TECHNISCHE UNIVERSITÄT WIEN

AUTOMATION & CONTROL INSTITUTE
INSTITUT FÜR AUTOMATISIERUNGS- & REGELUNGSTECHNIK

Design, Implementation and Validation of a Testbed for Elastic Robot Joints

BACHELOR THESIS

Conducted in partial fulfillment of the requirements for the degree of a

Bachelor of Science (BSc)

supervised by

Univ.-Prof. Dr.-Ing. Christian Ott
Annika Kirner, M.Sc.
Tobias Egle, M.Sc.

submitted at the

TU Wien
Faculty of Electrical Engineering and Information Technology
Automation and Control Institute

by

Patrick Pably
Matriculation number 11826211

Vienna, May 2025

Robotic Systems Lab (RSL)
A-1040 Wien, Gußhausstr. 27–29, Internet: https://www.acin.tuwien.ac.at/robotic-systems-lab/

Abstract

This thesis presents the design, implementation, and validation of a modular testbed for evaluating torsional springs in Series Elastic Actuators (SEAs), targeting their application in compliant robotic joints. SEAs offer intrinsic mechanical compliance through a spring integrated in series between actuator and link, enabling improved force control, shock tolerance, and energy efficiency. The developed testbed supports a static configuration in which a force torque sensor is connected to the testbed, making the link rigid. This enables the deflection of a spring, and further the evaluation of stiffness and energy loss by capturing a springs hysteresis. Furthermore, in dynamic configuration, this sensor is removed, allowing the link rotate freely and evaluate a springs dynamic damping behavior as well as the evaluation of custom controller strategies.

The core objective was to build a low-cost, test platform with support for both static and dynamic spring testing to evaluate torsional springs for Series Elastic Actuators in elastic robot joints. It shall be able to accommodate different spring sizes and be modular in nature. Additionally, a real-time capable software framework to facilitate communication with an actuator and sensors shall be provided to enable testings and incorporate custom user parameters for various tests including static, dynamic and control evaluation.

The testbed was built and evaluated using three torsional springs generated with an open-source MATLAB-based spring design tool [1]. These springs have been optimized for fused deposition modeling 3D printing and manufactured out of PLA, PETG, and PC-ABS. Results showed that PC-ABS performs best across most metrics, demonstrating linear stiffness and minimal damping. However, the evaluation also revealed mechanical limitations, such as compliance in the drive shaft and communication delays in the actuator, which currently limit the system's suitability for real-time control due to a low control frequency. Requiring replacement of the actuator to improve dynamic testing and a change of mechanical linkage or isolation of the spring measurement to improve the static test range. Nevertheless, the testbed provides a validated foundation for future research on compliant actuator design, control strategies, and spring optimization in robotic systems.

Contents

List of Figures

List of Tables

Symbols and notation

Symbol / Term	Image	Description
Actuator		motor, mechanical symbol
θ	-	theta, motor output shaft angle
Encoder		rotary encoder, mechanical symbol
$\mathbf{q}$	-	q, link angle
Spring		elastic element, torsional spring, mechanical symbol
FTS		Force Torque Sensor, mechanical symbol
Bearing		ball bearing, mechanical symbol

Table 1: Symbolic index

1 Introduction

1.1 Motivation

Series Elastic Actuators (SEAs) have emerged as a key technology in the field of compliant actuation, enabling enhanced safety, energy efficiency, and robustness in robotic systems, as they are needed in modern robotic systems such as Humanoids or Cobots [2, 3]. By introducing an elastic element between the motor and the load, SEAs allow for precise force control, mechanical shock absorption, and energy storage during cyclic motions. These characteristics are particularly beneficial in dynamic and unstructured environments, such as legged locomotion and manipulation of heavy or unpredictable loads.

Recent developments at RSL have focused on the design and integration of customized torsional springs into SEAs. Inspired by the automated spring design tools presented in [1], a 3D-printed spring has been evaluated in preliminary trials. These tests, conducted for a small-scale robotic platform named *Bruce*[4], demonstrated promising results. This sparked the need for a dedicated single-joint testbed to evaluate torsional springs. Such a platform is essential to support the iterative design process and to validate component behavior before system-level integration. This includes the means to testet control strategies for systems including an SEA.

However, the testbed proposed in this thesis will not be restricted to the specifications of the mentioned *Bruce* robot. Instead, it will be designed to accommodate a wider array of spring properties, including those suitable for rough manipulation like hammering, as well as springs for energy-efficient cyclic motions. Literature suggests that most SEAs developed for lower-limb exoskeletons and hip joints in wearable or legged robotics operate within a stiffness range of approximately up to 800 Nm/rad, with some systems reaching up to 1600 Nm/rad for high torque transmission [5].

1.2 Objective

The objective of this thesis is the design, implementation, and evaluation of a modular single-joint testbed for the characterization of torsional springs in Series Elastic Actuators (SEAs). In addition, some springs shall be designed and evaluated using this testbed. Optionally, a basic controller may be implemented to demonstrate the system's applicability for control integration tasks.

The design phase involves the mechanical concept for the testbed and the selection of a suitable motor and appropriate sensors. A suitable design for a spring shall be created

with the above-mentioned tool [1] and adapted to be compatible with this testbed. The testbed shall be modular in nature, allowing to test various torsional spring designs, and shall require minimal effort to integrate them.

Furthermore, software for actuator and sensor communication shall be designed to enable the required testing. The testbed shall resemble a robotic joint using an SEA, thus the software shall provide the means for testing control strategies for such a setup. Therefore, implementing the software will require a real-time capable system.

Due to the importance of cost-effectiveness in this work, existing components and 3D printing will be utilized as much as possible. Key components including a Raspberry Pi 5 to host the software module, a powerful actuator, a 6-axis Force Torque Sensor (FTS), and modular aluminum profiles to host the hardware are already available on site and will be utilized in this testbed.

