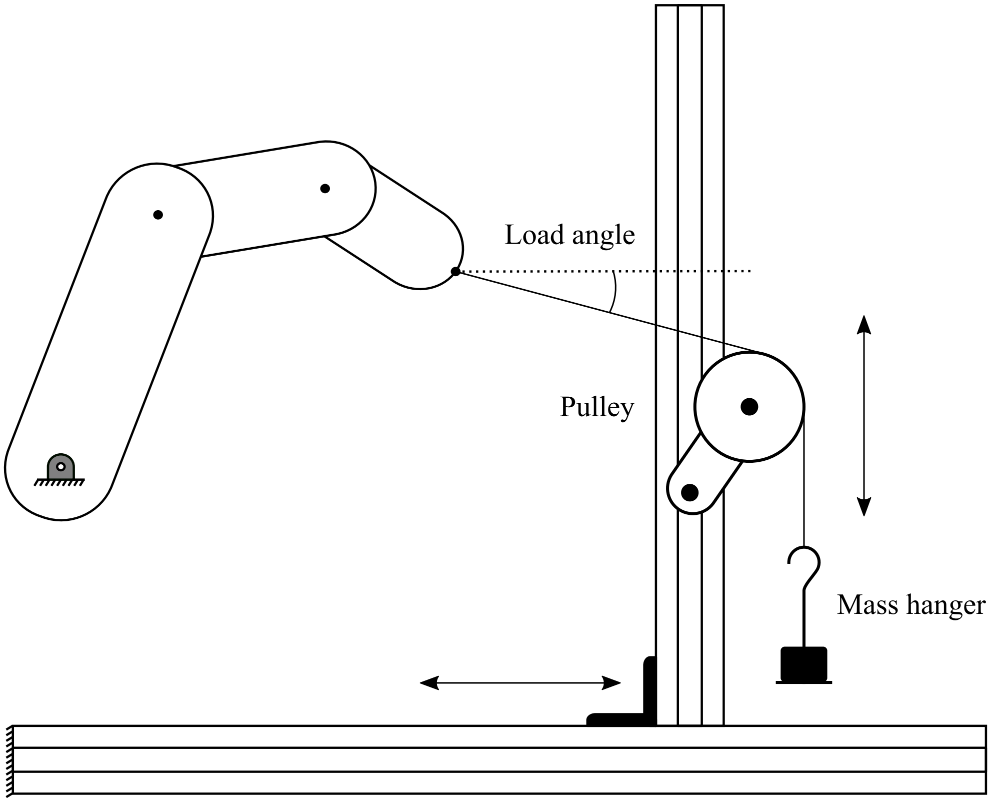

Figure 4.3. Endpoint Testing Apparatus

4.2.2 Finger Endpoint Testing Procedure

A known force was applied to the endpoint in several directions (as in Figure 4.4) to produce a partial planar compliance ellipse, or a graphical representation of endpoint deflection due to a specified load in all directions. There was not interference as shown for the 120° case since the load was symmetrically applied to either side of the finger. Five partial compliance ellipses were generated for each configuration, corresponding to loads of 100, 200, 300, 400, and 500 grams. While the numerical model would most closely match the experimental behavior for infinitesimal fingertip deflection, at which the joint angles and stiffnesses are closest to their commanded values, friction (within the mechanisms, joints, and/or cable routing) dominates the compliant behavior at small joint deflections. As the joints deflect, the finger configuration will change, and therefore the endpoint compliant behavior will also change. Despite this expected discrepancy between the commanded and experimental endpoint stiffness, higher loads are applied during endpoint testing to achieve greater deflection values in order to ensure the experimental results are more reflective of the compliant behavior of the finger, rather than limited by friction within the system.

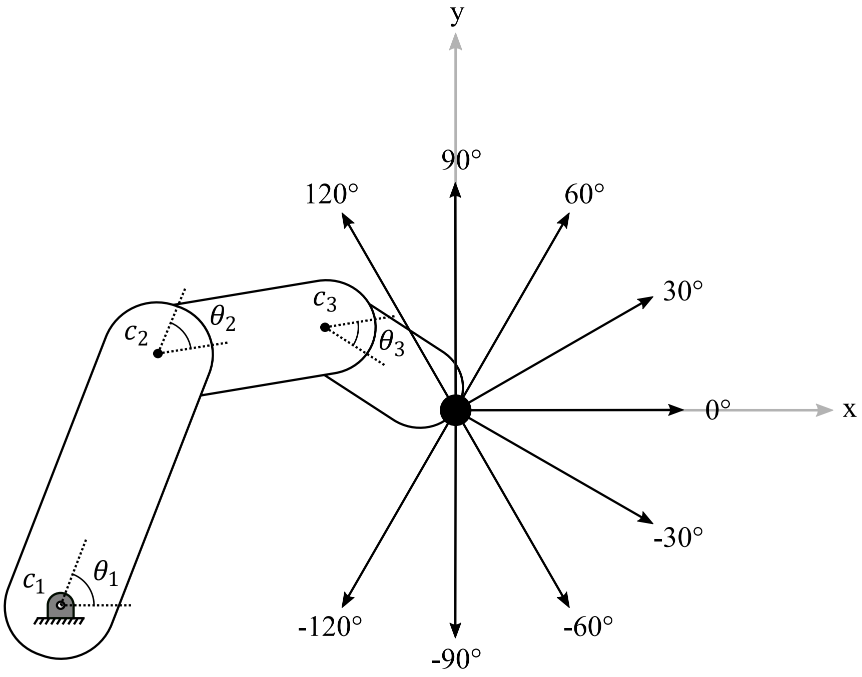

Figure 4.4. Endpoint Testing Method

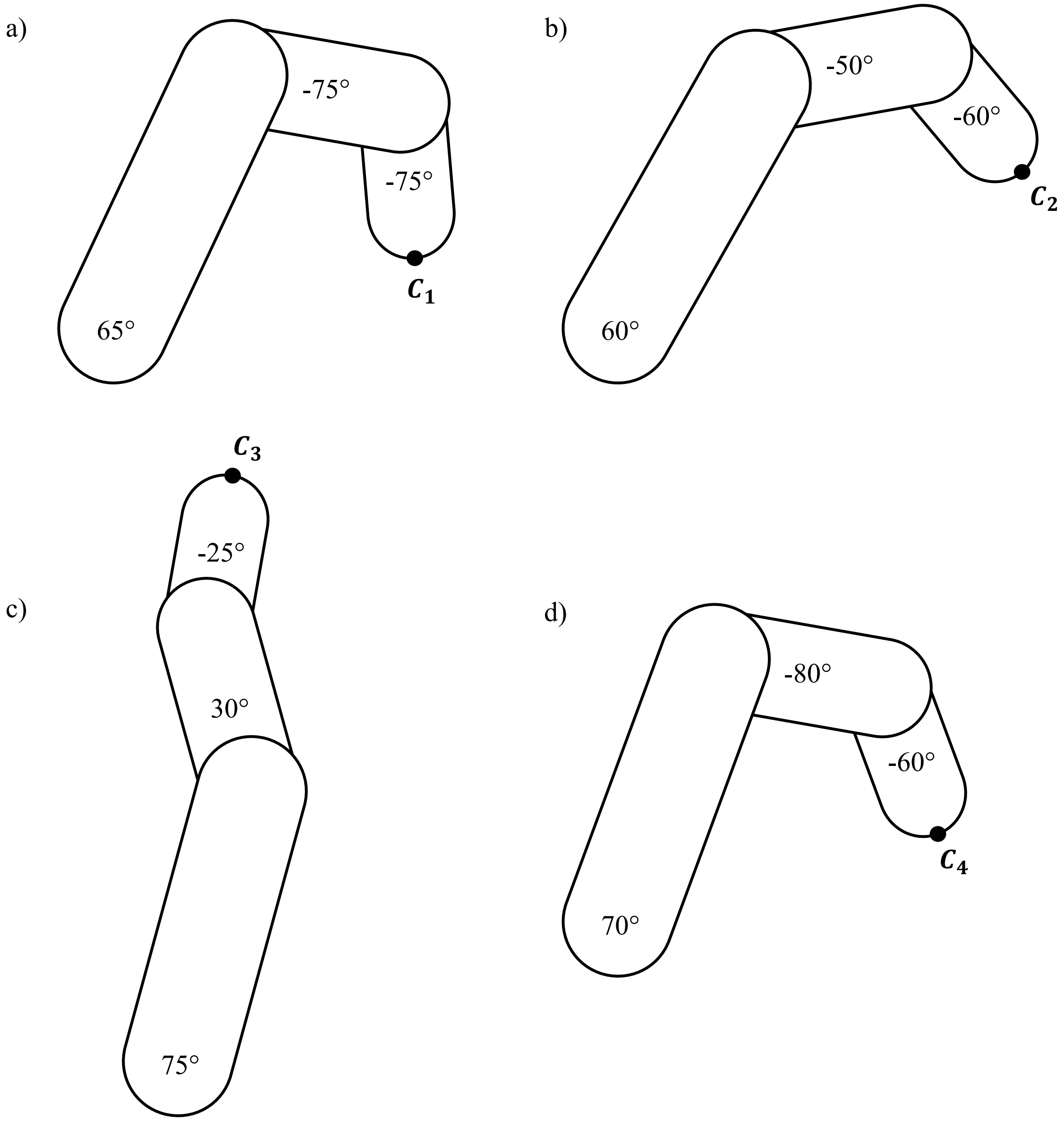

Figure 4.5. Finger Orientation for Compliance Matrices

The planar compliance matrices associated with the finger configurations illustrated in Figure 4.5 are:

$$\mathbf{C_1} = \begin{bmatrix} 4 & 0 \\ 0 & 4 \end{bmatrix} mm/N, \tag{4.3}$$

$$\mathbf{C_2} = \begin{bmatrix} 3 & 0 \\ 0 & 12 \end{bmatrix} mm/N, \tag{4.4}$$

$$\mathbf{C_3} = \begin{bmatrix} 15 & 0 \\ 0 & 0.15 \end{bmatrix} mm/N, \tag{4.5}$$

$$\mathbf{C_4} = \begin{bmatrix} 3 & 2 \\ 2 & 6 \end{bmatrix} mm/N. \tag{4.6}$$

These are not the compliance matrices to which the experimental endpoint performance is compared, as these could be expected only in ideal conditions (e.g., frictionless mechanisms, massless links, and zero manufacturing tolerances). They are instead nominal reference matrices used to calculate the desired joint angles and stiffnesses for each configuration being tested. In other words, the joint stiffnesses are calculated to achieve a given compliance matrix at a given configuration, assuming ideal conditions. These four matrices were selected in order to test the performance of the finger for several configurations with different expected endpoint compliance behaviors. The commanded joint angles and their associated stiffness values to achieve each finger configuration are presented in Table 4.1. Then, the experimental results are compared to an endpoint deflection model which includes analytical friction values to better represent the expected behavior at the fingertip.

4.2.3 Finger Endpoint Testing Results

The experimental endpoint deflection is compared to analytical (expected) endpoint deflection in order to evaluate the performance of the finger in achieving a range of endpoint compliance behavior.

Table 4.1. Commanded Joint Angles and Stiffnesses for Each Configuration

	Configurations			
	a	**b**	**c**	**d**
Joint Angle (°)				
MCPJ	65	60	75	70
PIPJ	-75	-50	30	-80
DIPJ	-75	-60	-25	-60
Joint Stiffness (N-mm/°)				
MCPJ	15.74	14.46	28.53	23.32
PIPJ	15.45	5.28	4.88	5.87
DIPJ	2.25	2.71	4.63	5.94

The analytical and experimental endpoint deflection results are given for configurations (a), (b), (c), and (d) in Figures 4.6, 4.8, 4.10, and 4.12, respectively. These include the endpoint deflection in each direction for loads of 100, 200, 300, 400, and 500 grams. For a direct comparison, the analytical and experimental endpoint deflection for the 300 gram load case are given on the same plot for configurations (a), (b), (c), and (d) in Figures 4.7, 4.9, 4.11, and 4.13, respectively. It should be noted that configuration (c) only allowed loading in directions from -90° to 90° due to the endpoint position relative to the housing. Configurations (a), (b), and (d) were loaded from -120° to 120° as illustrated in Figure 4.4.

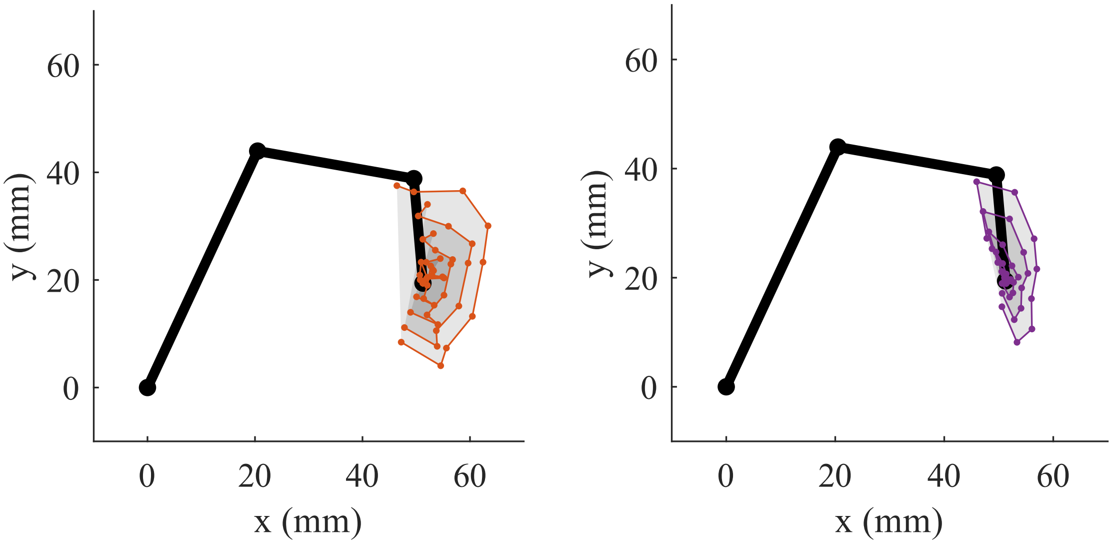

Figure 4.6. Analytical Endpoint Deflection (left) and Experimental Endpoint Deflection (right) for Configuration (a) at 100, 200, 300, 400, and 500 gram loads

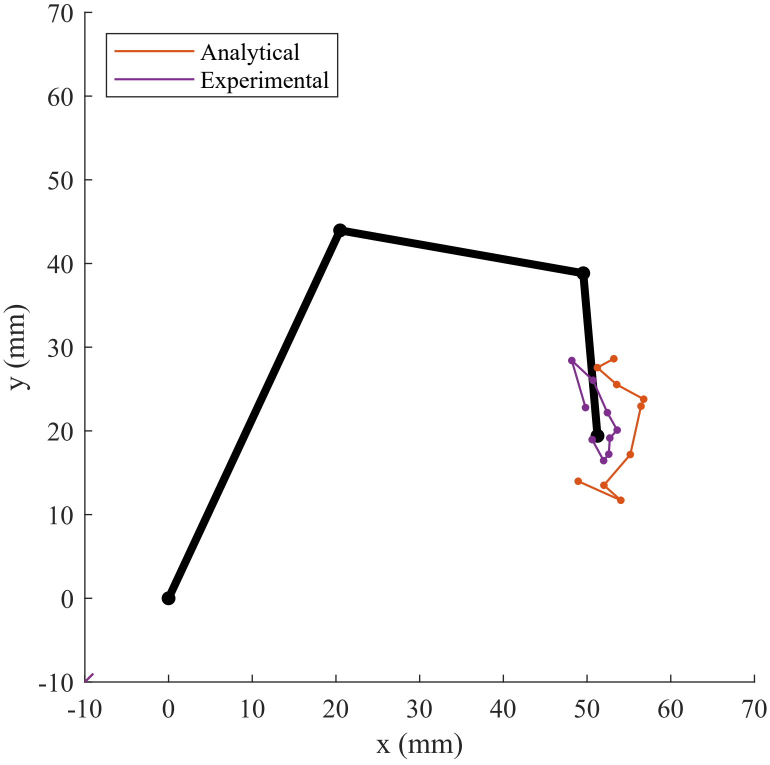

Figure 4.7. Analytical and Experimental Endpoint Deflection for Configuration (a) at 300 gram Load

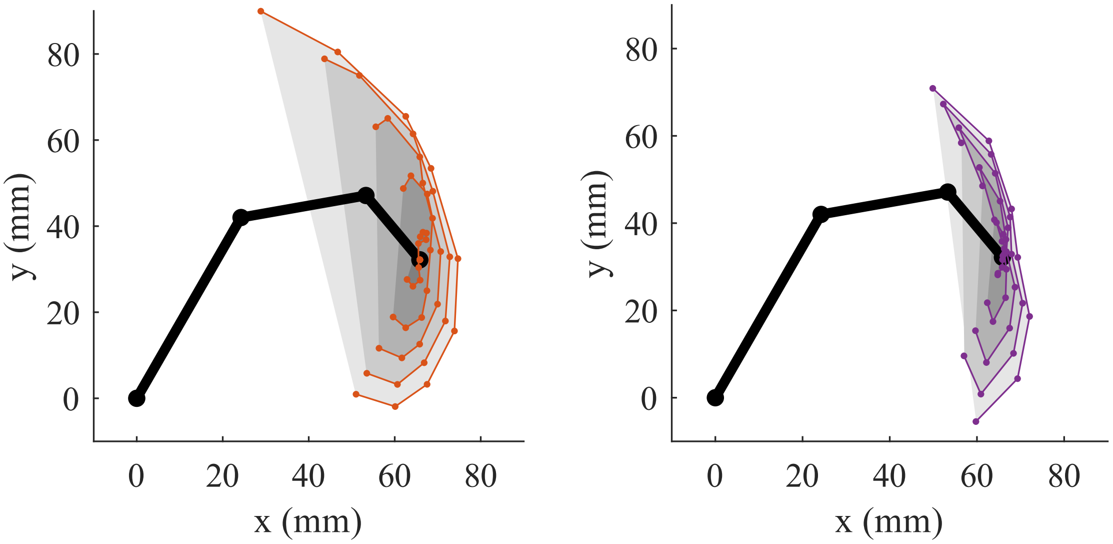

Figure 4.8. Analytical Endpoint Deflection (left) and Experimental Endpoint Deflection (right) for Configuration (b) at 100, 200, 300, 400, and 500 gram loads