Spring designs will be generated using the automated torsion spring tool introduced in [1] and modified to fit subtractive and additive manufacturing workflows. A set of example springs will be fabricated using FDM 3D printing to evaluate the feasibility of producing linear torsional springs with this method. These springs shall be tested with this testbed to determine their static characteristics, such as static stiffness, hysteresis, and energy losses, as well as for their dynamic properties such as damping behavior.

2 Related Work

2.1 Series Elastic Actuators (SEAs)

Modern robotic systems, such as collaborative and humanoid robots, often require safe and adaptive interaction with unstructured environments. Traditionally, such systems use rigid actuators, where compliance is introduced through control strategies like impedance or admittance control [3]. Although this allows for modulating the response of the system, it lacks intrinsic mechanical compliance and limits energy efficiency and shock tolerance.

In contrast, when introducing an elastic element between the motor and the load the inherent compliance transforms the actuator into a compliant force source [2]. This is called an elastic robot joint and can enable more accurate force control, mechanical shock absorption, and the ability to store and release mechanical energy during cyclic tasks such as locomotion or repetitive manipulation [6, 7]. These benefits are particularly advantageous in tasks involving physical human-robot interaction, operation in unpredictable environments, or harsh manipulation tasks.

2.1.1 SEA Architectures and Use Cases

In this context, two main design concepts exist: Series Elastic Actuators (SEAs) and Variable Stiffness Actuators (VSAs). In SEAs, a passive elastic element is placed in series between the motor and the load. This elasticity is fixed and determined solely by the mechanical properties of the elastic element [2]. In contrast, VSAs incorporate mechanisms that allow active stiffness adjustment during operation, offering greater adaptability to varying task requirements [3, 8].

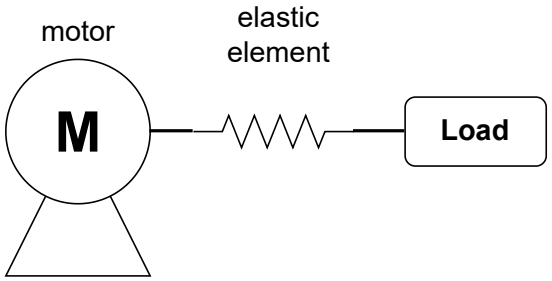

Figure 2.1: Series elastic actuator (SEA) sketch

While VSAs can modulate compliance in real time, they are typically more complex, heavier, and costly. SEAs, on the other hand, offer a simpler and more predictable structure, making them particularly attractive for applications requiring high force control fidelity, safety, and energy efficiency. SEAs can be broadly categorized by the geometry of the integrated elastic component:

- **Linear SEAs**, where a linear spring is used to provide compliance along a translation axis. These are commonly found in rehabilitation robotics and industrial applications.
- **Torsional SEAs**, where a torsion spring is placed between the motor and the output. These are ideal for rotational joints and are therefore widely used in legged robots and wearable devices [5].

This thesis focuses on the design and assessment of constant stiffness torsional SEAs.

2.1.2 Advantages and Disadvantages of SEAs

The primary benefits of SEAs include their natural mechanical compliance, enhanced force detection, mechanical shock absorption, and the ability to store and release energy for more efficient cyclic motions [9]. Constant stiffness torsional SEAs facilitate compact integration in rotary joints, feature a constant torque-deflection relationship, and are ideal for space-constrained applications [5].

However, SEAs also come with disadvantages: the fixed stiffness must be carefully selected for the target application, as it cannot be varied during runtime. This limits their adaptability compared to VSAs. Additionally, the spring element introduces compliance into the system, which can reduce the actuator bandwidth and lead to oscillatory dynamics that must be managed through control. This increases the complexity of the controller and requirements for sensors. For SEAs, manufacturing tolerances and material fatigue in spring elements can pose challenges, particularly when using low-cost fabrication methods such as Fused deposition Modeling(FDM) 3D printing [10].

Given the objective of developing a modular, low-cost testbed for evaluating torsion springs in SEA configurations, the focus lies on fixed-stiffness, torsional SEA designs. This decision reflects both the targeted use case and the desire for hardware simplicity. Furthermore, using FDM 3D printing to manufacture test springs enables rapid iteration during the design phase, albeit with compromises in stiffness accuracy and fatigue resistance compared to metal-based designs [10]. The selected approach enables quick prototyping and functional evaluation of torsional SEA architectures under realistic operating conditions.

2.2 Spring Design Tool

The design of the elastic element plays a critical role in the behavior of Series Elastic Actuators (SEAs), where performance depends on the specific torque deflection profile, fatigue resistance and compatibility with compact actuator geometries. Several research efforts have focused on designing and optimizing springs for SEA applications, using various

methodologies and computational tools to enhance their efficiency and adaptability. A common objective in these efforts is improving the energy density of springs, allowing for greater energy storage in designs [1, 5, 10].

For this project, a MATLAB-based design tool was selected to generate linear torsional springs. The tool presented in [1] was developed to enable rapid prototyping with customizable geometric and material parameters. This tool generates springs consisting of radially arranged inward-pointing flexures anchored to an outer ring. These flexures interact with a camshaft, forming a fixed-free beam configuration, allowing substantial angular deflection at relatively low stiffness compared to fixed-fixed geometries [1]. This configuration is particularly beneficial for SEAs, where low stiffness and large deflection contribute to increased energy storage and compliant interaction.

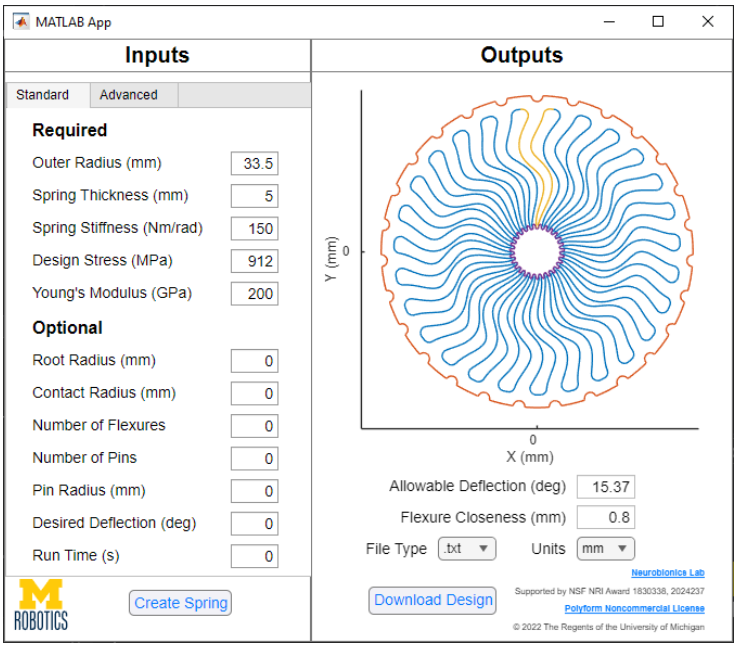

Figure 2.2: Springs design tool interface

2.2.1 Design parameters and Limitations

The output of the design tool includes the full geometry of the spring, along with key design parameters such as stiffness, allowable deflection, and the number of flexures. However, the current implementation assumes the use of a camshaft as an inner and outer interface. The tool allows changing geometric bounds such as the outer radius, spring thickness, root and contact radius, making it possible to specify the length for flexures, but also influencing the inner cam. Furthermore, a user can adjust the design stiffness, material properties with the Young's modulus and the design stress. Additionally, more advanced parameters can be selected to further customize the desired design, which will not be discussed here as it would exceed the scope of this introduction.

While the design tool is capable of producing springs that meet target stiffness values and geometric constraints, it has limitations. Notably, it assumes linear elastic behavior and isotropic material properties. Furthermore, some parameters such as the spring shape and the optimizing strategy are hard-coded and cannot be directly modified by the user. Nevertheless, its rapid generation capability makes it a suitable choice for iterative spring development in prototyping phases.

2.2.2 This works approach

Originally, the design tool was intended to support conventional subtractive manufacturing of metal springs with well-defined linear elastic properties. However, in this work the tool is repurposed to generate designs suitable for 3D printing with materials such as PLA, PETG, and PC-ABS. These materials exhibit different fatigue and elastic behaviors than metals and introduce an additional dependency on the manufacturing method. Thus, the spring configuration of the inner cam, especially concerning the minimum tip radius and gap tolerances, needs to be adjusted for FDM printing.

Furthermore, the outer cam interface is undesirable since this feature may introduce mechanical play. Therefore, in this work, the outer connection will be redesigned to reduce potential inaccuracies. The inner cam connection will not be changed since this would directly interfere with stiffness.

3 Design of the SEA Testbed

The primary aim of the experimental testbed is to evaluate the characteristics of torsion springs of different sizes, providing a modular structure. In addition, its goal is to provide a framework to test control strategies for SEA systems, therefore simulating an elastic robot joint.

3.1 Conceptual Design of the Testbed

The testbed achieves this with the conceptual mechanical design, shown in Figure 3.1. It features an actuator, capable of measuring its own output shaft angle, which is coupled in series with a spring, forming a SEA-System. Torque generated by the actuator is transmitted through the spring onto a drive shaft which is directly coupled with a rotational position sensor to measure the drive shaft angle position and consequently the spring output angle position. Furthermore, to measure the forces experienced by a spring, a force-torque sensor(FTS) can be connected to the drive shaft. A link arm is attached to the drive shaft to accommodate additional loads and expand this system to an elastic robot joint. In addition, two ball bearings incorporated into a tower structure are introduced close to the link to support its weight.

This framework establishes two different testbed configurations. Each configuration is managed by the control system, which is responsible for maintaining communication with all components. The first configuration is designed for static tests, utilizing the setup mentioned above without incorporating any weighting on the link. In contrast, the second configuration is intended for dynamic testing. This is achieved by removing the FTS, therefore permitting the link to move freely, effectively simulating an elastic robot joint which can be loaded via the link.

Static Test Configuration

This configuration aims to evaluate the characteristic stiffness of a spring, capturing the deflection angle and the torque experienced by a spring. It's name comes from the assumption of a rigid drive shaft due to its connection to the force-torque sensor. The link is kept in an upright position to minimize its impact on torque readings beyond those applied by the actuator due to gravity. Under these conditions, the force applied by the actuator causes the spring to deflect from its resting state as the drive shaft cannot rotate. The FTS measures the torque imposed on the system, and the position of the motor output shaft angle equals the spring deflection angle.