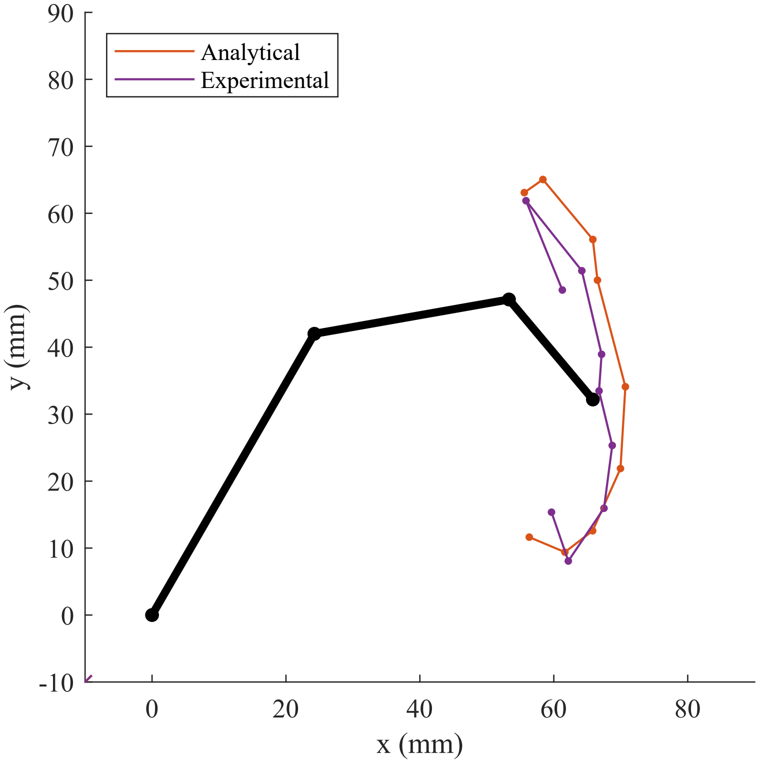

Figure 4.9. Analytical and Experimental Endpoint Deflection for Configuration (b) at 300g Load

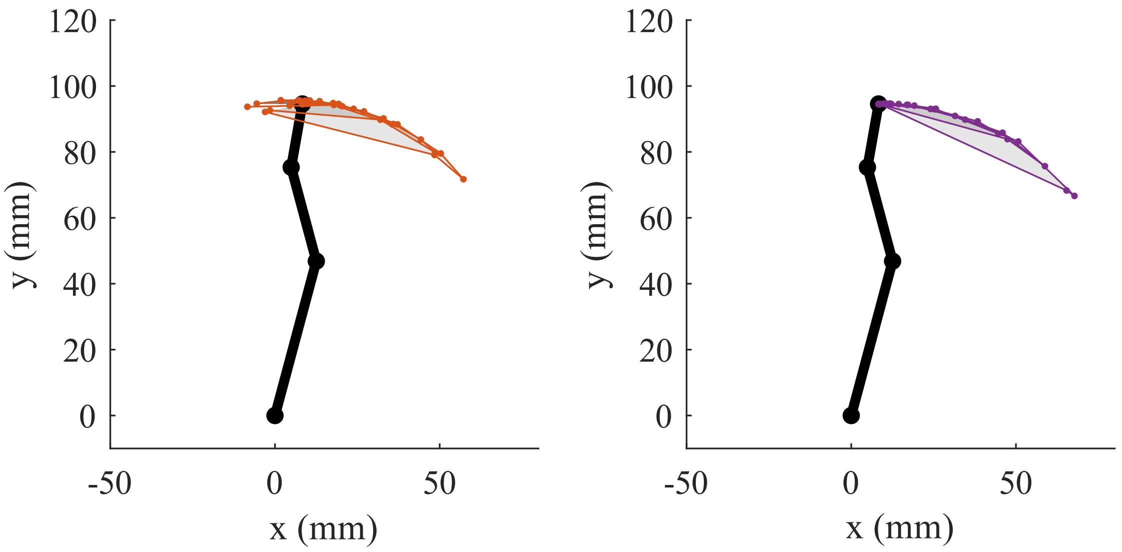

Figure 4.10. Analytical Endpoint Deflection (left) and Experimental Endpoint Deflection (right) for Configuration (c) at 100, 200, 300, 400, and 500 gram loads

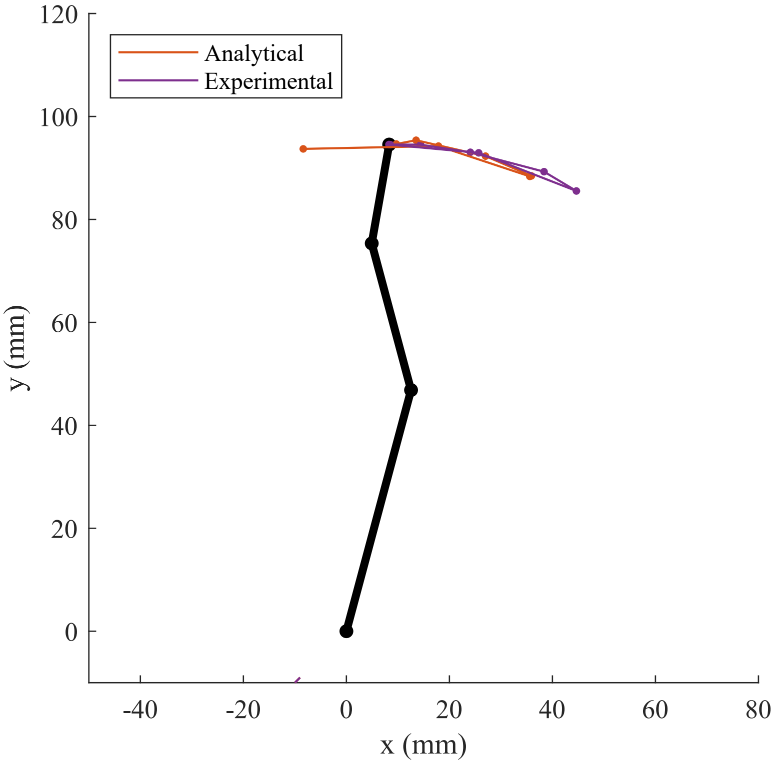

Figure 4.11. Analytical and Experimental Endpoint Deflection for Configuration (c) at 300g Load

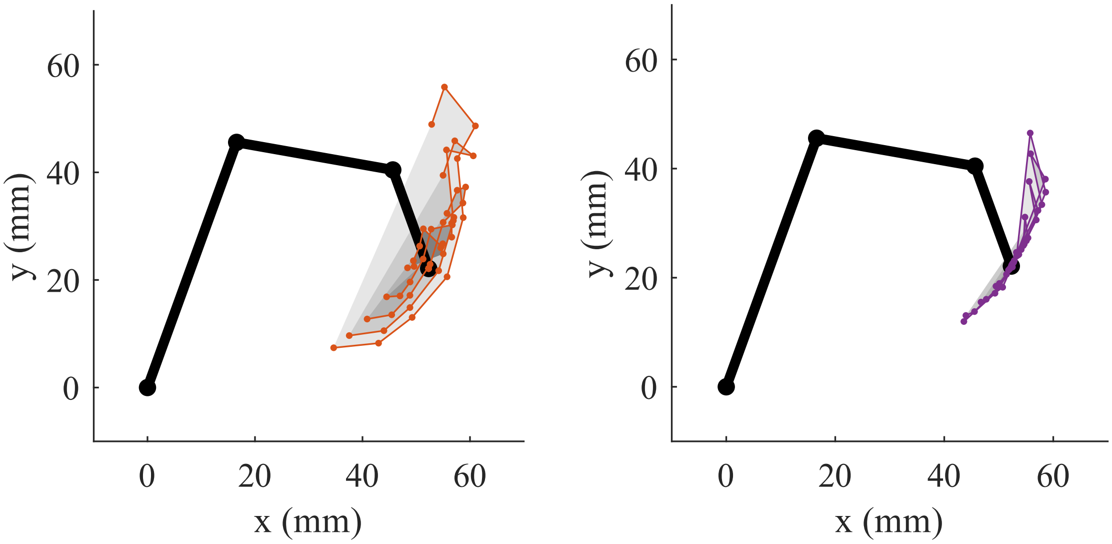

Figure 4.12. Analytical Endpoint Deflection (left) and Experimental Endpoint Deflection (right) for Configuration (d) at 100, 200, 300, 400, and 500 gram loads

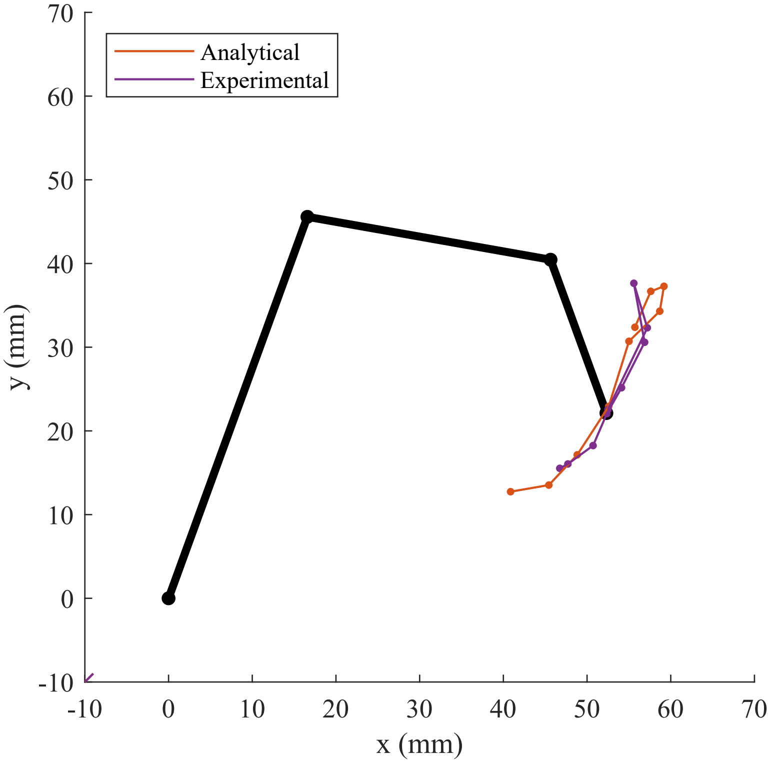

Figure 4.13. Analytical and Experimental Endpoint Deflection for Configuration (d) at 300g Load

4.3 Discussion

Configurations (b), (c), and (d), illustrated in Figure 4.5, match the expected shape of the partial compliance ellipse well, while configuration (a) exhibited much less compliance in the x-direction than expected from the analytical model. The friction torque estimate is likely not a full representation of the friction affecting the joints in each configuration. The friction torque in each joint was modeled using the fit lines generated for the single-joint testing and validated with torque and stiffness measurements to determine the effects of proximal joint stiffness and configuration on distal joint friction. There may also be friction within the joints where the tendons controlling the PIP and DIP joints are in contact due to the central tendon routing method. This friction would occur between the dowel pins routing cables through the MCP and PIP joints as illustrated in Section 2.3.

Higher loading conditions (and, consequently, larger joint deflections) were investigated to ensure the endpoint compliance results were not dominated by friction in the joints. While constant joint stiffness was assumed in the analytical endpoint deflection model, this assumption would not be accurate for large joint deflections. As tension applied to the cable increases with loading on the endpoint, the lever within the affected mechanism would be moved from its set position (angle), effectively changing the stiffness of the joint. Despite this expected discrepancy between the commanded and experimental stiffness values, the experimental endpoint deflection magnitude matched the analytical results well for most configurations at the 100, 200, 300, and 400 gram loading conditions. At 500 grams, the estimated endpoint deflection became much larger than the experimental results, especially for configurations (b) and (d).

CHAPTER 5

DISCUSSION & CONCLUSIONS

The primary objective of this work was to design a 3-DOF planar finger achieving a large range of controllable 2-dimensional linear elastic behavior at the fingertip. This was achieved through the parametric optimization of the VSA mechanisms in order to produce a large range of controllable linear elastic behavior at each joint as well as ensuring decoupling of the elastic and kinematic behavior in the joints.

5.1 Discussion of Results

The optimized VSA mechanism produced quadratic tension-deflection behavior in the tendons. While the experimental relationships did not closely match the analytical MATLAB results and differed between joints, quadratic behavior was achieved, allowing the antagonistic variable stiffness actuators to generate linear variable stiffness. The maximum to minimum stiffness ratio given by the optimization was approximately 8, while the testing displayed ratios of 5-6. This linear behavior was used to adjust joint stiffness values during endpoint testing.

The antagonistic VSAs produced linear variable stiffness in the joints, but friction in the system resulted in nonlinear joint torque-deflection behavior at lower loads, especially for the higher stiffness settings. This friction torque was included in the expected endpoint deflection model. The shape and magnitude of the experimental endpoint compliance ellipses matched the analytical results well for most configurations.

The joints were designed to be kinematically and elastically decoupled. The friction appears to be mainly stiffness-dependent in each joint, although the DIP joint friction may be affected by the configuration of previous joints.

5.2 Work Contributions

The improved optimization method presented in Chapter 3 included a greater number of geometric parameters and accounts for the initial tension in the extensions springs. The initial tension introduced by coils in contact in conventional extension springs presents as an initial VSA mechanism stiffness. This initial stiffness occurs because a minimum load on the spring must be reached to allow separation of the coils. The quadratic tension-deflection behavior in each mechanism successfully produced linear variable stiffness behavior in each joint. This behavior allows the simple control of the endpoint compliance matrix through the adjustment of individual joint angles and stiffness values.

The finger was designed such that the tendon forces remain centered along the finger to reduce unwanted moments in the system. This transmission method performed as expected and there was no observed bending or buckling of components within the VSA mechanisms or finger. Further, the novel cable routing method constrained tendons to the center of each joint to allow the individual adjustment of the stiffness and configuration of each joint.

The friction torque model used to more accurately produce the analytical endpoint compliance ellipses involves finding the initial torque required to move each joint as well as considering how the finger configuration and cable routing may affect friction in the joints. This investigation went beyond infinitesimal endpoint displacements to ensure friction did not dominate the endpoint compliance results.

5.3 Future Work

In order to improve the current design, friction should be reduced. Minimizing friction throughout the system may involve a redesign of the cable routing within the mechanisms, housing, and/or joints. Future work would further investigate the effects of the tendon routing method on friction torque measured at the joints. Additional testing may provide a better model of the friction affecting endpoint deflection, allowing a more accurate comparison between experimental and theoretical endpoint behavior.

A greater stiffness range may be possible if the population of springs considered were expanded. In this thesis work, a maximum spring length of 50 mm was enforced to maintain compactness of the housing. Future implementation should consider using longer springs within the VSA mechanisms to increase the maximum spring deflection, which may increase the linear stiffness range at the joints. This would require that the optimization be performed again to determine the optimal mechanism layout with the modified spring specifications.

REFERENCES

[1] W. Friedl, M. Chalon, J. Reinecke, and M. Grebenstein, "FAS A flexible antagonistic spring element for a high performance over actuated hand," in *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1366–1372, 2011.

[2] R. P. Moore, "Design, Assessment, and Comparison of Antagonistic, Cable-Driven, Variable Stiffness Actuators," Master's thesis, Marquette University, 2020.

[3] K. Laurin-Kovitz, J. Colgate, and S. Carnes, "Design of components for programmable passive impedance," in *Proceedings. 1991 IEEE International Conference on Robotics and Automation*, vol. 2, pp. 1476–1481, 1991.

[4] A. Dollar and R. Howe, "Towards Grasping in Unstructured Environments: Optimization of Grasper Compliance and Configuration," in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*, vol. 3, pp. 3410–3416, 2003.

[5] D. E. Whitney, "Quasi-Static Assembly of Compliantly Supported Rigid Parts," *Journal of Dynamic Systems, Measurement, and Control*, vol. 104, pp. 65–77, 03 1982.

[6] M. Peshkin, "Programmed compliance for error corrective assembly," *IEEE Transactions on Robotics and Automation*, vol. 6, no. 4, pp. 473–482, 1990.

[7] K. Kosuge and M. Shimizu, "Planar parts-mating using structured compliance," in *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No.01CH37180)*, vol. 3, pp. 1477–1482 vol.3, 2001.

[8] S. Huang and J. Schimmels, "Sufficient conditions used in admittance selection for force-guided assembly of polygonal parts," *Robotics and Automation, IEEE Transactions on*, vol. 19, pp. 737 – 742, 09 2003.

[9] S. Huang and J. Schimmels, "Spatial Admittance Selection Conditions for Frictionless Force-Guided Assembly of Polyhedral Parts in Single Principal Contact," *IEEE Transactions on Robotics*, vol. 22, no. 2, pp. 225–239, 2006.

[10] N. Hogan and S. P. Buerger, "Impedance and Interaction Control," in *Robotics and Automation Handbook*, pp. 19-1–19-24, CRC press, 2005.

[11] D. E. Whitney, "Remote Center Compliance Gripper System." *US4537557A*, Aug 1985.

[12] G. Pratt and M. Williamson, "Series elastic actuators," in *Proceedings 1995 IEEE/RSJ International Conference on Intelligent Robots and Systems. Human Robot Interaction and Cooperative Robots*, vol. 1, pp. 399–406, 1995.

[13] C. Fitzgerald, “Developing Baxter,” in *2013 IEEE Conference on Technologies for Practical Robot Applications (TePRA)*, pp. 1–6, 2013.

[14] ReThink Robotics, “Sawyer, the high performance collaborative robot.” ReThink Robotics GmbH. Accessed 2022. [Online]. Available: https://www.rethinkrobotics.com/sawyer.

[15] M. Diftler, J. Mehling, M. Abdallah, N. Radford, L. Bridgwater, A. Sanders, R. Askew, D. Linn, J. Yamokoski, F. Permenter, B. Hargrave, R. Platt, R. Savely, and R. Ambrose, “Robonaut 2 - The first humanoid robot in space,” in *2011 IEEE International Conference on Robotics and Automation*, pp. 2178–2183, 2011.

[16] S. Migliore, E. Brown, and S. DeWeerth, “Biologically Inspired Joint Stiffness Control,” in *Proceedings of the 2005 IEEE International Conference on Robotics and Automation*, pp. 4508–4513, 2005.

[17] R. V. Ham, T. G. Sugar, B. Vanderborght, K. W. Hollander, and D. Lefeber, “Compliant actuator designs,” *IEEE Robotics & Automation Magazine*, vol. 16, no. 3, pp. 81–94, 2009.