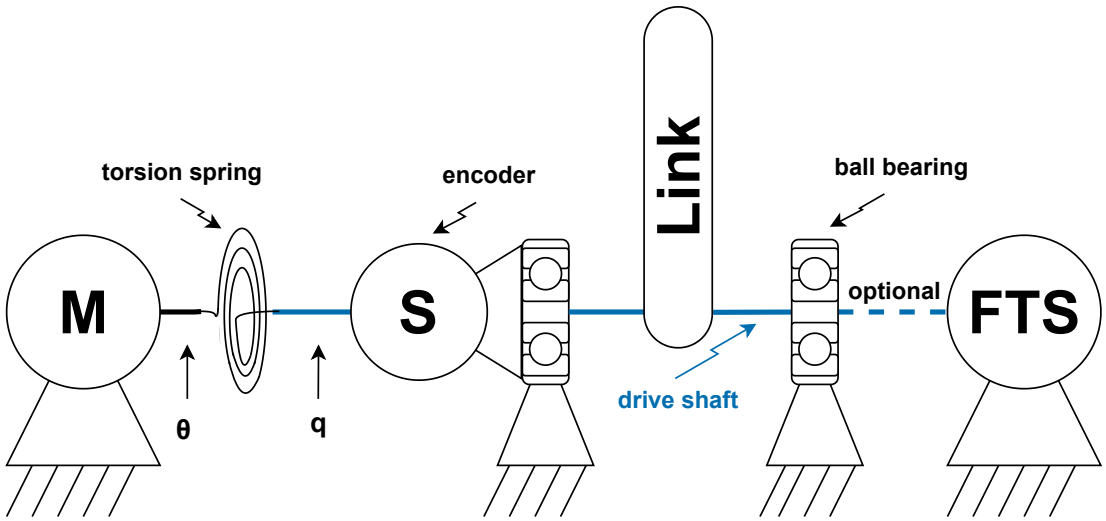

Figure 3.1: Mechanical concept graph

Dynamic Test Configuration

The dynamic configuration serves as a framework for testing dynamic properties of the system, such as damping behavior and for testing control strategies, offering an interface for the actuator and providing angular measurements for evaluation. In this configuration, the FTS is removed to allow the drive shaft to rotate freely, consequently also removing a direct measurement of the torque transmitted through the spring. The encoder is employed to measure the spring's output shaft angle, which, alongside the actuator angle, is used to calculate the spring's deflection angle by taking their difference. Using this angle and the spring stiffness recorded in the static configuration, a model-based estimate of the transmitted torque can be derived. This method proves advantageous when implementing controllers with this system.

CAD and Prototyping

A key design principle of the testbed is modularity, which allows the evaluation of a wide variety of springs. To achieve this, 3D-printed connectors, which have to be adapted to different spring sizes and geometries, as well as off the shelf components are utilized to allow rapid prototyping. In addition, cost efficiency was a primary consideration in component selection. In Figure 3.2 the CAD of the prototype is shown in exploded view. The placement of each component follows the described mechanical concept, with added connectors and a base to host the structure. To simplify the image screws, the drive shaft, the power supply, and the control system are not displayed.

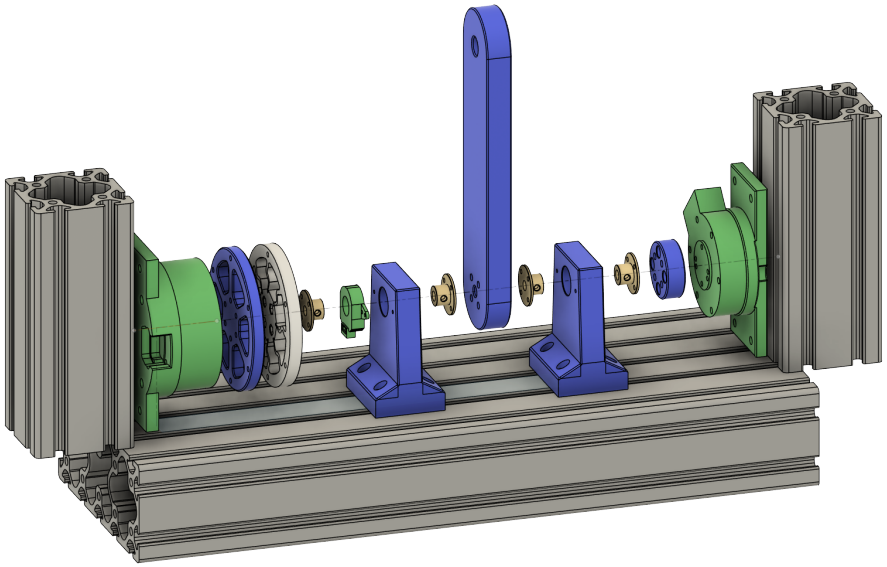

Figure 3.2: Exploded CAD of prototype, without shaft and fasteners in static configuration

3.2 Testbed Components

This section outlines components and the design of connectors for this experimental testbed, focusing on cost effectiveness and modularity and easy spring replacement.

3.2.1 Actuator

The motor **MyActuator RMD-X8-Pro-H 1:6 V3** [11](cf. Fig. 3.3(a)) was chosen primarily for its cost-effectiveness and availability on site. It features a nominal torque of 10 Nm and a maximum of 20 Nm with a gear ratio of $1:6$, providing a powerful choice for torque control tasks. Additionally, this model provides three on board controllers for torque, speed and position of the output shaft. A CAN-Bus interface with a baud-rate of $1\,\mathrm{M}\frac{\mathrm{bit}}{\mathrm{s}}$ provides a sufficient foundation for control frequencies up to 1 kHz. Furthermore this actuator posseses two encoders, 16bit for input- and 14bit for the output angle measurement. Furthermore, position resolution is described to be $0.01°$ and a backlash of $0.167°$. These features enable the integration of the fundamental capabilities necessary for the operation of this testbed and experimental control strategies. Table B.1 provides an overview of key features.

3.2.2 Force Torque Sensor (FTS)

The used **AFT200-D80-EC** [12, 13](cf. Fig. 3.3(b)) is a FTS capable of simultaneously measuring forces and torques in three axes. This sensor was chosen because it is available in the lab and its torque measurement range of 15 Nm, at a resolution of 0.015 Nm, which

aligns well with the nominal torque of the actuator. The sensor is interfaced through EtherCAT enabling high-frequency data pooling. In this project only the torque around the Z-axis is processed. The manufacturer provides an accuracy reading of 1%. Table B.2 provides selected features of this sensor.

3.2.3 Absolute Rotary Encoder

This encoder is responsible for measuring the spring output shaft angle and in combination with the actuator angle provides the spring defection angle. Thus, a cost-effective product with similar resolution of 14 bit is desirable.

The chosen **AMT212F-V** [14](cf. Fig. 3.3(c)) Sensor is a modular shaft size on axis mounted absolute rotary encoder with an RS-485 interface and a resolution of 14 bit($\approx 0.022°$) at an accuracy of 0.2°. This sensor has been chosen for it's cost effectiveness and it's alignment with the output shaft encoder resolution of the motor. This sensor is interfaced using an RS485 bus with a communication baud rate of 115 200 $\frac{\text{bit}}{\text{s}}$. Its slim protocol allows samples rates at magnitudes up to 1 kHz. Table B.3 provides selected features of this sensor.

3.2.4 Drive Shaft and Flange

To comply with the modular design principle and the components chosen, a combination of a nickel-plated steel driveshaft [15](cf. Fig. 3.3(d)) and hardened steel flanges [16](cf. Fig. 3.3(e)) has been chosen for force transmission. This provides a unified connection method for the testbed components.

The drive shaft has a diameter of 8 mm and it's circular profile is fattened on one side to enhance the connection between the flanges and the shaft. In addition, grooves have been drilled into the flat surface to prevent the clamping screws of the flange from slipping(cf. Fig. 3.3(f)). A shaft length of 170 mm was selected to ensure a minimum spacing of about 10 mm between each component along the drive shaft, allowing access to each component for assembly.

The flange is equipped with four `M4` mounting holes positioned equidistantly to attach to adapters 24 mm from the central axis. This flange-shaft connection has an inner diameter of 8 mm and includes two `M4` threaded holes for clamping screws. Table B.4 provides the specifications.

3.2.5 Connectors/Adapters and Base

Since this testbed is designed to evaluate various spring designs, the decision is made to use 3D printing to manufacture connectors and adapters. This approach is cost-effective and offers the flexibility of easy replacements if needed. However, it may introduce some additional play in the transmission of forces. Consequently, these parts are designed with these considerations in mind, incorporating extra precautions to minimize potential issues, but maintaining a streamlined design to avoid unnecessary material usage.

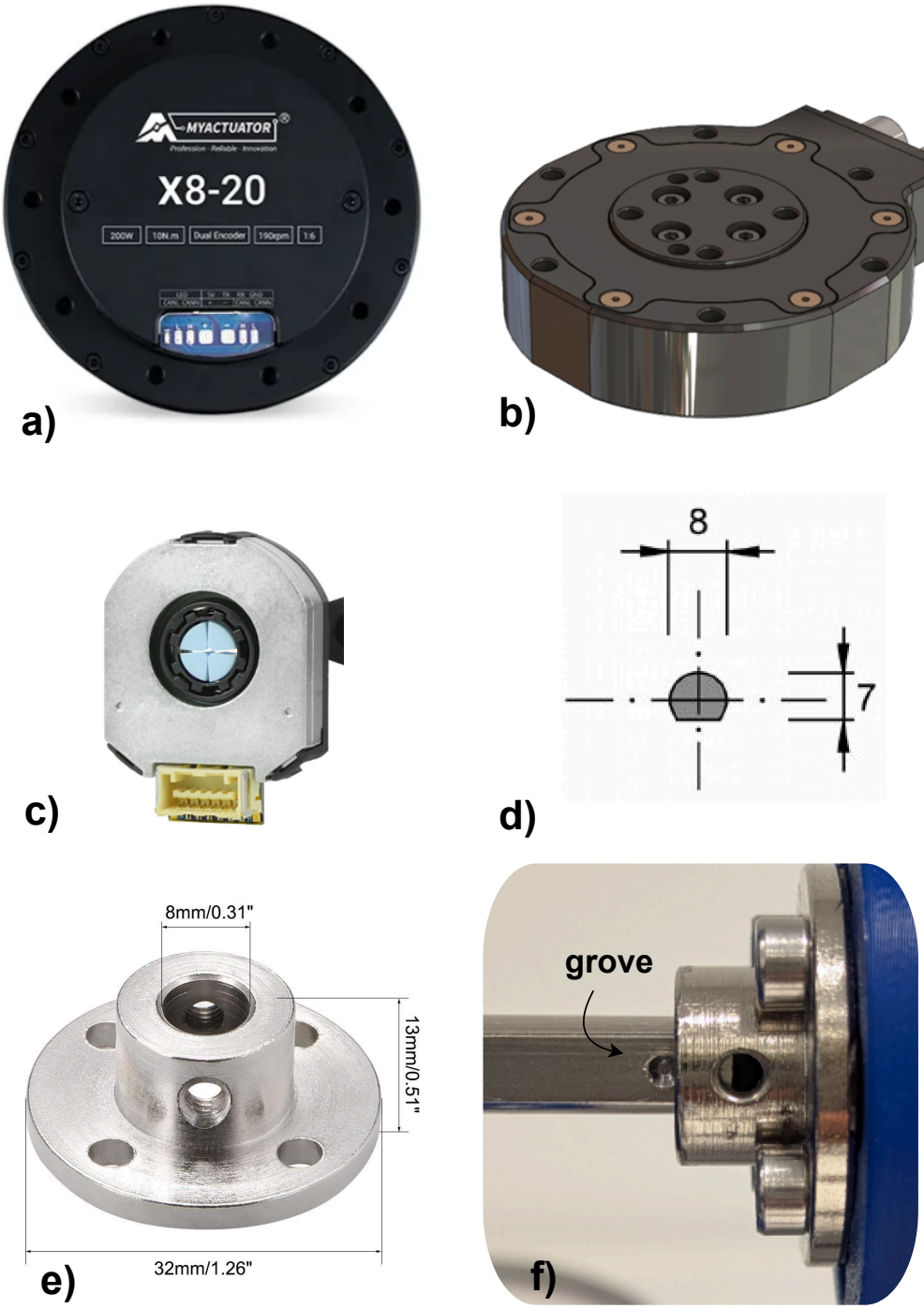

Figure 3.3: Components: a) Actuator, b) FTS, c) Absolute Rotary Encoder, d) Drive Shaft Profile, e) Flange, f) Drive Shaft Grove

Therefore, printing parameters such as the percentage of infill and wall thickness are chosen to support these design objectives. The percentage of filling varies depending on the purpose from 50 to 100% and wall thickness is set from 2 to 3 mm. Figure 3.4 shows the CAD of these components.