[18] B. Vanderborght, A. Albu-Schäeffer, A. Bicchi, E. Burdet, D. Caldwell, R. Carloni, M. Catalano, O. Eiberger, W. Friedl, G. Ganesh, M. Garabini, M. Grebenstein, G. Grioli, S. Haddadin, H. Höppner, A. Jafari, M. Laffranchi, D. Lefeber, F. Petit, and S. Wolf, “Variable Impedance Actuators: a Review,” *Robotics and Autonomous Systems*, vol. 61, p. 1601–1614, 12 2013.

[19] J. Choi, S. Park, W. Lee, and S.-C. Kang, “Design of a robot joint with variable stiffness,” in *2008 IEEE International Conference on Robotics and Automation*, pp. 1760–1765, 2008.

[20] S. Wolf and G. Hirzinger, “A new variable stiffness design: Matching requirements of the next robot generation,” in *2008 IEEE International Conference on Robotics and Automation*, pp. 1741–1746, 2008.

[21] J. M. Schimmels and D. R. Garces, “The Arched Flexure VSA: A compact variable stiffness actuator with large stiffness range,” in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 220–225, 2015.

[22] C.-P. Chou and B. Hannaford, “Static and dynamic characteristics of McKibben pneumatic artificial muscles,” in *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, vol. 1, pp. 281–286, 1994.

[23] M. Trumić, K. Jovanovic, and A. Fagiolini, “Decoupled nonlinear adaptive control of position and stiffness for pneumatic soft robots,” *The International Journal of Robotics Research*, vol. 40, pp. 277–295, 02 2020.

[24] M. Grebenstein, M. Chalon, G. Hirzinger, and R. Siegwart, “Antagonistically driven finger design for the anthropomorphic DLR Hand Arm System,” in *2010 10th IEEE-RAS International Conference on Humanoid Robots*, pp. 609–616, 2010.

[25] M. Grebenstein, A. Albu-Schäffer, T. Bahls, M. Chalon, O. Eiberger, W. Friedl, R. Gruber, S. Haddadin, U. Hagn, R. Haslinger, H. Höppner, S. Jörg, M. Nickl, A. Nothhelfer, F. Petit, J. Reill, N. Seitz, T. Wimböck, S. Wolf, and G. Hirzinger, "The DLR hand arm system," in *2011 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3175–3182, 06 2011.

[26] S. Wolf, O. Eiberger, and G. Hirzinger, "The DLR FSJ: Energy based design of a variable stiffness joint," in *2011 IEEE International Conference on Robotics and Automation*, pp. 5082–5089, 2011.

[27] F. Petit, M. Chalon, W. Friedl, M. Grebenstein, A. Albu-Schäffer, and G. Hirzinger, "Bidirectional antagonistic variable stiffness actuation: Analysis, design & Implementation," in *2010 IEEE International Conference on Robotics and Automation*, pp. 4189–4196, 2010.

[28] F. Petit, W. Friedl, H. Höppner, and M. Grebenstein, "Analysis and Synthesis of the Bidirectional Antagonistic Variable Stiffness Mechanism," *IEEE/ASME Transactions on Mechatronics*, vol. 20, no. 2, pp. 684–695, 2015.

[29] R. Moore and J. M. Schimmels, "Design of a Quadratic, Antagonistic, Cable-Driven, Variable Stiffness Actuator," in *International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, vol. 10: 44th Mechanisms and Robotics Conference (MR), 08 2020.

[30] W. Friedl, H. Höppner, F. Schmidt, M. A. Roa, and M. Grebenstein, "CLASH: Compliant Low Cost Antagonistic Servo Hands," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6469–6476, 2018.

[31] Shadow Robot Company, "Dexterous Hand Series: The World's Most Dexterous Humanoid Robot Hands." Shadow Robot Company. Accessed 2022. [Online]. Available: https://www.shadowrobot.com/dexterous-hand-series/ .

[32] F. Rothling, R. Haschke, J. J. Steil, and H. Ritter, "Platform portable anthropomorphic grasping with the bielefeld 20-DOF shadow and 9-DOF TUM hand," in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 2951–2956, 2007.

[33] M. Grebenstein, M. Chalon, W. Friedl, S. Haddadin, T. Wimböck, G. Hirzinger, and R. Siegwart, "The hand of the DLR Hand Arm System: Designed for interaction," *The International Journal of Robotics Research*, vol. 31, no. 13, pp. 1531–1555, 2012.

[34] S. Huang and J. M. Schimmels, "Realization of point planar elastic behaviors using revolute joint serial mechanisms having specified link lengths," *Mechanism and Machine Theory*, vol. 103, pp. 1–20, 2016.

[35] B. Alexander and V. Kotiuk, "Proportions of Hand Segments," *International Journal of Morphology*, vol. 28, pp. 755–758, 09 2010.

[36] J. D. Emery, “Pulley Belt Length, Circle Tangents, and Conic Sections,” 12 2012. Cupdf. Accessed 2022. [Online]. Available: https://cupdf.com/document/pulley-belt-length-circle-tangents-and-conic-belt-length-circle-tangents-and.html?

APPENDIX A

PHYSICAL DESIGN FEATURES

This appendix presents several physical design features of the VSA mechanisms and finger not included in the main body of this thesis.

A.1 Tendon Anchoring

The tendon-driven actuation of the joints is enabled by the anchoring of the tendons to the links. At each pulley, two perpendicular set screws hold a dowel pin against a notch in the pulley, effectively clamping the cable between the pulley and dowel pin. This is illustrated in the CAD model and photo in Figures A.1 and A.2, respectively.

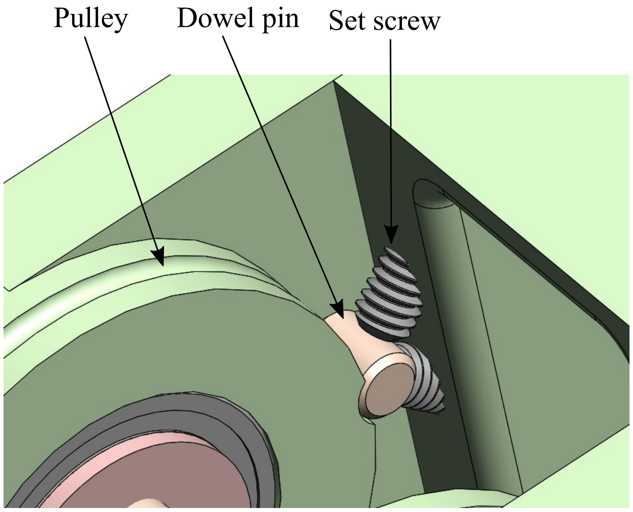

Figure A.1. Finger Model showing Tendon Attachment Method

The tendons are anchored within each VSA mechanism at the motor shaft, shown in Figure A.3, which winds the cable to increase or decrease the length of cable within the system. The end of the tendon is wrapped around a tightened cable termination screw, then is fed through a hole where it enters and wraps around the winding pulley.

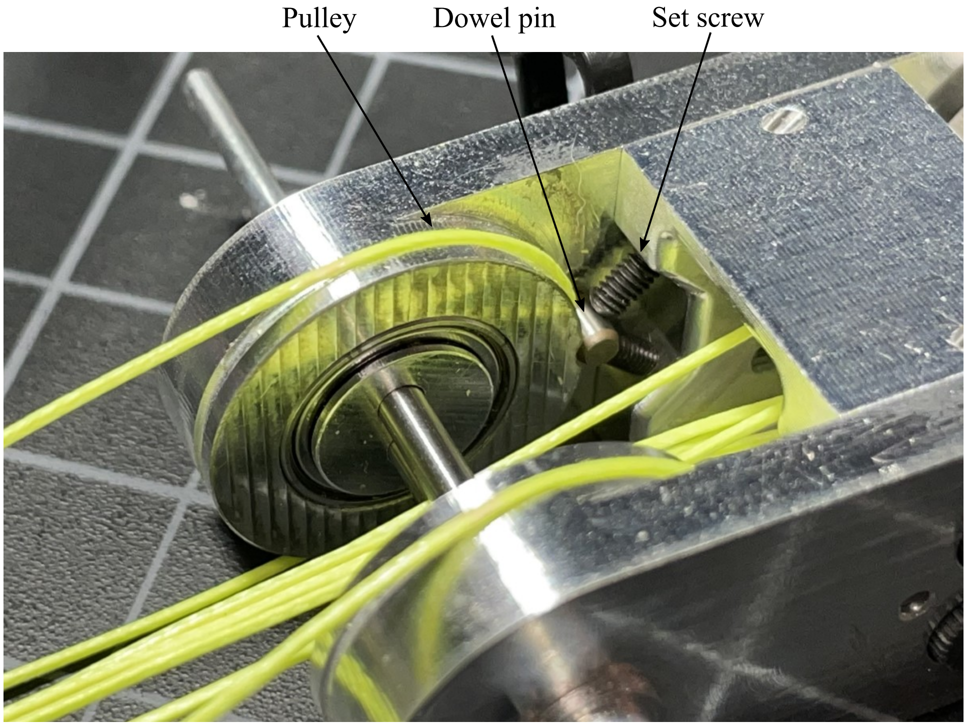

Figure A.2. Photo showing MCP Joint Tendon Attachment Method

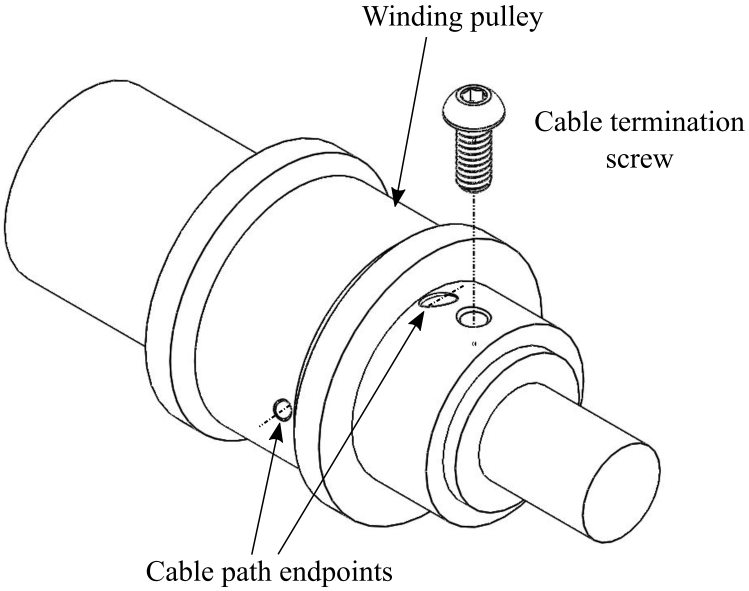

Figure A.3. Winding Motor Shaft Model

A.2 Tendon Routing

The lever, shown in Figure A.4, was designed such that the cable does not interfere with the lever in its initial position. The base of the lever is clamped to a freely rotating dowel pin. On the other end, the lever pulley is centered in a slot and the two springs are connected on either side of the lever. A slot on one side of the lever provides clearance for the tendon between the motor pulley and lever pulley as shown in Figure A.5.

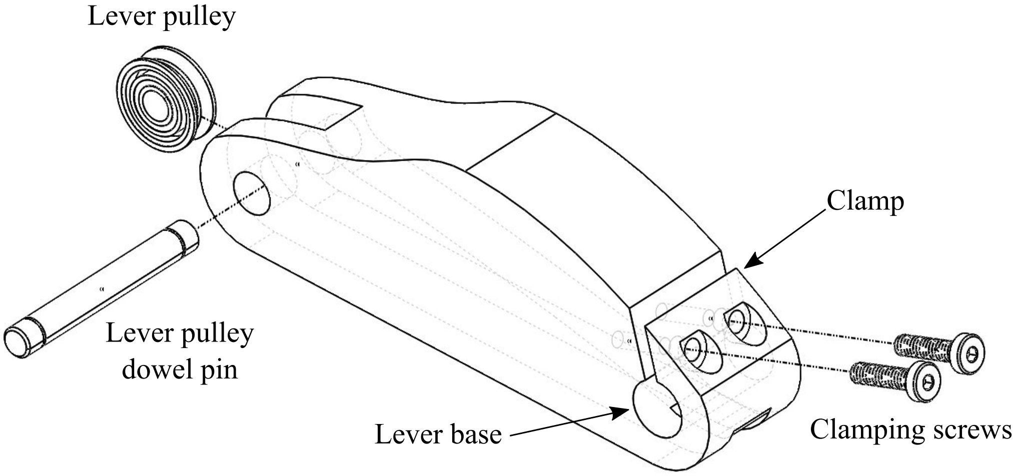

Figure A.4. Lever Piece Model

The joint center tendon routing method of the MCP joint is shown in Figure A.6. The same method is used in the PIP joint. Two dowel pins ensure that the cable passes through the center of the joint. The pins are held between freely rotating disks to minimize any effect of the joint position on the tension in the cable.