Actuator to Spring

A connector between the actuator and a spring(cf. Fig. 3.4(b)) has been designed to simplify the replacement of a spring under examination. The connection employs a direct link on the outer ring of a spring. To maintain only contact with the outer ring, spacers are included in the design. In addition, to improve linkage, threaded inserts [17], of size M4 x 6.35 mm, are incorporated. This method simplifies the spring design, requiring only a hole for the connecting screws of a spring.

Spring to Shaft

To attach a spring to the drive shaft, the previously described flange interface will be used. This interface is designed without threads to further simplify the design and fabrication of the spring. A screw-nut assembly is adopted to directly connect the inner cam of the spring to the flange(cf. Fig. 3.5(a)). Thus, a spring does not require and further manipulations after it is printed and can be tested right away.

For the integration of this approach within the spring design, four nuts are inserted into recesses in the spring body. This method ensures that the connection between the flange and the spring does not disrupt the actuator-spring interface, given the constrained space. This is crucial for proper deflection of a spring. A template for easy integration into the spring design has been created and is shown in Figure 3.5.

Link and Ball Bearings

Given that this testbed integrates a link capable of supporting loads to mimic the functionality of an elastic robotic joint, it becomes essential to include bearings to support the additional weight. This is done to avoid applying unwanted radial or axial load stresses to a spring.

A bearing support is designed to look like a tower to accommodate one or two bearings [18] with a press fit(cf. Fig. 3.4(d)). Two of these towers are placed on both sides of the connecting link to support it's weight. The ball bearing can withstand a static radial load of up to 1370 N and a dynamic radial load of 3330 N. In addition, these components are designed to securely position angle sensors on either side by providing a threadless screw hole for the connection screw. This method results in a predefined fixed height for the drive shaft, which consequently restricts the dimensions of the radial spring. For this experimental configuration, a height of 80 mm was selected.

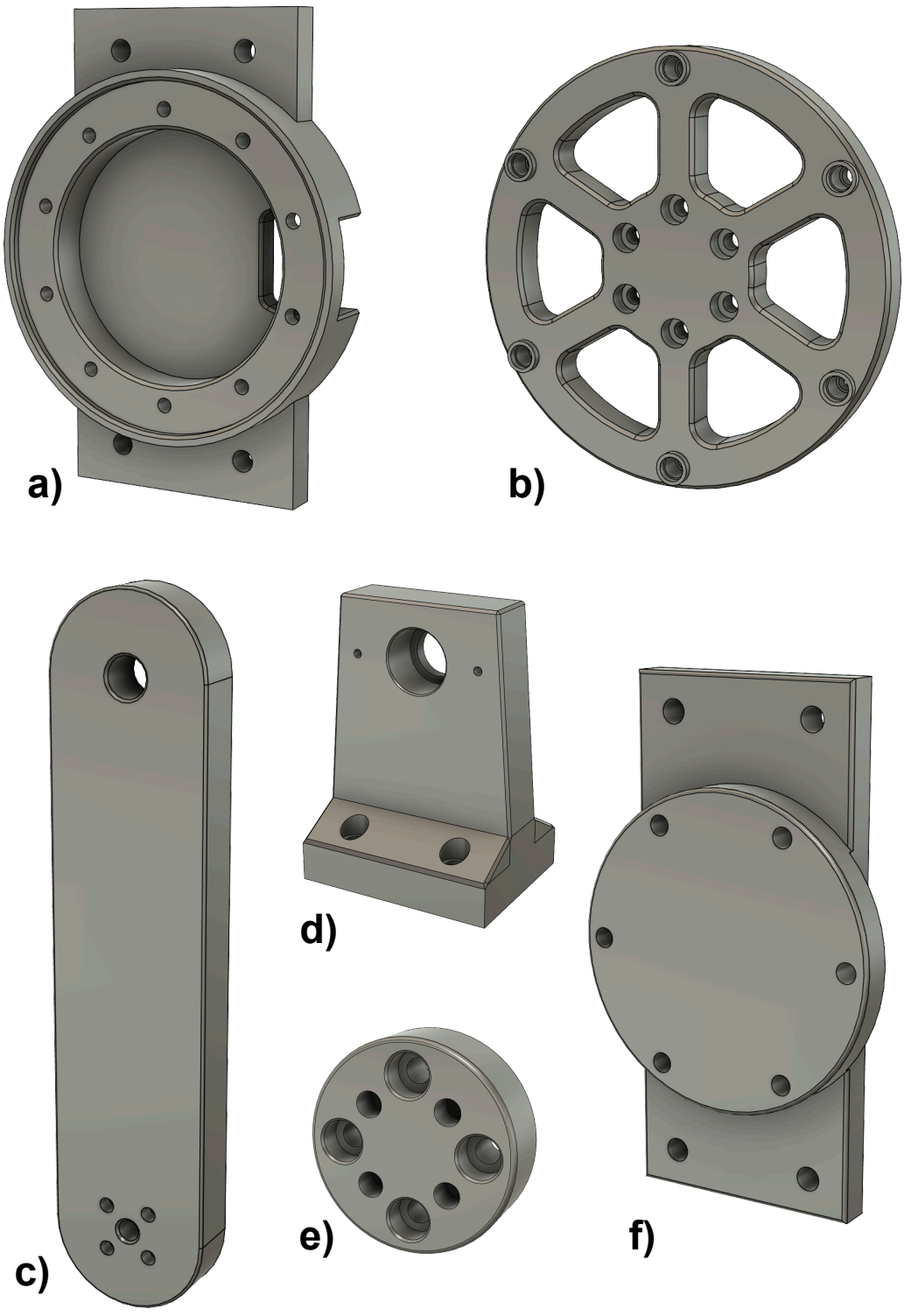

Figure 3.4: Adapters/Connectors: a) Actuator to ITEM Profile, b) Actuator to Spring, c) Link, d) Ball Bearing Tower, e) Flange to FTS, f) FTS to ITEM Profile

The link is designed to establish a connection between a load and the drive shaft at a specific distance, effectively emulating a robotic joint(cf. Fig. 3.4(c)). This particular distance has been specified as 200 mm from the drive shaft axis to the load axis. The choice of this distance serves the purpose of replicating small-scale robotic environments. The linkage to the driveshaft is established through two flanges located on either end, which facilitate a more efficient transfer of forces. To enhance this interface, the design of the link incorporates four holes, each fitted with thread inserts to ensure a robust and rigid connection. At the end of the link, a hole allows for the attachment of weights as a load.

Shaft to FTS

The interface between shaft and FTS is challenging. To accommodate this connection in this limited space due to similar connection interface dimensions, the screws that connect to the FTS are recessed. Moreover, since a flange does not provide a threaded connection, thread inserts are embedded in the connector. This results in a compact design(cf. Fig. 3.4(e)) that is printed with an infill of 100%.

Base

The Base of this testbed is composed of **ITEM Aluminum profiles** [19], chosen due to their modular nature and availability on site. The setup features a pair of vertically oriented profiles, which are connected to a more substantial and horizontally oriented profile. This configuration allows for vertical and horizontal adjustments of the actuator and FTS. Consequently, it can accommodate a wide variety of spring dimensions. For vertical adjustment, specialized base plates for both the actuator and the FTS(cf. Fig. 3.4(a) and (b)) are designed to connect to the ITEM profiles. Horizontal adjustments are provided by the base profile and its rail-like structures. Table 3.1 lists the specifications of the profiles used.

Remark: The dimensions can appear somewhat arbitrary, which is a result of being limited by the specific components readily accessible on-site.

Parameter	**Profile**	**length mm**
Frame/Base	8-200x80	665
Actuator tower	8-80x80	267
FTS tower	8-80x80	307

Table 3.1: Base profiles dimensions

Spring to FTS and Actuator to Shaft

Additionally, implementing a connection from the spring to FTS(cf. Fig. 3.5(c)) and the actuator to the drive shaft(cf. Fig. 3.5(b)) is considered beneficial for testing purposes. This would allow switching the position of FTS and actuator, and therefore also on which side of the spring the link angle is measured. These adapters have been designed with the design considerations mentioned above for each parts purpose, respectively.

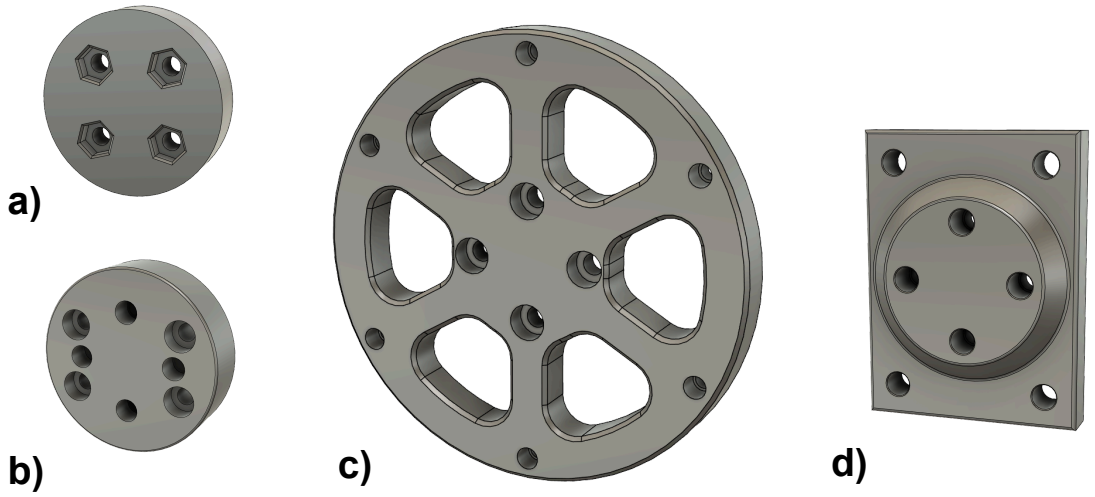

Figure 3.5: Additional adapters/connectors: a) Spring to shaft, b) Actuator to shaft, c) Spring to FTS and d) Base to Spring

Base to Spring

In order to perform dynamic damping tests a connection from the base to a spring is desirable(cf. Fig. 3.5(d)). This component is designed to be combined with the Spring to FTS adapter and provides a base plate for this adapter to connect to the ITEM base instead of the FTS. In this way a spring can be connected to the base and be deflected with the link. The design makes use of inserted threads to essentially mimic the FTS connection interface.

3.2.6 Control System

The Control System is responsible for communication with the actuator and sensors via their respective interface and running the control software. A **Raspberry Pi 5** [20] (RPI5) is used, as it was already available on site and provides an Ethernet interface for EtherCAT. This Micro Computer features a 2.4 GHz quadcore 64-bit Arm Cortex-A76 CPU and a Real-Time clock. This enables it to perform time-sensitive tasks in parallel, which is crucial for the testbed functionality. Since this model does not provide an RS-485 or CAN bus, additional hardware is required to communicate with the encoder and actuator.