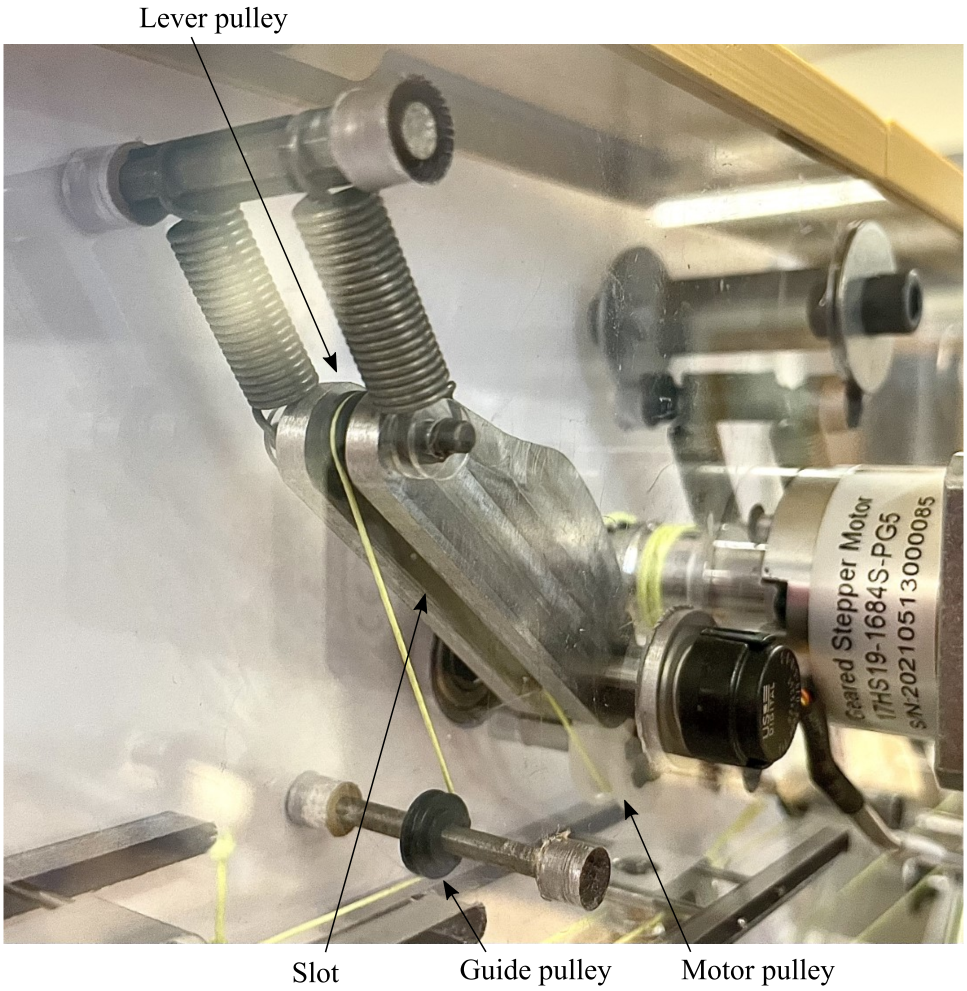

Figure A.5. Photo showing VSA Mechanism Lever

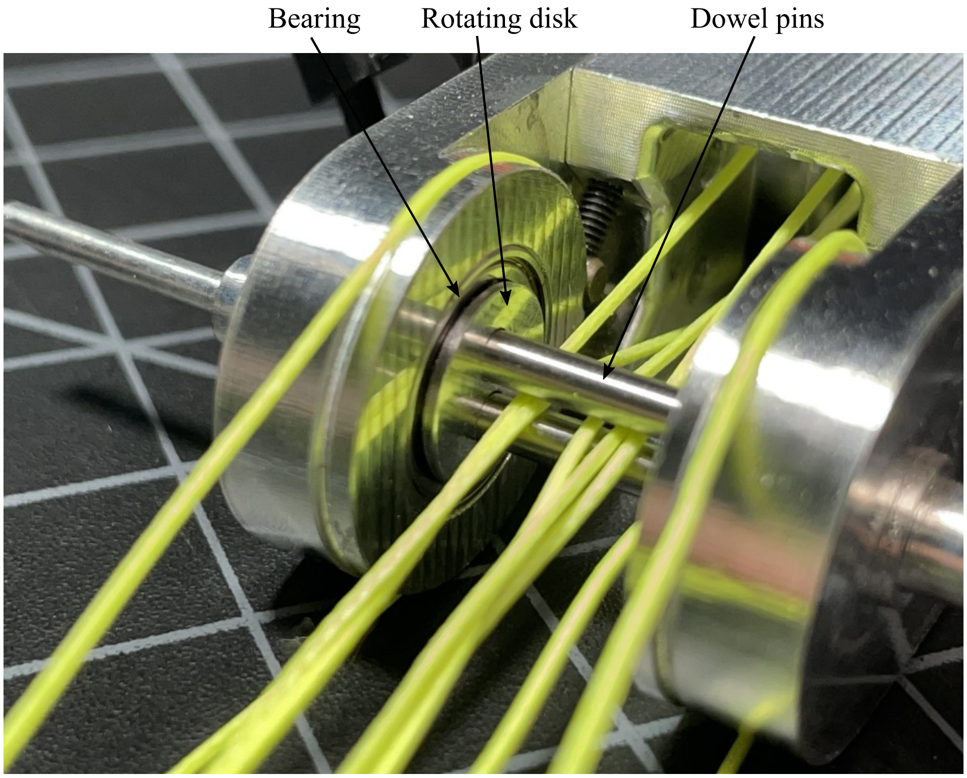

Figure A.6. Photo showing MCP Joint Cable Routing

A.3 Prototype Photos

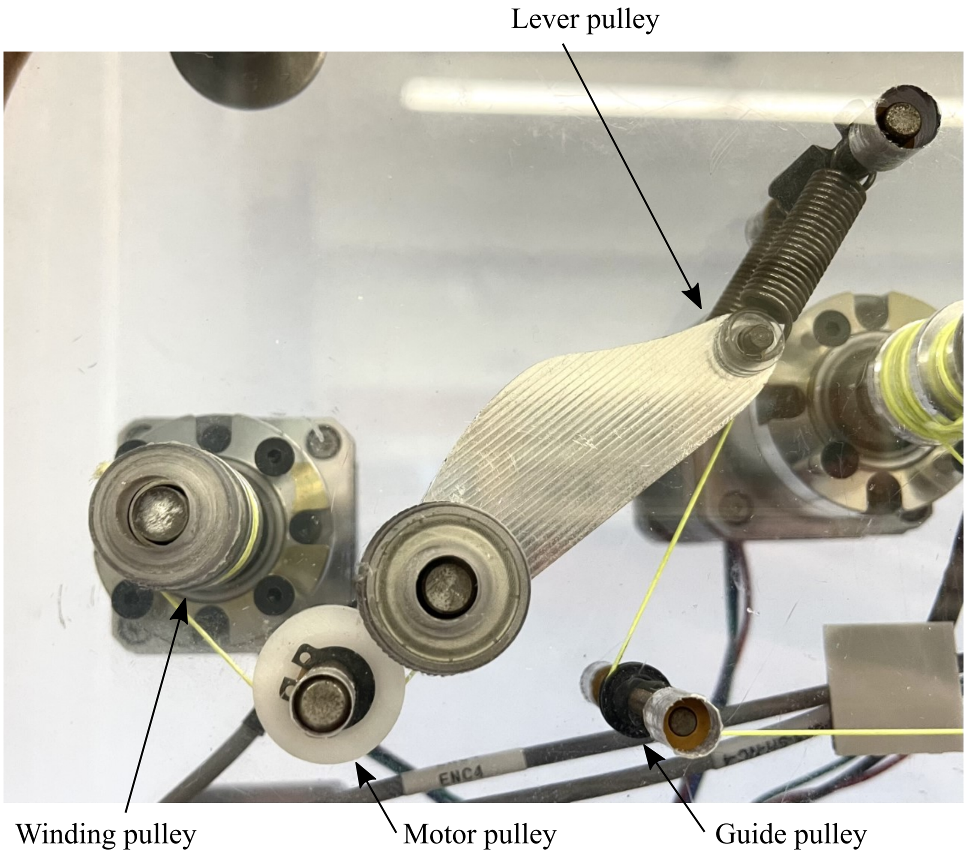

Figure A.7. Photo showing Single VSA Mechanism

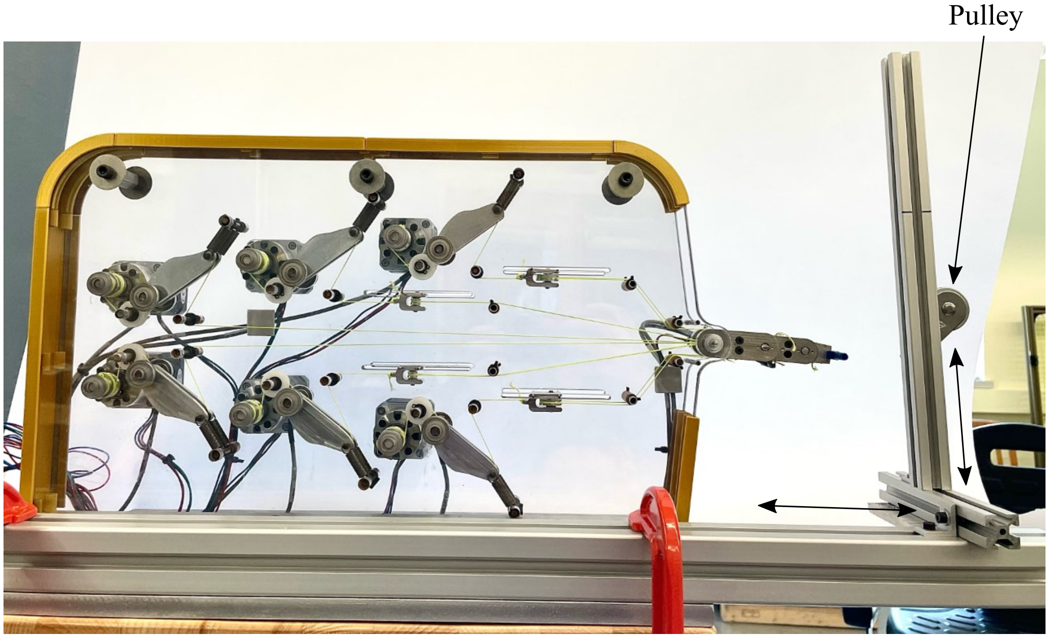

Figure A.8. Photo showing Variable Stiffness Finger Prototype Test Fixture

APPENDIX B

INITIAL MECHANISM STIFFNESS

The initial tension in the extension springs results in an initial stiffness in the mechanism. The magnitude of the spring force F_s is given by

$$F_s = K\Delta L_s + T_i \tag{B.1}$$

where the spring constant K, spring deflection ΔL_s from initial spring length L_i, and initial spring tension T_i contribute to F_s. Expanding Equation 3.2 for the moment about the lever base $\mathbf{M_O}$ to include the magnitude of the spring force results in

$$\mathbf{M_O} = ((\mathbf{r_1} \times \hat{\mathbf{T}}_\mathbf{1}) + (\mathbf{r_2} \times \hat{\mathbf{T}}_\mathbf{2}))T + (\mathbf{r_S} \times (K\Delta L_s + T_i)\hat{\mathbf{F}}_\mathbf{s}) = \mathbf{0}. \tag{B.2}$$

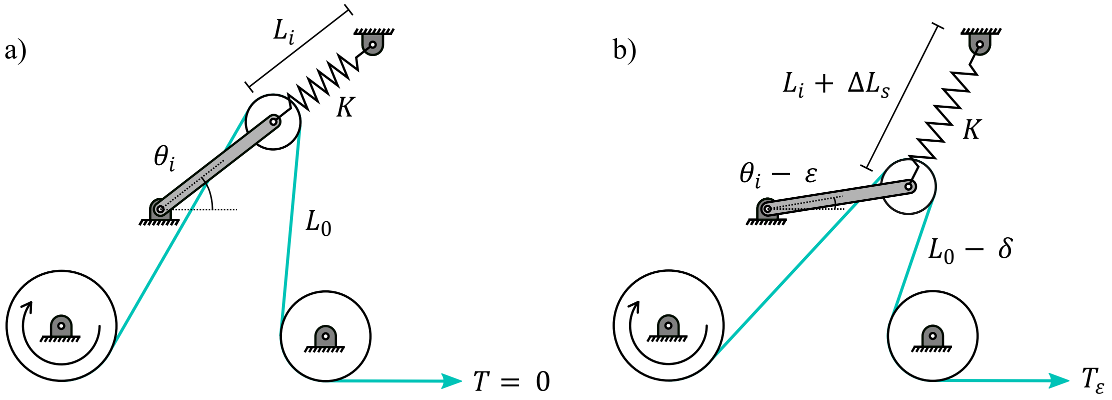

Figure B.1. VSA mechanism at a) its initial position and b) an infinitesimal lever angle deflection ε (shown as a large deflection for illustration purposes)

When $\mathbf{r_s}$ and $\mathbf{F_s}$ (illustrated in Figure 3.3) are collinear as in Figure B.1(a), the cross product determining the spring force vector is equal to zero, despite a nonzero spring force magnitude F_s due to initial tension T_i in the spring. The stiffness is nonzero as the tension in the cable must be high enough to overcome T_i in order to cause infinitesimal angular motion ε of the lever. This is given by

$$K_i = \frac{T_\varepsilon}{\delta} \tag{B.3}$$

where the initial stiffness of the mechanism K_i occurs at an infinitesimal lever angle deflection ε associated with a cable deflection of δ. The spring deflection ΔL_s, cable deflection δ from the initial cable length L_0, and cable tension T_ε due to lever rotation ε are illustrated in Figure B.1(b).

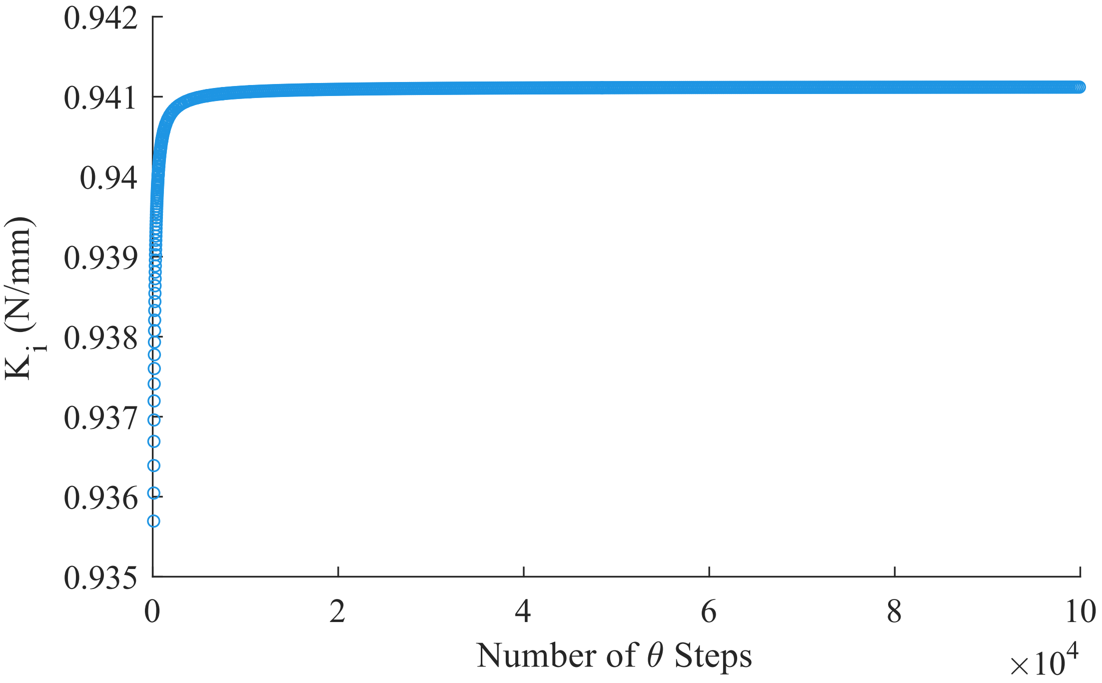

Figure B.2. Convergence of Initial Mechanism Stiffness with Increasing Number of Steps (Decreasing Step Size) used in Lever Angle Array in Numerical Model

In order to ensure that the initial mechanism stiffness found by the numerical model was not a result of numerical error, the tension-deflection and stiffness-deflection curves were generated using an increasing number of evaluated points. The numerical model calculates cable tension and cable length for an array of lever angles (from θ_i to θ_f). The number of steps used in this array increased for each evaluation.

In Figure B.2, the first value of each calculated stiffness array is plotted against the number of steps used in the lever angle array for a single evaluation. For the optimal mechanism layout, the initial mechanism stiffness found by the numerical model converges to 0.9411 N/mm. This value varies with mechanism layout parameters and spring specifications.

APPENDIX C

CONTROL SYSTEM PROGRAMMING

This appendix contains flowcharts for the joint homing and configuration routines and their subroutines. These are basic overviews of the program created in LabVIEW for homing and controlling the three joints of the variable stiffness finger. Section C.1 presents the homing routine and the subroutines for homing the lever angles and joint angles. Section C.2 presents the joint configuration routine and the subroutines for executing the commanded lever angles and joint angles.

C.1 Homing Routine

Figure C.1 outlines the main homing routine. Figures C.2 and C.3 describe the lever angle and joint angle homing subroutines, respectively. The subroutines are used several times within the main routine, e.g., the same lever angle homing subroutine is used for homing each VSA.

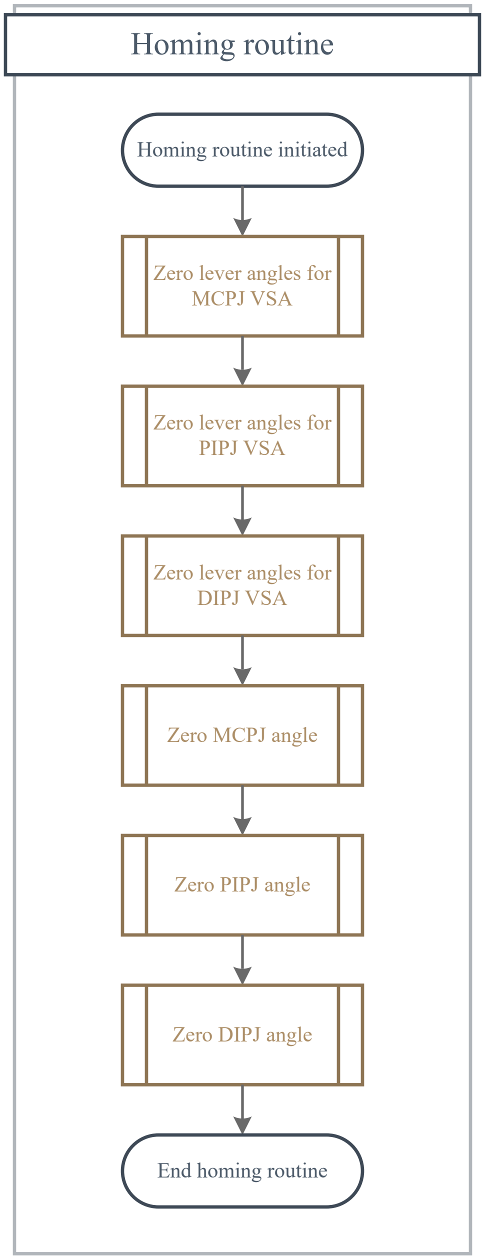

Figure C.1. Main Homing Routine Flowchart

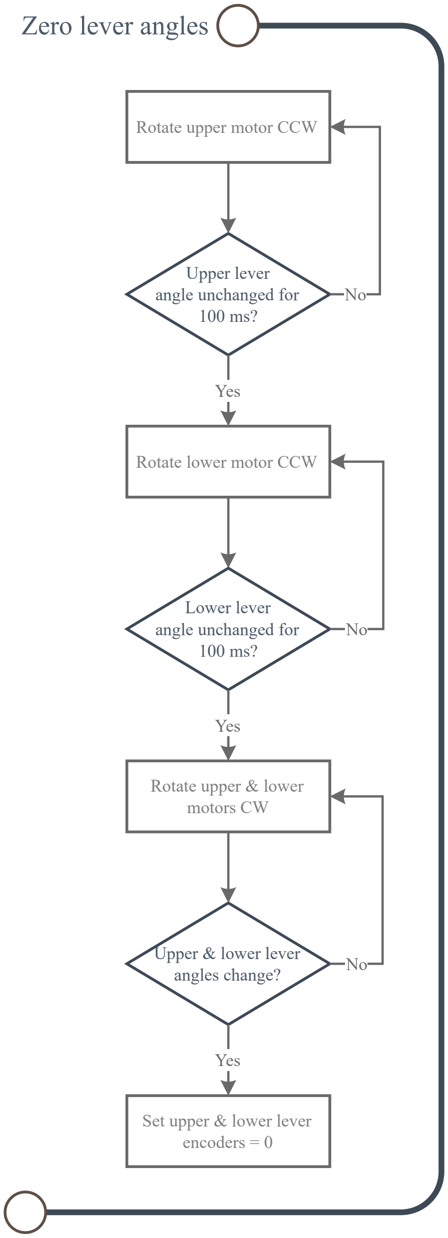

Figure C.2. Lever Angle Homing Subroutine Flowchart

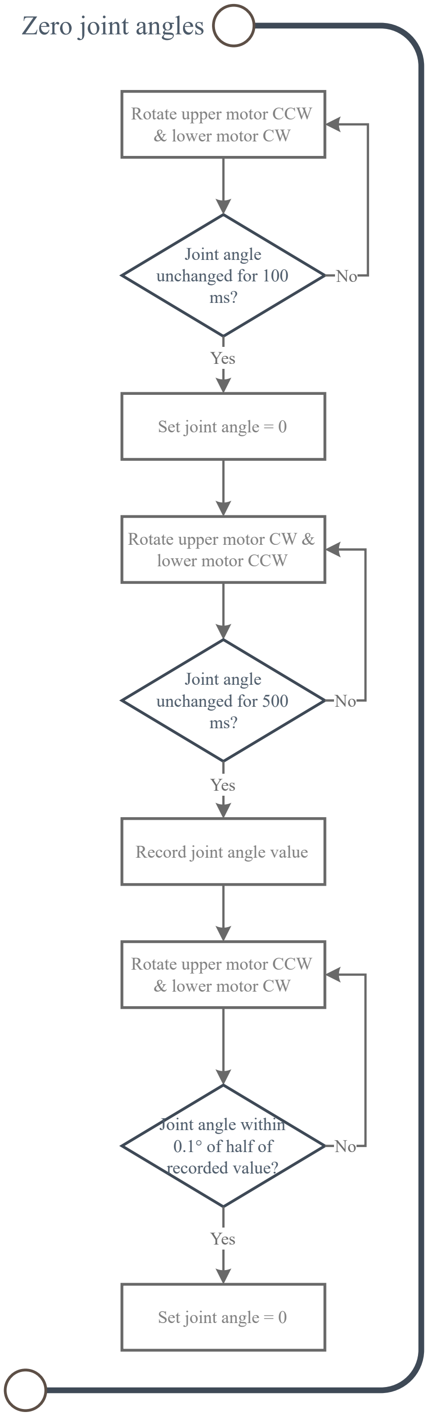

Figure C.3. Joint Angle Homing Subroutine Flowchart

C.2 Joint Configuration Routine

Figure C.4 outlines the main joint configuration routine. Figures C.5 and C.6 describe the lever angle and joint angle configuration subroutines, respectively. The stiffness of each joint is calculated using the VSA lever angles, provided by encoders, and the experimental relationships between lever angle and joint stiffness found in Chapter 3.

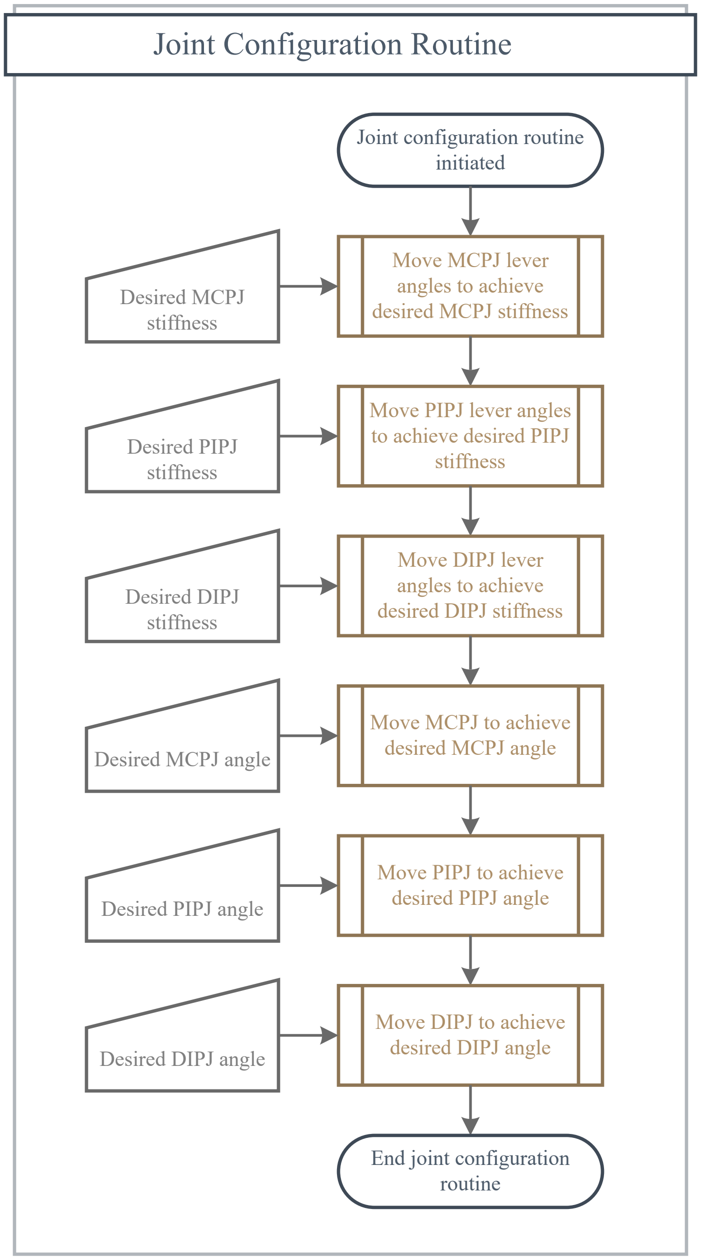

Figure C.4. Joint Configuration Routine Flowchart

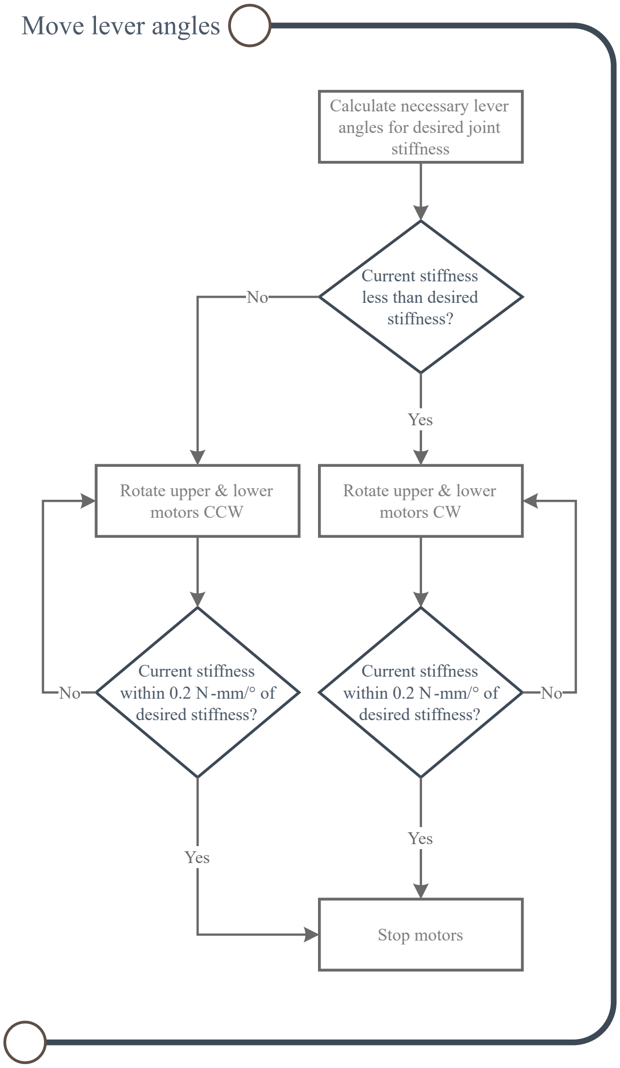

Figure C.5. Lever Angle Subroutine Flowchart

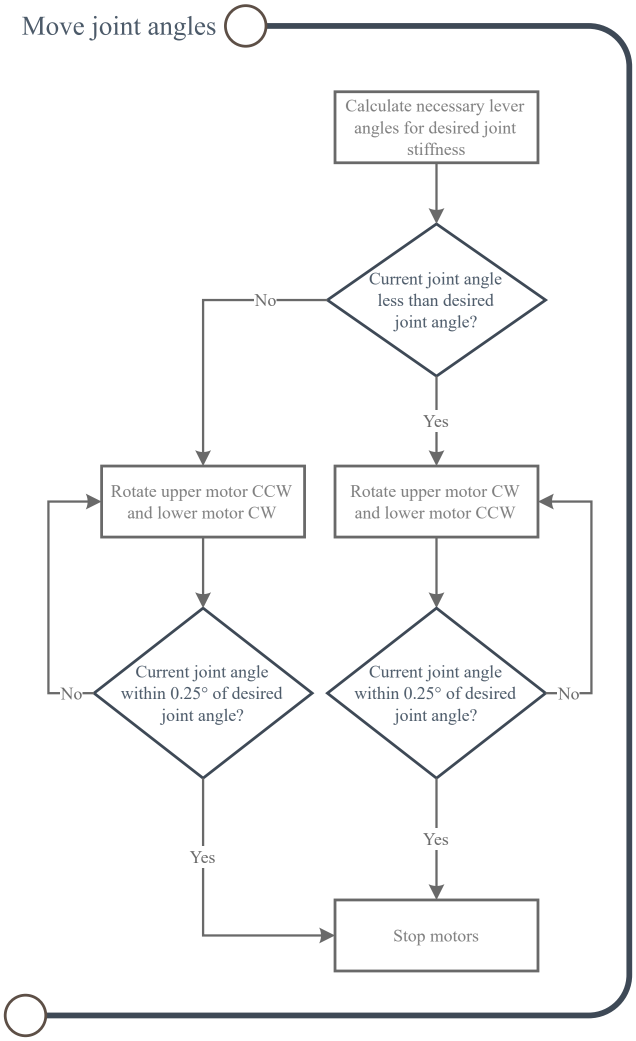

Figure C.6. Joint Angle Subroutine Flowchart

APPENDIX D

ANTAGONISTIC COMPLIANT ACTUATOR TESTING

This appendix contains the results of the testing performed in Chapter 4. In Section D.1, the results of the joint torque-deflection tests are presented for lever angles from 5° to 30° as well as one graph showing the initial joint torque due to friction for each joint at each lever angle. The procedure and results for the experimental validation of friction torque estimates are presented in Section D.2.

D.1 Joint Torque-Deflection Results

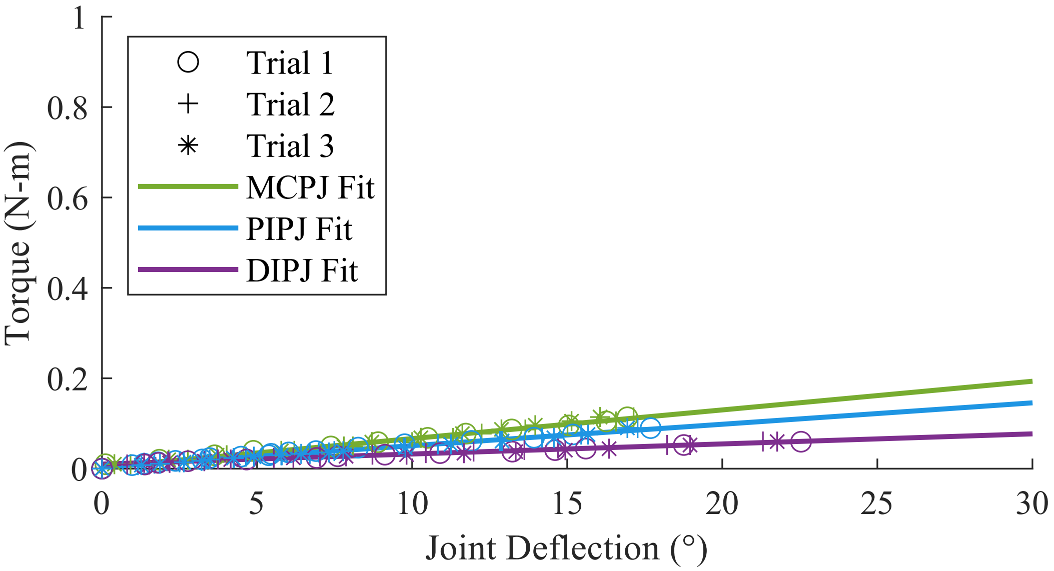

Figure D.1. Torque vs. Joint Deflection for 5° Lever Angles

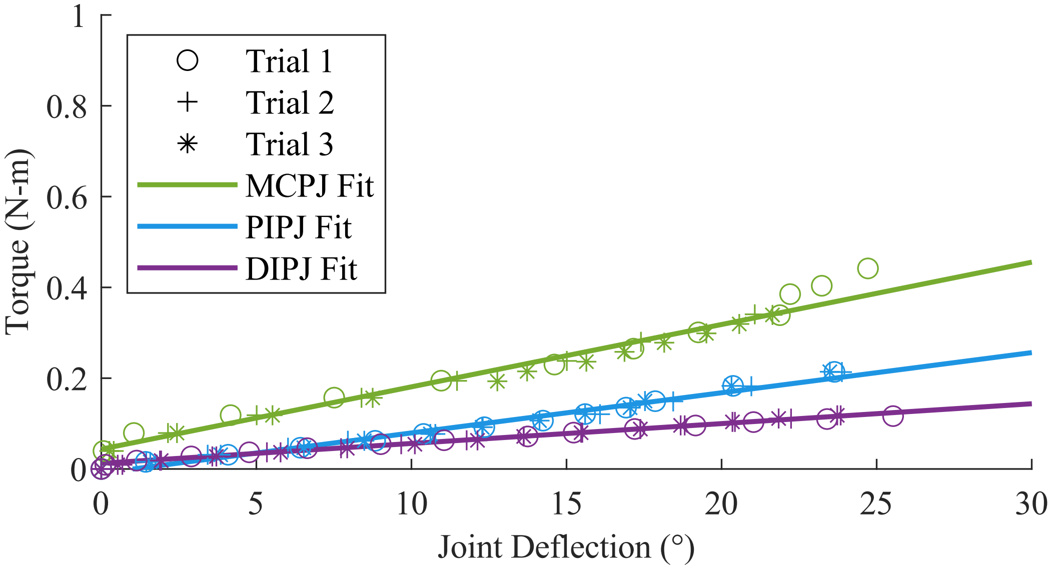

Figure D.2. Torque vs. Joint Deflection for 10° Lever Angles

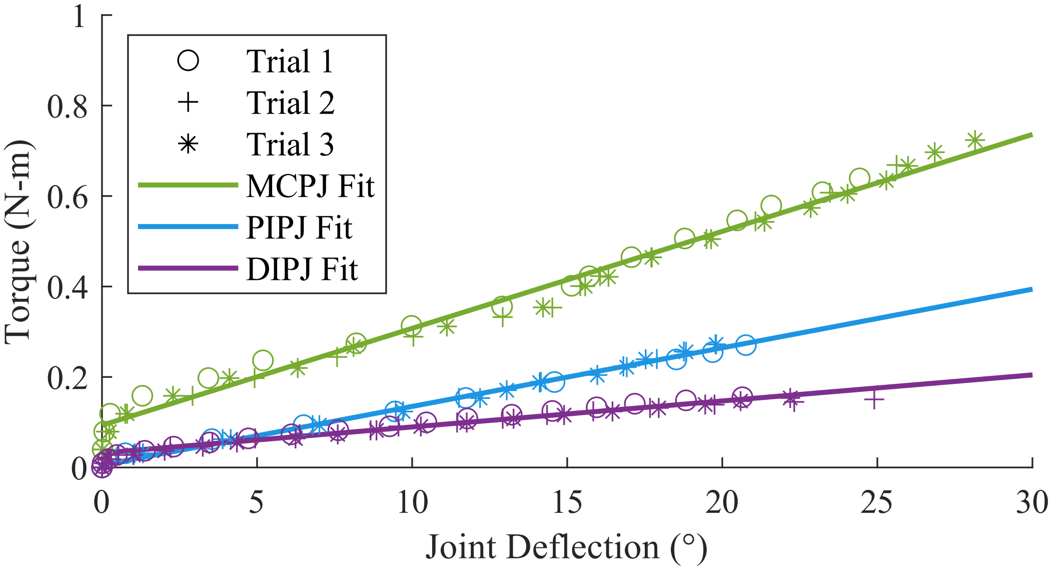

Figure D.3. Torque vs. Joint Deflection for 15° Lever Angles

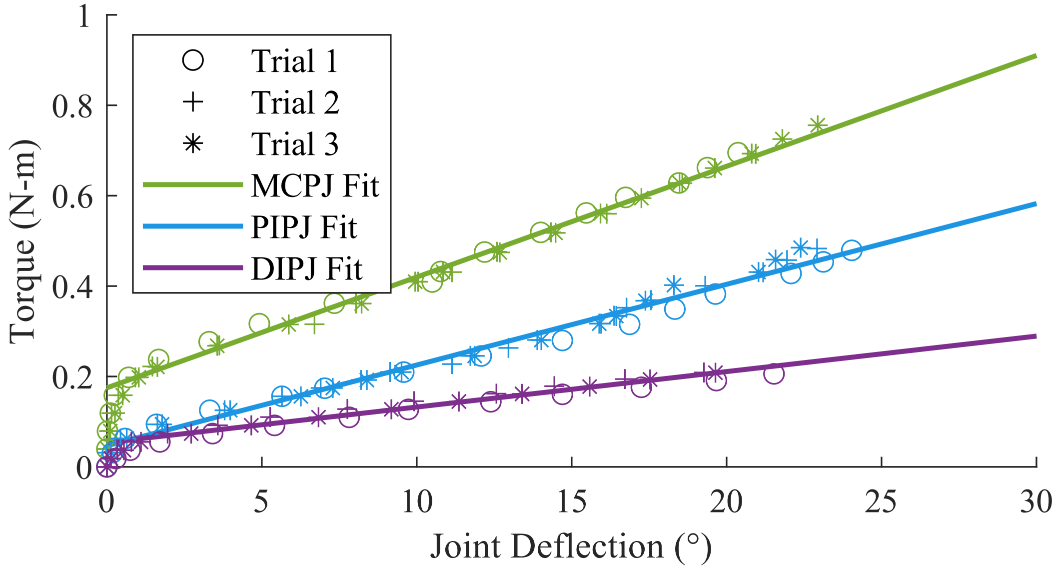

Figure D.4. Torque vs. Joint Deflection for 20° Lever Angles

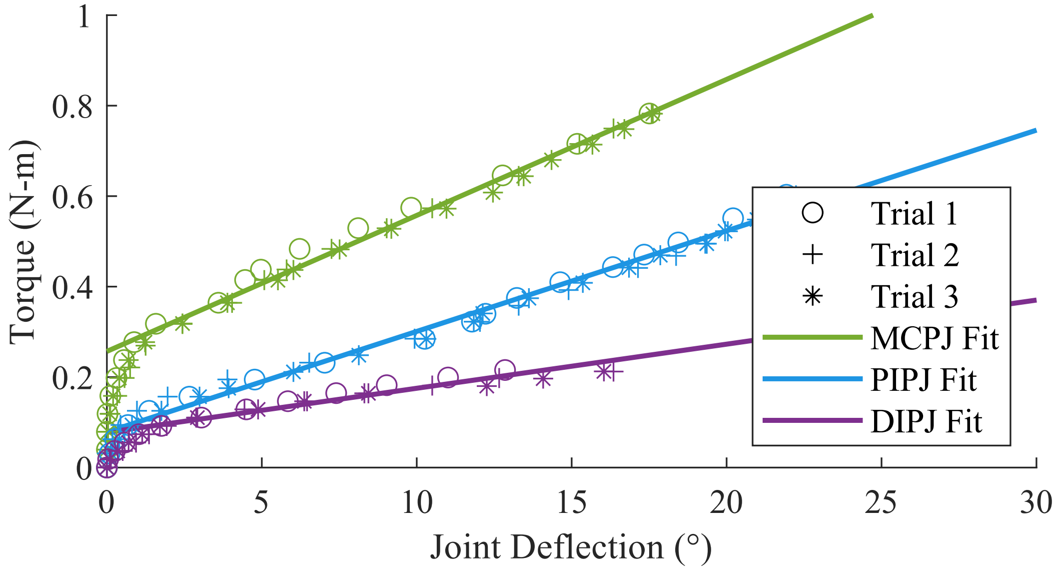

Figure D.5. Torque vs. Joint Deflection for 25° Lever Angles

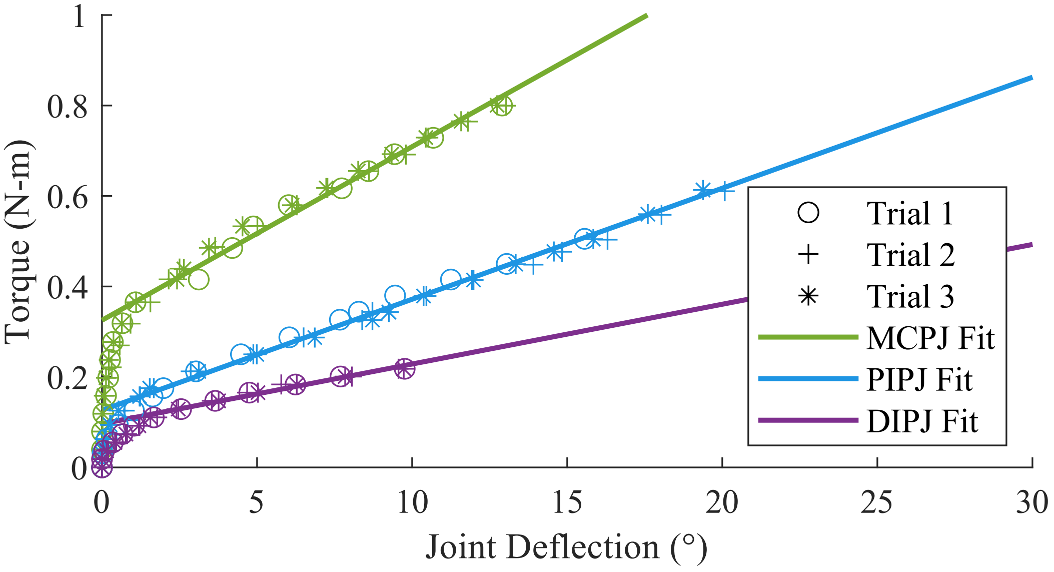

Figure D.6. Torque vs. Joint Deflection for 30° Lever Angles

D.2 Joint Friction (Initial Torque) Validation Test

The initial joint torque shown in Figure 4.2 is modeled only as a function of joint stiffness. Validation tests were performed to determine the accuracy of this model and establish any dependence on the PIP or DIP joint configuration.

The testing apparatus shown in Figure A.8 was used to generate a torque at a single joint as indicated in Figure D.7. In configuration (A), all joint locations are fixed with the DIP joint set at a -90° angle and torque applied at the DIP joint. In configuration (B), the MCP and PIP joints are fixed and the PIP joint is set at -90° with torque applied at the PIP joint. Configuration (C) is similar to (B), except all joint locations are fixed and torque is instead applied at the DIP joint. The joint torque-deflection relationship was taken at two stiffness conditions for each configuration. Each stiffness condition (i.e., commanded K_1, K_2, and K_3) approximately corresponded to the stiffness values used in the endpoint compliance testing.

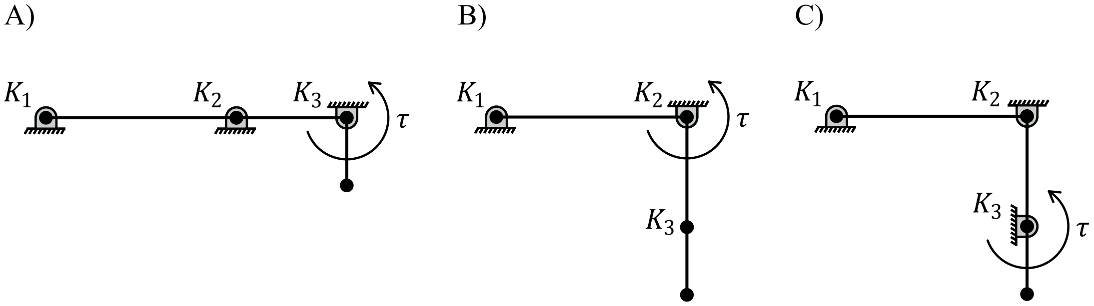

Figure D.7. Configurations for Friction Validation Test

The initial torque due to friction for the joint being tested in each configuration is taken as the y-intercept of the average torque-deflection curves for the linear portion of the data as shown in Figures D.8-D.13. The largest discrepancy between the model in Figure 4.2 and the validation tests was for the initial torque of the DIP joint from tests for configuration (C). This may indicate a relationship between the friction at the DIP joint and the configuration of prior joints. The higher of the two estimates (the value given by the relationships in Figure 4.2 vs. the initial torque provided by this testing) for a given joint stiffness and configuration was used as a constant friction torque in the calculations for the expected endpoint compliance behavior in Chapter 4.

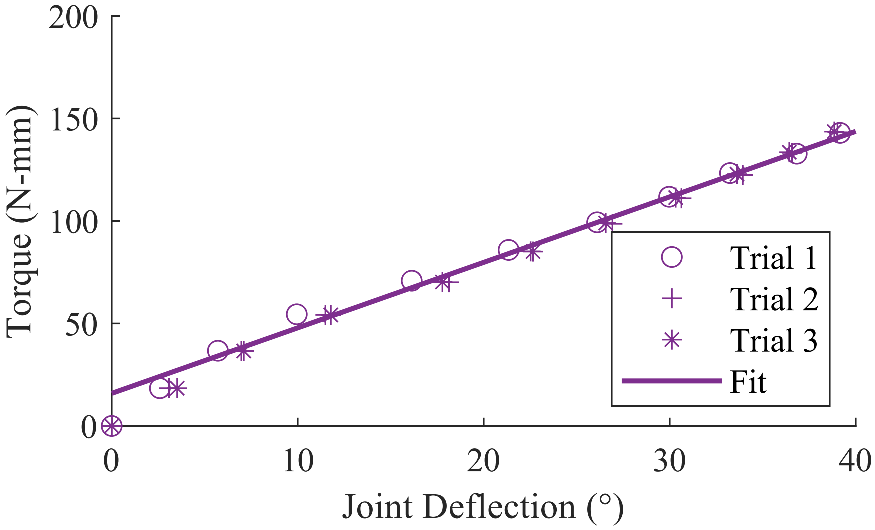

Figure D.8. Joint Torque-Deflection Curve for the DIP Joint in Configuration (A) where $K_1 \approx 15$ N-mm/°, $K_2 \approx 15$ N-mm/°, $K_3 \approx 2.5$ N-mm/°

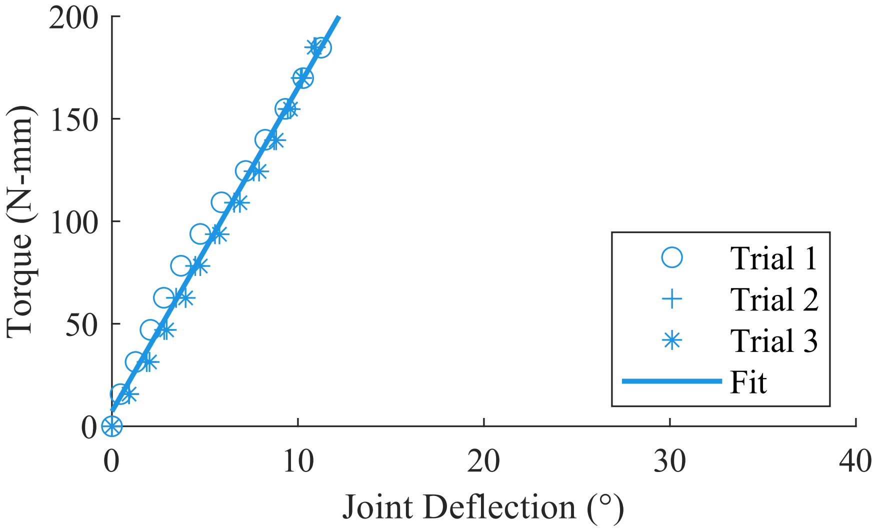

Figure D.9. Joint Torque-Deflection Curve for the PIP Joint in Configuration (B) where $K_1 \approx 15$ N-mm/°, $K_2 \approx 15$ N-mm/°, $K_3 \approx 2.5$ N-mm/°

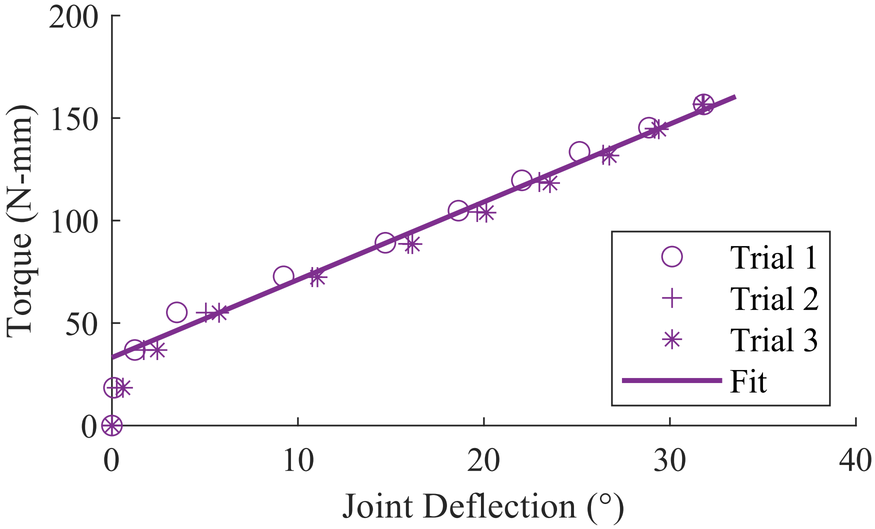

Figure D.10. Joint Torque-Deflection Curve for the DIP Joint in Configuration (C) where $K_1 \approx 15$ N-mm/°, $K_2 \approx 15$ N-mm/°, $K_3 \approx 2.5$ N-mm/°

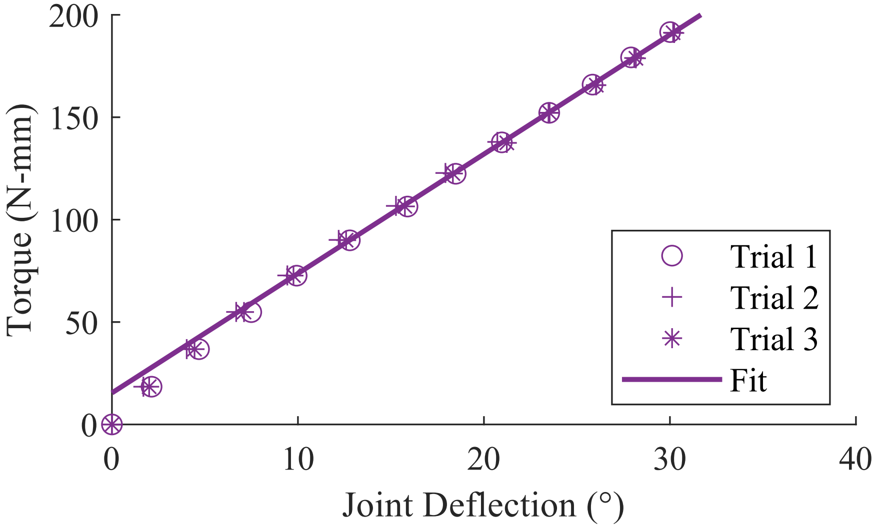

Figure D.11. Joint Torque-Deflection Curve for the DIP Joint in Configuration (A) where $K_1 \approx 25$ N-mm/°, $K_2 \approx 5$ N-mm/°, $K_3 \approx 5$ N-mm/°

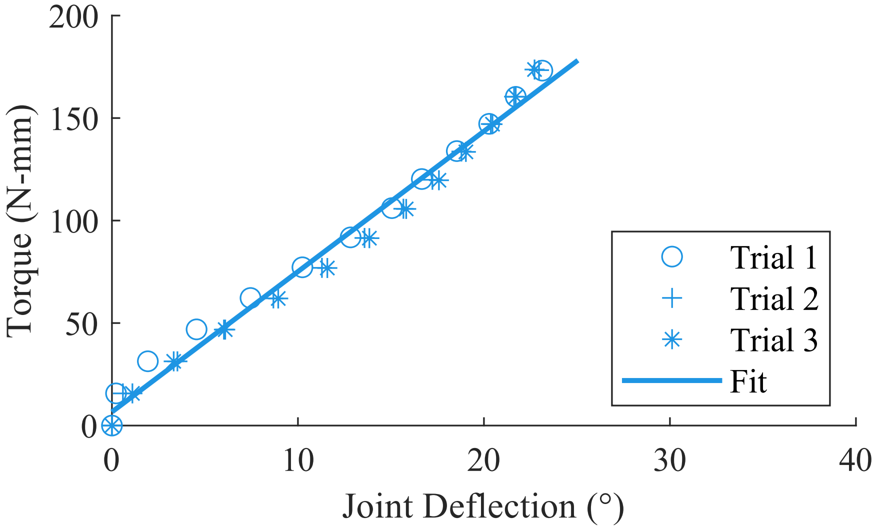

Figure D.12. Joint Torque-Deflection Curve for the PIP Joint in Configuration (B) where $K_1 \approx 25$ N-mm/°, $K_2 \approx 5$ N-mm/°, $K_3 \approx 5$ N-mm/°

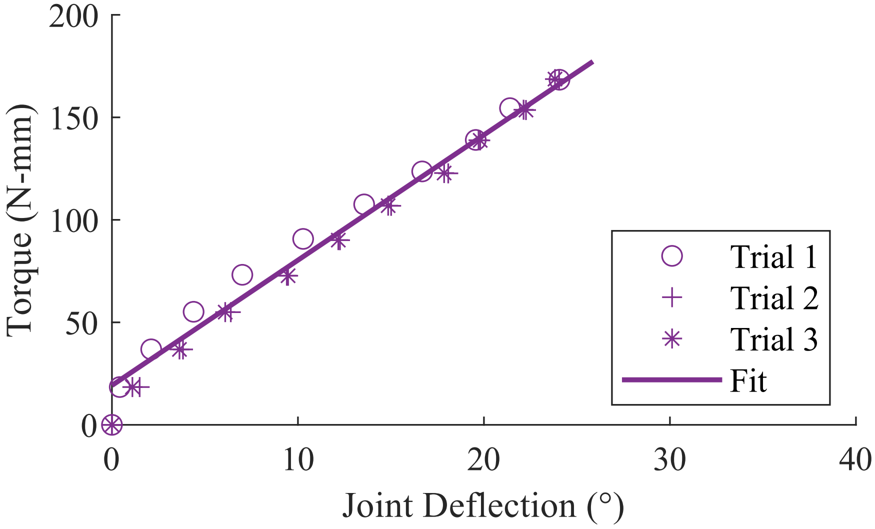

Figure D.13. Joint Torque-Deflection Curve for the DIP Joint in Configuration (C) where $K_1 \approx 25$ N-mm/°, $K_2 \approx 5$ N-mm/°, $K_3 \approx 5$ N-mm/°

APPENDIX E

MECHANISM GEOMETRY OPTIMIZATION

This appendix contains the MATLAB code used in the optimization of the VSA mechanism design parameters. The main script (LeverOptim_ga) initiates the genetic algorithm and sets parameter bounds while the other six functions run inside the main loop. These functions are used for the geometric analysis, force calculations, and calculation of objective function values.

E.1 LeverOptim_ga_v3_scaled.m

```
% Author: Allison Goetz
% Description:  Optimization of Lever Quadratic Spring Mechanism
% Parameters to follow a desired quadratic force/displacement path
% of a spring pulley system. Optimizing to find lowest deviation
% from desired curve using genetic algorithm optimization method

% Inputs:
%            R_i -          (scalar) lever length

%            guideLocX -    (scalar) horizontal distance from center
%                                    of motor pulley to center of
%                                    guide pulley

%            leverXval -    (scalar) horizontal distance from center
%                                    of motor pulley to lever base

%            leverYval -    (scalar) vertical distance from center
%                                    of motor pulley to lever base

%            theta_i -      (scalar) initial lever angle from horizontal

%            motorRadius -  (scalar) radius of fixed motor pulley

%            leverRadius -  (scalar) radius of moving lever pulley

%            guideRadius -  (scalar) radius of fixed guide pulley

% Outputs:  tensionArray - (vector) sequence of cable tension values
%                                    along spring pulley path

%            lengthArray -  (vector) sequence of change in cable
%                                    length from no-load state as
%                                    cable is pulled

%            thetaArray -   (vector) sequence of lever angles from
%                                    no-load state as cable is pulled
```

```

%           kArray -       (vector) sequence of mechanism stiffness
%                                   values along spring pulley path
%                                   (dTdL)

%           d2TdL2 -       (vector) sequence of derivative of
%                                   kArray values

%           ObjVal -       (scalar) objective function value

%           b -     (scalar) intercept of stiffness fit line

%           m -     (scalar) slope of stiffness fit line

%           B -     (scalar) min fit stiffness [N/mm]

%           D -     (scalar) max fit stiffness [N/mm]

%           usedL - (scalar) change in cable length at max fit
%                            stiffness [mm]

%           maxStrainindex -   (scalar) index of lengthArray at which
%                                       spring reaches max strain
%                                       [integer]

%           lengthMaxStrain -  (scalar) change in cable length at
%                                       maximum spring strain [mm]

%           springFreeLength - (scalar) free length of spring [mm]

%    subplots: T vs L
%              K vs L
%              C vs L
%              mechanism initial geometry

clearvars
close all
clc

%% Initialization of Parameters
    R_i = 30;                   % lever length
    guideLocX = 50;             % guide pulley location x-coordinate
    leverXval = 8;             % lever base location x-coordinate
    leverYval = 8;             % lever base location y-coordinate
    theta_i = 60;               % initial lever angle
    motorRadius = 7;            % radius of fixed motor pulley
    leverRadius = 7;            % radius of moving lever pulley
    guideRadius = 7;            % radius of fixed guide pulley
x0 = [R_i; guideLocX; leverXval; leverYval; theta_i; motorRadius;
    leverRadius; guideRadius];
%% Optimization Constraints
% A & b matrices prevent various inadmissable scenarios from happening

A = [];
```

```
b = [];
Aeq = [];
beq = [];
lb = [50, 40, 10, 11, 45, 7,  4,  4]';       % lowerbounds
ub = [50, 40, 15, 13, 60, 11, 11, 8]';     % upperbounds
%% Genetic Algorithm

options = optimoptions('ga','PopulationSize',1e6,'UseParallel',true,'MaxGenerations',...
    1000*length(x0),'FunctionTolerance',1e-9,...
    'PlotFcn',{@gaplotbestindiv,@gaplotbestf});

[X,FVAL,EXITFLAG,OUTPUT] = ga(@ObjFun_v3_scaled,8,A,b,Aeq,beq,lb,ub,[],options);

%% Output Results
% This portion reruns the results of the optimization in order to obtain the results in the workspace.
ResultsOut_v3_scaled(X)
[tensionArray, lengthArray, thetaArray, kArray, d2TdL2, ObjVal, b, m, B, D, usedL, maxStrainindex, lengthMaxStrain, springFreeLength] = LeverAnalysis_v3_scaled(X);
```

E.2 ObjFun_v3_scaled.m

```
% Description: Relays the objective function from the main loop
% LeverOptim_ga_v3_scaled.m to the internal kinematics script
% (LeverAnalysis_v3_scaled.m)
function e = ObjFun_v3_scaled(x0)

try
[~, ~, ~, ~, ~, e, ~, ~, ~, ~, ~, ~, ~, ~] = LeverAnalysis_v3_scaled(x0);

catch
    e = 100000;
end
```

E.3 LeverAnalysis_v3_scaled.m

```
% Inputs:   x0 - (vector) of 8 parameter values of mechanism

% Outputs:  tensionArray -    (vector) sequence of Cable Tension Values
%                                      along spring pulley path

%           lengthArray -     (vector) sequence of change in cable
%                                      lengths along spring pulley path
```

```

%           thetaArray -      (vector) sequence of lever angle values

%           kArray -          (vector) sequence of mechanism stiffness
%                                      values along spring pulley path
%                                      (dTdL)

%           d2TdL2 -          (vector) sequence of derivative of
%                                      kArray values

%           ObjVal -          (scalar) in order to minimize objective
%                                      value, maxDeviation must be low
%                                      and cable travel length must
%                                      be high

%           devMag2 -         (scalar) inner product of deviation
%                                      array of tension array values
%                                      vs. desired quadratic values

%           dev1Mag2 -        (scalar) inner product of deviation
%                                      array of stiffness array values
%                                      vs. desired linear values

%           dev2Mag2 -        (scalar) inner product of deviation
%                                      array of stiffness derivative
%                                      values vs. desired constant value

%           maxLength -       (scalar) maximum change in cable length
%                                      from initial cable length

%           lengthMaxStrain - (scalar) cable deflection value at which
%                                      the spring has reached its
%                                      specified maximum strain

function [tensionArray, lengthArray, thetaArray, kArray, d2TdL2, ObjVal,
    b, m, B, D, usedL, maxStrainindex, lengthMaxStrain, springFreeLength
        ] =
    LeverAnalysis_v3_scaled(x0)

K = 2.416;                         % [N/mm] spring constant (for 2 springs)
springFreeLength = 31.75;          % [mm] specific to selected spring
Tension_i = 6.22;                  % [Newtons] specific to selected spring
maxStrain = 0.79;                  % [unitless] specific to selected spring

R_i = x0(1);                       % lever length
leverXval = x0(3);                 % lever base location x-coordinate
leverYval = x0(4);                 % lever base location y-coordinate
theta_i = x0(5);                   % initial lever angle
motorRadius = x0(6);               % radius of fixed motor pulley
leverRadius = x0(7);               % radius of moving lever pulley
guideRadius = x0(8);               % radius of fixed guide pulley

% lever base location constraints
if sqrt(leverXval^2+leverYval^2) <= motorRadius + 5
```

```
    x0(3) = 0;
    leverYval = 0;
    x0(4) = 0;
else % nothing
end
%
% determine theta_f
if leverYval >= (R_i + leverRadius + motorRadius)
    theta_f = -89; % [°]
else
theta_f = -asind((leverYval - guideRadius - leverRadius)/(R_i)) + 5;
end

% create theta array & initialize tension/length arrays
thetaArray = linspace(theta_i,theta_f,1000);

[lengthArray, tensionArray, strainArray] =
   TensionDeflectionFun_v3_scaled(thetaArray, x0, K, springFreeLength,
   Tension_i);

for j = 2:length(lengthArray)
    if lengthArray(j) < lengthArray(j-1)
        lengthArray = lengthArray(1:j-1);
        tensionArray = tensionArray(1:j-1);
        strainArray = strainArray(1:j-1);
        break
    else % nothing
    end
end

if max(strainArray) <= maxStrain
    maxStrainindex = length(lengthArray);
else
maxStrainindex = find(strainArray <= maxStrain,1,'last');
end
lengthMaxStrain = lengthArray(maxStrainindex);

% array differentiation
  dT = diff(tensionArray);
  dL = diff(lengthArray);
% (derivative of tension) vs deflection = stiffness vs deflection curve
  kArray = dT./dL;
% (derivative of stiffness) vs deflection curve
  d2T = diff(kArray);
  d2TdL2 = d2T./dL(1:end-1);

k_i = find(kArray > 1.1*kArray(1),1);

% max motor torque at cable tension = 100 N
if tensionArray(maxStrainindex) <= 100
  % truncate length array at max strain before fitting line
    lengthArrayTrunc = lengthArray(k_i:(maxStrainindex-1));
  % truncate stiffness array at max strain before fitting line
    kArrayTrunc = kArray(k_i:(maxStrainindex-1));
```

```
else
    maxTensionindex = find(tensionArray <= 100,1,'last');
  % truncate length array at max tension
    lengthArrayTrunc = lengthArray(k_i:(maxTensionindex));
  % truncate stiffness array at max tension
    kArrayTrunc = kArray(k_i:(maxTensionindex));
end

% first bisection value for analysis
h = floor(length(lengthArrayTrunc)/2);
f = h;
% initialize iteration variable
  j = 1;

% 10 bisection iterations
while j < 11
% range of x values for fit line
  lengthArrayfit = lengthArrayTrunc(1:f);
% range of y values for fit line
  kArrayfit = kArrayTrunc(1:f);
% linear regression
  p = polyfit(lengthArrayfit, kArrayfit, 1);
% regression line y values
  yfit = polyval(p,lengthArrayfit);
% initialize error vector
  kErr = zeros(f,1);

% populate kErr vector of stiffness deviation from fit line
for i = 1:length(kErr)
    kErr(i) = (abs(kArrayfit(i) - yfit(i)))/yfit(i);
end
h=floor(h/2);

% check if stiffness values follow fit line within 10 percent
if max(kErr) <= 0.1
  % increase x value range if error less than or equal to 10 percent
    f = floor(f + h);
else
  % decrease x value range if error greater than 10 percent
    f = floor(f - h);
end

% index fit/error loop iteration
j=j+1;
end

b = p(2);                                 % fit intercept
m = p(1);                                 % fit slope
B = kArrayTrunc(1);                       % actual intercept
D = kArrayTrunc(length(kErr));            % max stiffness
usedL = lengthArrayTrunc(length(kErr));   % length at max stiffness

% objective value maximizes max stiffness (squared) and cable length,
% and minimizes initial stiffness
```

```
ObjVal = 1000*abs(B/((D^2)*usedL));

end
```

E.4 InitialGeomFun_v3_scaled.m

```
%  Inputs: x0 - (Vector) of 8 parameter values of mechanism

%  Outputs: motorCenter - (vector) position of center of motor pulley

%           leverPoint -  (vector) position of fixed end of lever

%           springPoint - (vector) position of fixed end of spring

%           guideCenter - (vector) position of center of guide pulley

%           cableLength - (scalar) total initial cable length

%% Function

function [motorCenter, leverPoint, springPoint, guideCenter, cableLength_i] = InitialGeomFun_v3_scaled(x,springFreeLength)

R_i = x(1);                 % lever length
guideLocX = x(2);           % guide pulley center x-coordinate
leverXval = x(3);           % lever base location x-coordinate
leverYval = x(4);           % lever base location y-coordinate
theta_i = x(5);             % initial lever angle
motorRadius = x(6);         % radius of fixed motor pulley
leverRadius = x(7);         % radius of moving lever pulley
guideRadius = x(8);         % radius of fixed guide pulley

% Finding Initial Orentiation
% calculates the geometry of the lever mechanism

% center point of motor pulley (output)
  motorCenter = [0, 0, 0]';
% center point of fixed guide pulley (output)
  guideCenter = [guideLocX, -(motorRadius-guideRadius), 0]';
% X&Y coordinates for bottom of guide pulley
  xGuideBottom = guideCenter(1);
  yGuideBottom = guideCenter(2)-guideRadius;
% fixed spring attachment point
  springPoint = [(leverXval+(R_i + springFreeLength)*cosd(theta_i)),
                (leverYval+(R_i + springFreeLength)*sind(theta_i)), 0]';

% X&Y Coordinate for bottom of motor pulley
  xMotorBottom = 0; yMotorBottom = -motorRadius;
% fixed lever base point
  leverPoint = [leverXval leverYval 0]';
```

```
% initial position of center point of spring pulley (output)
  leverPulleyCenter = [(leverPoint(1)+cosd(theta_i)*(R_i)),
                       (leverPoint(2)+sind(theta_i)*(R_i)), 0]';

% calculate tangent points and vector from motor pulley to spring pulley
  [xMotorTangent, yMotorTangent, xSpringTangentL, ySpringTangentL,
     tangentMotorToSpring] = crosstan(motorCenter,motorRadius,
     leverRadius,leverPulleyCenter,1);
% Flip direction of motor to spring tangent vector
  tangentMotorToSpring=-tangentMotorToSpring;
% magnitude of cable length from motor pulley to spring pulley
  cableLengthMotorToSpring = norm(tangentMotorToSpring);
% calculate tangent points and vector from spring pulley to guide pulley
  [xSpringTangentR, ySpringTangentR, xGuideTangent, yGuideTangent,
     tangentSpringToGuide] = crosstan(leverPulleyCenter,leverRadius,
     guideRadius,(guideCenter-leverPulleyCenter),0);
% magnitude of cable length from spring pulley to guide pulley
  cableLengthSpringToGuide = norm(tangentSpringToGuide);

% calculate the cable contact length on the lever pulley
  leverPulleyRayLeft = [xSpringTangentL-leverPulleyCenter(1)
                        ySpringTangentL-leverPulleyCenter(2) 0];
% vector from center of spring pulley to tangent point where the
% cable from the motor pulley contacts the lever pulley
  leverPulleyRayRight = [xSpringTangentR-leverPulleyCenter(1)
                         ySpringTangentR-leverPulleyCenter(2) 0];
% vector from center of spring pulley to tangent point where the
% cable to the guide pulley contacts the lever pulley
  leverPulleyAngleRight = atan2d(leverPulleyRayRight(2),
                                 leverPulleyRayRight(1));
  leverPulleyAngleLeft = atan2d(leverPulleyRayLeft(2),
                                leverPulleyRayLeft(1));
if leverPulleyAngleLeft<0
    leverPulleyAngleLeft = leverPulleyAngleLeft+360;
end

% calculate arcangle of cable contact on the spring pulley
  leverPulleyArcangle = leverPulleyAngleLeft-leverPulleyAngleRight;
% calculate normalized length of cable contact on spring pulley
  leverPulleyArclength = pi*leverRadius*(leverPulleyArcangle/180);

% calculate the cable contact length on the motor pulley
  motorRayLeft = [motorCenter(1)-xMotorBottom
                  motorCenter(2)-yMotorBottom 0];
  motorRayRight = [motorCenter(1)-xMotorTangent
                   motorCenter(2)-yMotorTangent 0];
  motorArcangle = atan2d(norm(cross(motorRayLeft,motorRayRight)),
                         dot(motorRayLeft,motorRayRight));
  motorArclength = pi*motorRadius*(motorArcangle/180);

% calculate the cable contact length on the guide pulley
  guideRayRight = [guideCenter(1)-xGuideBottom
                   guideCenter(2)-yGuideBottom 0];
  guideRayLeft = [guideCenter(1)-xGuideTangent
```

```
                  guideCenter(2)-yGuideTangent 0];
  guideArcangle = atan2d(norm(cross(guideRayRight,guideRayLeft)),
                         dot(guideRayRight,guideRayLeft));
  guideArclength = pi*guideRadius*(guideArcangle/180);

% total initial cable length in mechanism
  cableLength_i = leverPulleyArclength + cableLengthMotorToSpring +
                  cableLengthSpringToGuide + motorArclength +
                  guideArclength;
end
```

E.5 TensionDeflectionFun_v3_scaled.m

```
% Inputs: theta - (vector) series of angles between horizontal
%                          x-axis and lever

%         x0 -    (vector) of 8 parameter values of mechanism

% Outputs: lengthArray -  (vector) deflection of cable from initial
%                                  length at each lever theta value

%          tensionArray - (vector) magnitudes of the tension in
%                                  cable at each cable length value

%          strainArray - (vector) strain in string at each cable
%                                 length value

function [lengthArray, tensionArray, strainArray] =
   TensionDeflectionFun_v3_scaled(theta, x, K, springFreeLength,
   Tension_i);
% Parameters
R_i = x(1);                 % lever length
leverXval = x(3);           % lever termination point x-coordinate
leverYval = x(4);           % lever termination point y-coordinate
motorRadius = x(6);         % radius of fixed motor pulley
leverRadius = x(7);         % radius of moving lever pulley
guideRadius = x(8);         % radius of fixed guide pulley

% Geometry
[motorCenter, leverPoint, springPoint, guideCenter, cableLength_i] =
   InitialGeomFun_v3_scaled(x,springFreeLength);

% Initialize arrays
tensionArray = zeros(length(theta)-2,1);
lengthArray = zeros(length(theta)-2,1);
strainArray = zeros(length(theta)-2,1);

for i = 1:length(theta)-2
    leverPulleyCenter = [(leverPoint(1) + cosd(theta(i))*(R_i)),
                         (leverPoint(2) + sind(theta(i))*(R_i)), 0]';
```

```
   springAttachmentPoint = [(leverXval+(R_i*cosd(theta(i)))),
                            (leverYval+(R_i*sind(theta(i)))), 0]';
 % Find points where cable contacts motor pulley and lever pulley
   [xMotorTangent, yMotorTangent, xLeverTangentL, yLeverTangentL,
      tangentMotorToLever, ~, ~] = crosstan(motorCenter,motorRadius,
      leverRadius,leverPulleyCenter,1);
   xMotorBottom = 0; yMotorBottom = -motorRadius; xGuideBottom =
      guideCenter(1); yGuideBottom = guideCenter(2)-guideRadius;
   cableLengthMotorToLever = norm(tangentMotorToLever);
   tangentMotorToLever=-tangentMotorToLever;
 % Find points where cable contacts lever pulley and guide pulley
   [xLeverTangentR, yLeverTangentR, xGuideTangent, yGuideTangent,
      tangentLeverToGuide, ~, ~] = crosstan(leverPulleyCenter,
      leverRadius,guideRadius,(guideCenter-leverPulleyCenter),0);

 % Angle and arclength calculations to determine length
 % of cable in system
   cableLengthLeverToGuide = norm(tangentLeverToGuide);
   leverPulleyRayLeft = [xLeverTangentL-leverPulleyCenter(1)
                         yLeverTangentL-leverPulleyCenter(2) 0];
   leverPulleyRayRight = [xLeverTangentR-leverPulleyCenter(1)
                          yLeverTangentR-leverPulleyCenter(2) 0];
   leverPulleyAngleRight = atan2d(leverPulleyRayRight(2),
                                  leverPulleyRayRight(1));
   leverPulleyAngleLeft = atan2d(leverPulleyRayLeft(2),
                                 leverPulleyRayLeft(1));
   if leverPulleyAngleLeft<0
       leverPulleyAngleLeft = leverPulleyAngleLeft+360;
   end
   leverPulleyArcangle = leverPulleyAngleLeft-leverPulleyAngleRight;
   leverPulleyArclength = pi*leverRadius*(leverPulleyArcangle/180);
   motorRayLeft = [motorCenter(1)-xMotorBottom
                   motorCenter(2)-yMotorBottom 0];
   motorRayRight = [motorCenter(1)-xMotorTangent
                    motorCenter(2)-yMotorTangent 0];
   motorArcangle = atan2d(norm(cross(motorRayLeft,motorRayRight)),
   dot(motorRayLeft,motorRayRight));
   motorArclength = pi*motorRadius*(motorArcangle/180);
   guideRayRight = [guideCenter(1)-xGuideBottom
                    guideCenter(2)-yGuideBottom 0];
   guideRayLeft = [guideCenter(1)-xGuideTangent
                   guideCenter(2)-yGuideTangent 0];
   guideArcangle = atan2d(norm(cross(guideRayRight,guideRayLeft)),
   dot(guideRayRight,guideRayLeft));
   guideArclength = pi*guideRadius*(guideArcangle/180);

% Current cable length
   cableLengthNew = cableLengthMotorToLever + leverPulleyArclength
   + cableLengthLeverToGuide + motorArclength + guideArclength;
% deflection of cable length in current position from initial position
   deflection = cableLength_i - cableLengthNew;

% Unit vectors
 % Direction of the tension from the motor pulley to spring pulley
```

```
    unitTensionVector1 = tangentMotorToLever/norm(tangentMotorToLever);
  % Direction of the tension from the spring pulley to guide pulley
    unitTensionVector2 = tangentLeverToGuide/norm(tangentLeverToGuide);
  % current vector from spring pulley center to fixed spring
  % attachment point
    currentSpringVector = springPoint - springAttachmentPoint;
  % Direction of spring force
    unitSpringVector = currentSpringVector/norm(currentSpringVector);
  % current vector from lever pulley center to fixed lever
  % attachment point
    normalVector = leverPulleyCenter - leverPoint;

  % vector from fixed lever attachment point to the point where the
  % cable contacts the lever pulley on the left side
    rVector1 = [xLeverTangentL; yLeverTangentL; 0] - leverPoint;
  % vector from fixed lever attachment point to the point where the
  % cable contacts the lever pulley on the right side
    rVector2 = [xLeverTangentR; yLeverTangentR; 0] - leverPoint;

 % Spring force
  % current deflection of spring from free length
    springDeflection = norm(currentSpringVector) - springFreeLength;
  % current strain in spring
    springStrain = springDeflection/springFreeLength;
  % Force applied to spring pulley due to current spring tension
    springForce = K*(springDeflection) + Tension_i;

 % Given known positions of the pulleys and the force applied by the
 % spring, the current tension in the cable can be calculated.
 % A*x = b from sum of moments = 0
    A = cross(rVector1, unitTensionVector1) +
        cross(rVector2, unitTensionVector2);
    b = -cross(normalVector, unitSpringVector*springForce);
    tensionMagnitude = b(3)/A(3);

lengthArray(i) = deflection;
tensionArray(i) = tensionMagnitude;
strainArray(i) = springStrain;

end
end
```

E.6 ResultsOut_v3_scaled.m

```matlab
function  ResultsOut_v3_scaled(x)

close all

% Results
[tensionArray, lengthArray, thetaArray, kArray, d2TdL2, ObjVal, b, m, B, D, usedL, ~, lengthMaxStrain, springFreeLength] = LeverAnalysis_v3_scaled(x);

% display outputs when function runs
  magnitude = D/B;
% stiffness magnitude change
  disp(['magnitude = ' num2str(magnitude)])
% length at max allowable spring deflection
  disp(['usedL = ' num2str(usedL)])
% equation of fit line from bisection loop
  sprintf('stiffnessEq: y = %d*x + %d',m,b)


% Geometry
[motorCenter, leverPoint, springPoint, guideCenter, ~] = InitialGeomFun_v3_scaled(x,springFreeLength);
    R_i = x(1);
    theta_i = x(5);
    leverPulleyCenter_i = [(leverPoint(1)+(R_i)*cosd(theta_i)),
    (leverPoint(2)+(R_i)*sind(theta_i)),0]';
    motorRadius = x(6);
    leverRadius = x(7);
    guideRadius = x(8);
    theta_used = thetaArray(find(lengthArray <= usedL,1,'last'));
    %theta_max = thetaArray(maxStrainindex);

% plot T vs L against desired quadratic function
figure(1)
subplot(1,2,1)
plot(lengthArray,tensionArray,lengthArray,((m/2)*lengthArray.^2 +
    b.*lengthArray),'LineWidth',1.25);
axis([0 (lengthMaxStrain + 2) 0 150])
line([lengthMaxStrain lengthMaxStrain],[0 1000],'Color','r',
    'LineStyle','--');
hh=text(23.7,20,'Max strain','FontName','times','FontSize',10);
set(hh,'Rotation',90);
title('Tension vs Deflection','FontName','times','FontSize',11)
xlabel('\DeltaL (mm)','FontName','times','FontSize',11)
ylabel('Tension (N)','FontName','times','FontSize',11)
legend({'Calculated','Desired'},'FontName','times','FontSize',10,
    'Location','Northwest')
set(gca,'FontName','times','FontSize',11)

% plot K vs L against desired linear function
subplot(1,2,2)
```

```
plot(lengthArray(1:end-1),kArray,lengthArray(1:end-1),(b +
    m*lengthArray(1:end-1)),'LineWidth',1.25)
axis([0 (lengthMaxStrain + 2) 0 15])
line([lengthMaxStrain lengthMaxStrain],[0 1000],'Color','r',
    'LineStyle','--');
line([usedL usedL],[1.1*B D],'Color',[0.5,0.5,0.5],'LineStyle','-');
line([2.15 usedL],[1.1*B 1.1*B],'Color',[0.5,0.5,0.5],'LineStyle','-');
line([2.15 usedL],[D D],'Color',[0.5,0.5,0.5],'LineStyle','-');
line([2.15 2.15],[1.1*B D],'Color',[0.5,0.5,0.5],'LineStyle','-');
h=text(23.7,10,'Max strain','FontName','times','FontSize',10);
set(h,'Rotation',90);
title('Stiffness vs Deflection','FontName','times','FontSize',11)
xlabel('\DeltaL (mm)','FontName','times','FontSize',11)
ylabel('Stiffness (N/mm)','FontName','times','FontSize',11)
legend({'Calculated','Desired'},'FontName','times','FontSize',10,
    'Location','Northwest')
set(gca,'FontName','times','FontSize',11)

% plot initial system orientation
figure(2)
title({['ObjVal = ',num2str((ObjVal),'%.5f')] ['R_i=',num2str(x(1),
    '%.2f'), ', x_G=',num2str(x(2),'%.2f'), ', x_L=',num2str(x(3),
    '%.2f'), ', y_L=',num2str(x(4),'%.2f') ', \theta_i=',num2str(x(5),
    '%.2f'), ', R_M=',num2str(x(6),'%.2f'), ', R_L=',num2str(x(7),
    '%.2f'), ', R_G=',num2str(x(8),'%.2f')]},'FontName',
    'times','FontSize',11)

% pulleys
centers = [motorCenter(1), motorCenter(2); leverPulleyCenter_i(1),
   leverPulleyCenter_i(2); guideCenter(1), guideCenter(2)];
radii = [motorRadius; leverRadius; guideRadius];
viscircles(centers,radii,'LineWidth',2,'LineStyle','-');

% lever
line1X = [leverPoint(1) leverPoint(1)+x(1)*cosd(theta_i)];
line1Y = [leverPoint(2) leverPoint(2)+x(1)*sind(theta_i)];
line(line1X,line1Y,'Color','black','LineWidth',3);
line2X = [leverPoint(1) leverPoint(1)+x(1)*cosd(theta_used)];
line2Y = [leverPoint(2) leverPoint(2)+x(1)*sind(theta_used)];
line(line2X,line2Y,'Color',[0.4,0.4,0.4],'LineWidth',0.5);
viscircles([(leverPoint(1)+x(1)*cosd(theta_used)),(leverPoint(2)+
x(1)*sind(theta_used))],leverRadius,'Color',[.8,0,0],'LineWidth',0.5);

% spring
line3X = [leverPoint(1)+x(1)*cosd(theta_i) springPoint(1)];
line3Y = [leverPoint(2)+x(1)*sind(theta_i) springPoint(2)];
line(line3X,line3Y,'Color','green','LineStyle',':','LineWidth',2);
line4X = [line2X(2) springPoint(1)];
line4Y = [line2Y(2) springPoint(2)];
line(line4X,line4Y,'Color','green','LineStyle',':','LineWidth',0.5);
hold on

% lever termination point
plot(leverPoint(1),leverPoint(2),'Marker','square','MarkerSize',8,
```

```
    'MarkerFaceColor','black','Color','black');

% spring termination point
plot(springPoint(1),springPoint(2),'Marker','square','MarkerSize',8,
    'MarkerFaceColor','black','Color','black');
xlim([-20 90]); ylim([-20 90]);
axis square
set(gca,'FontName','times','FontSize',11)
xlabel('x (mm)','FontName','times','FontSize',11)
ylabel('y (mm)','FontName','times','FontSize',11)

end
```

E.7 crosstan.m

```
% Author: Ryan Moore
% Description: Calculates the crossed tangent points between two
   circles.
%This is used to find the tangent points of the cable and the pulleys as
    well as the force vector of the cable tensions.
%This function is called in TensionDeflectionFun_v3_scaled.m.

% Inputs:   C1 - (vector) of center point of the first circle (pulley)

%           r1 - (scalar) radius of first circle (pulley)

%           r2 - (scalar) radius of second circle (pulley)

%           L  - (vector) of length between the two circles

%           tb - (scalar) "top or bottom" crossed tangent. 1 is used
%                         from the cable between the motor and lever
%                         pulley and 0 is used from the cable between
%                         the lever and guide pulley.

% Outputs:  x1 - (scalar) x location of the tangent point on first
%                         circle

%           y1 - (scalar) y location of the tangent point on first
%                         circle

%           x2 - (scalar) x location of the tangent point on second
%                         circle

%           y2 - (scalar) y location of the tangent point on second
%                         circle

%           T  - (vector) of crossed tangent from first circle to
%                         second circle from (x1,y1) to (x2,y2)

```

```
function [x1 y1 x2 y2 T alpha angle] = crosstan(C1,r1,r2,L,tb)
%% cross tangents to two circles
x = [50, 0, 0]';                     %  Establishes global x direction
C2 = L+C1;                           %  Center of second circle
d2 = L/(1+r1/r2);                    %  Distance between intersection
%                                       point of tangent line and x axis
%                                       to the center of the second
%                                       circle
d1 = L-d2;                           %  Distance between the center of
%                                       the first circle to the
%                                       intersection point of tangent
%                                       line and x axis.

cros = cross(x,L);
angle = atan2d(cros(3), dot(x,L));   %  Angle between x axis and actual
%                                       vector from one circle center to
%                                       the other
alpha = acosd((r1+r2)/norm(L));      %  Angle between vector from one
%                                       circle center to the other and
%                                       tangent point of cable on
   circle.
if tb > 0
    alpha = -alpha;                  %  Flips angle if using the other
%                                       crossed tangent line
end
alpha = alpha+angle;                 %  adds angles together to achieve
%                                       true angle of vector from
%                                       global x-axis
x1=r1*cosd(alpha)+C1(1);             %  x location of tangent point
%                                       on first circle
y1=r1*sind(alpha)+C1(2);             %  y location of tangent point
%                                       on first circle
beta=180+alpha;                      %  rotates alpha by 180 degrees to
%                                       calculate other circle points
x2=L(1)+r2*cosd(beta)+C1(1);         %  x location of tangent point
%                                       on second circle
y2=L(2)+r2*sind(beta)+C1(2);         %  y location of tangent point
%                                       on second circle
T = [x2-x1; y2-y1; 0];               %  Vector of crossed tangent from
%                                       first circle to second circle.
```

APPENDIX F

ANALYTICAL ENDPOINT DEFLECTION

This appendix contains the MATLAB code used in the calculation and visualization of the theoretical endpoint deflection. This is achieved using the endpoint load, joint angles and compliances, and estimated joint friction, as described in Chapter 5.

F.1 Deflection_analytical.m

```

% Inputs:   configuration -  (scalar) 1, 2, 3, or 4 corresponding to
%                               configurations (a), (b), (c), and (d),
%                               respectively

%           c1, c2, c3 -     (scalars) compliance of each joint

%           Q1, Q2, Q3 -     (scalars) nominal commanded joint angles

%           ft1, ft2, ft3 -  (scalars) friction torque of each joint


% Outputs:  Endpoints -      (matrix) i x j matrix of row vectors where
%                                     i = 1:2n for n loading angles and
%                                     (row i contains x-coordinates and
%                                      row i+1 contains y-coordinates)
%                                     j = 1:m+1 for m weight increments
%                                     (column 1 is initial endpoint
%                                     coordinates, columns 2:m+1 are
%                                     endpoint coordinates at each
%                                     weight increment


green = '#77AC30';
blue = '#1E95E3';
purple = '#7E2F8E';
orange = '#D95319';

config = 1; % 1, 2, 3, or 4

%% Enter joint compliance & angles

% compliance [°/N*mm] and initial joint angles [°]

% [4 0; 0 4]
if config == 1
c1 = 0.06353;
c2 = 0.06472;
```

```
c3 = 0.44444;
Q1 = 65;
Q2 = -75;
Q3 = -75;
ft1 = 68.02;
ft2 = 7.27;
ft3 = 33.2;

% % [3 0; 0 12]
elseif config == 2
c1 = 0.06916;
c2 = 0.18939;
c3 = 0.36900;
Q1 = 60;
Q2 = -50;
Q3 = -60;
ft1 = 56.36;
ft2 = 0.74;
ft3 = 10.33;

% [15 0; 0 0.15]
 elseif config == 3
c1 = 0.03505;
c2 = 0.20504;
c3 = 0.21593;
Q1 = 75;
Q2 = 30;
Q3 = -25;
ft1 = 207.55;
ft2 = 1.36;
ft3 = 21.9;

% [3 2; 2 6]
elseif config == 4
c1 = 0.04288;
c2 = 0.17036;
c3 = 0.16835;
Q1 = 70;
Q2 = -80;
Q3 = -60;
ft1 = 145.6;
ft2 = 6.7;
ft3 = 31.52;

end

%% Plot finger orientation

% segment lengths [mm]
l1 = 48.5;
l2 = 29.5;
l3 = 19.5;

% plot finger segments
```

```
J1 = [0; 0];
J2 = l1*[cosd(Q1); sind(Q1)];
J3 = J2 + l2*[(cosd(Q1+Q2)); (sind(Q1+Q2))];
O = J3 + l3*[(cosd(Q1+Q2+Q3)); (sind(Q1+Q2+Q3))];
line([J1(1) J2(1)],[J1(2) J2(2)],'LineWidth',3,'Color','k')
hold on
line([J2(1) J3(1)],[J2(2) J3(2)],'LineWidth',3,'Color','k')
line([J3(1) O(1)],[J3(2) O(2)],'LineWidth',3,'Color','k')
hold on

% plot origin and endpoint
plot(J1(1),J1(2),'.','MarkerSize',16,'Color','k')
plot(O(1),O(2),'.','MarkerSize',16,'Color','k')
plot(J2(1),J2(2),'.','MarkerSize',16,'Color','k')
plot(J3(1),J3(2),'.','MarkerSize',16,'Color','k')
hold on
if config == 1
xlim([-10 70]); ylim([-10 70]);
axis square
end
if config == 2
    xlim([-10 90]); ylim([-10 90]);
    axis square
end
if config == 3
    xlim([-50 80]); ylim([-10 120]);
    axis square
end
if config == 4
    xlim([-10 70]); ylim([-10 70])
    axis square
end
xlabel('x (mm)'), ylabel('y (mm)')
set(gca,'FontName','times','FontSize',10,'TitleFontSizeMultiplier',1)

%% Loading

% number of increments
N = 5;

% weight (g)
maxweight = 500;

% direction (°)
if config == 3
   angles = [-90 -60 -30 0 30 60 90];
elseif config == 1 || 2 || 4
    angles = [-120 -90 -60 -30 0 30 60 90 120];
end

Torques = zeros(3*length(angles),N+1);
Endpoints = zeros(2*length(angles),N+1);

for m = 1:length(angles)
```

```
angle = angles(m);

% initial joint angles [°]
q1 = Q1;
q2 = Q2;
q3 = Q3;

for i = 1:N+1
weight = ((i-1)/N)*maxweight;
[Torque, dq] = incremental(weight,angle,c1,c2,c3,q1,q2,q3,ft1,ft2,ft3);

Torques(3*m-2,i) = Torque(1);
Torques(3*m-1,i) = Torque(2);
Torques(3*m,i) = Torque(3);

q1 = Q1+dq(1);
q2 = Q2+dq(2);
q3 = Q3+dq(3);

Endpoints(2*m-1,i) = l1*cosd(q1) + l2*cosd(q1+q2) + l3*cosd(q1+q2+q3);
Endpoints(2*m,i) = l1*sind(q1) + l2*sind(q1+q2) + l3*sind(q1+q2+q3);
plot(Endpoints(2*m-1,i),Endpoints(2*m,i),'.','MarkerEdgeColor',orange)
hold on
end
end

for n = 2:N+1
line([Endpoints(1,n), Endpoints(3,n)],
     [Endpoints(2,n), Endpoints(4,n)],'Color',orange)
line([Endpoints(3,n), Endpoints(5,n)],
     [Endpoints(4,n), Endpoints(6,n)],'Color',orange)
line([Endpoints(5,n), Endpoints(7,n)],
     [Endpoints(6,n), Endpoints(8,n)],'Color',orange)
line([Endpoints(7,n), Endpoints(9,n)],
     [Endpoints(8,n), Endpoints(10,n)],'Color',orange)
line([Endpoints(9,n) Endpoints(11,n)],
     [Endpoints(10,n) Endpoints(12,n)],'Color',orange)
line([Endpoints(11,n) Endpoints(13,n)],
     [Endpoints(12,n) Endpoints(14,n)],'Color',orange)
if config == 3
    % do nothing
elseif config == 1 || 2 || 4
   line([Endpoints(13,n) Endpoints(15,n)],
        [Endpoints(14,n) Endpoints(16,n)],'Color',orange)
   line([Endpoints(15,n) Endpoints(17,n)],
        [Endpoints(16,n), Endpoints(18,n)],'Color',orange)
end
end

if config == 1
    title({['Configuration (a)'],
           ['Analytical Endpoint Deflection']},'FontSize',11)
elseif config == 2
    title({['Configuration (b)'],
```

```
           ['Analytical Endpoint Deflection']},'FontSize',11)
elseif config == 3
    title({['Configuration (c)'],
           ['Analytical Endpoint Deflection']},'FontSize',11)
elseif config == 4
    title({['Configuration (d)'],
           ['Analytical Endpoint Deflection']},'FontSize',11)
end
set(gca,'Color','none');
set(gcf,'units','inches','position',[6,3,2.5,2.5])
```

F.2 incremental.m

```
% Inputs:   weight -         (scalar) weight applied to endpoint

%           angle -          (scalar) angle of weight applied to
%                                     endpoint

%           c1, c2, c3 -     (scalars) compliance of each joint

%           q1, q2, q3 -     (scalars) current joint angles

%           ft1, ft2, ft3 -  (scalars) friction torque of each joint


% Outputs:  Torquenew -      (vector) adjusted torque at each joint

%           dq -             (vector) angular deflection of each
%                                     joint

function [Torquenew, dq] = incremental(weight,angle,c1,c2,c3,q1,q2,q3,
    ft1,ft2,ft3)

% Compliance [°/N*mm]
C = [c1 0 0; 0 c2 0; 0 0 c3];

% Force vector [N]
fx = weight*0.0098*cosd(angle);
fy = weight*0.0098*sind(angle);
f = [fx; fy];

% segment lengths [mm]
l1 = 48.5;
l2 = 29.5;
l3 = 19.5;

% endpoint coordinates [mm]
x = l1*cosd(q1) + l2*cosd(q1+q2) + l3*cosd(q1+q2+q3);
y = l1*sind(q1) + l2*sind(q1+q2) + l3*sind(q1+q2+q3);

% endpoint coordinates vector [mm]
X = [x; y];
```

```
% Jacobian
J = [-l1*sind(q1)-l2*sind(q1+q2)-l3*sind(q1+q2+q3),
    -l2*sind(q1+q2)-l3*sind(q1+q2+q3),
    -l3*sind(q1+q2+q3);
    l1*cosd(q1)+l2*cosd(q1+q2)+l3*cosd(q1+q2+q3),
    l2*cosd(q1+q2)+l3*cosd(q1+q2+q3),
    l3*cosd(q1+q2+q3)];

Torquenew = zeros(3,1);

% Torque on each joint, neglect torque due to link weight [N*mm]
Torque = J'*f;
if sign(Torque(1)) >= 0 && Torque(1) >= ft1
Torquenew(1) = Torque(1) - ft1;
elseif sign(Torque(1)) >= 0 && Torque(1) < ft1
    Torquenew(1) = 0;
else, Torquenew(1) = Torque(1) + ft1;
end

if sign(Torque(2)) >= 0 && Torque(1) >= ft2
Torquenew(2) = Torque(2) - ft2;
elseif sign(Torque(2)) >=0 && Torque(2) < ft2
    Torquenew(2) = 0;
else, Torquenew(2) = Torque(2) + ft2;
end

if sign(Torque(3)) >= 0 && Torque(3) >= ft3
Torquenew(3) = Torque(3) - ft3;
elseif sign(Torque(3)) >= 0 && Torque(3) < ft3
    Torquenew(3) = 0;
else, Torquenew(3) = Torque(3) + ft3;
end

% Joint deflection [°]
dq = C*Torquenew;

end
```