Additional Hardware

In order to extend the communication capabilities of the control system to a CAN- and RS485 bus, a hardware shield, also called hardware attached on top (HAT), is used. The **RS485/CAN HAT** [21] converts up to two serial peripheral interfaces (SPIs) to the RS485- and CAN Bus. Furthermore, these buses are isolated and can be operated parallel to each other. The RS-485 interface supports baud rates from 300 to 921600 $\frac{\text{bit}}{\text{s}}$ and the CAN bus up to 1 M$\frac{\text{bit}}{\text{s}}$. Selected key features for the control system are listed in Table B.5 and B.6.

3.3 Spring Design and Manufacturing

The following describes how the Matlab design tool, from [1], is used to generate linear torsion springs for this work. The tool gives the user a variety of options to customize and export the generated spring design. In this project, three different types of materials are used to print springs, PLA, PETG and PC-ABS. This Section features exemplary parameters for a spring manufactured out of PETG. Table B.14 shows the parameters for this spring and in Section B.3 the parameters for each spring discussed in this work are displayed. Each spring is assigned a tag that is used as an identifier consisting of a simple numbering system with a prior material label (`Material IDXX`). Figure 3.6 shows the tool interface.

3.3.1 Design Parameters

This section examines the design parameters of the test springs created for this project. The design of the test springs requires a coordinated evaluation of the parameters rather than assigning arbitrary values to each individually. Moreover, several parameters are constrained or shaped by the specific design decisions of this testbed architecture.

The outer geometric bounds, or “outer radius“ and “Spring thickness“ are set to values similar to what could be used in small-scale robotic applications. In addition, this size was deemed to be manageable in handling and complies with the size restrictions imposed by the height of the testbed axis. The outer radius is set to 50 mm with a thickness of 10 mm.

The inner geometric bound, or “contact radius“, is primarily dictated by the connection to the flange, which features a 16 mm radius mating interface. To ensure enough space for the mechanical linking mechanism, a radius of 20 mm is selected.

The inner and outer bounds set limitations to the available space for flexures and further for the achievable stiffness and deflection. Since the number and length of a flexure contribute to the energy density and capacity of a spring, in addition to material properties.

The material parameters used in the spring design are generally based on datasheet values provided by filament manufacturers. In the present example, the values for PETG from [22] are used. To account for material inconsistencies and fatigue effects, a safety

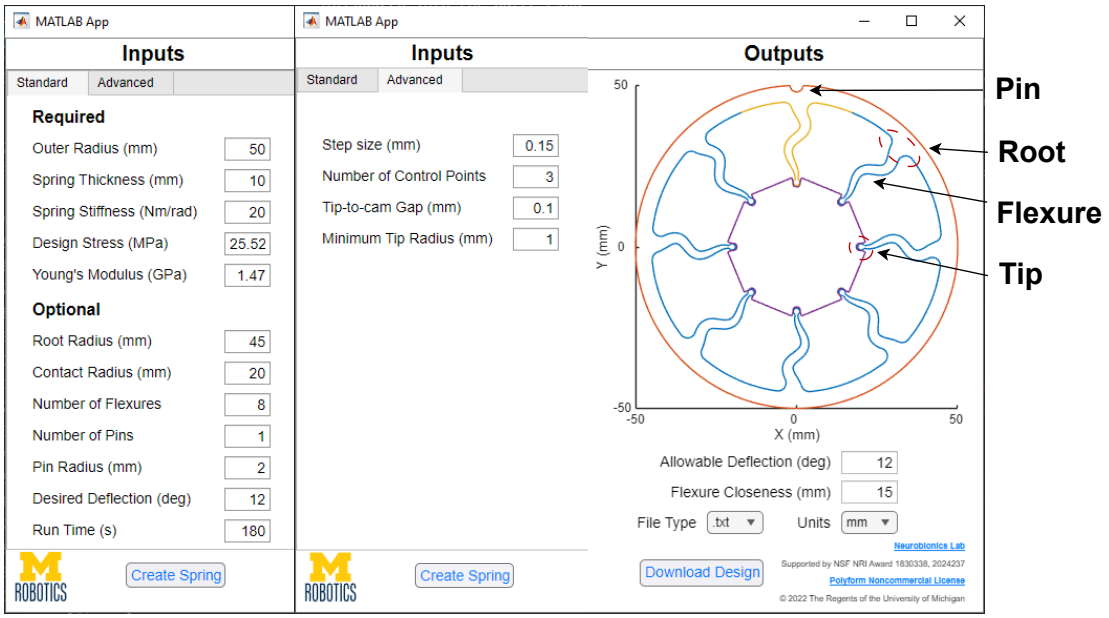

Figure 3.6: Graphical user interface of the spring design tool[1]

factor of 1.25 is applied to the specified tensile strength, for the "Design Stress" parameter. The Young's modulus is taken directly from the manufacturer's data without modification. Since 3D printing introduces anisotropic material properties, the printing parameters must be chosen carefully to ensure that the printed part complies with the applied stress assumptions.

The target stiffness, or "spring stiffness", was deliberately set to a relatively low value of 20 $\frac{\text{Nm}}{\text{rad}}$. This ensures that the spring remains easily deflectable by hand, simplifying handling and initial validation. Although this value is low compared to the intended stiffness test range of the testbed, it provides a safe margin to account for deviations caused by modeling assumptions or manufacturing tolerances.

The target stiffness, or "spring stiffness", was deliberately chosen to a lower value to ensure that potential deviations due to manufacturing tolerances or modeling assumptions could be accommodated. So, even if the fabricated spring exhibits stiffness higher or lower than anticipated, the testbed remains capable of significantly deflecting the spring. The target stiffness is set to 20 $\frac{\text{Nm}}{\text{rad}}$.

Spring deflection is a key parameter that, along with the stiffness values, defines the maximum potential energy that a spring could save. For this testbed with its chosen geometric limits and target stiffness, achievable deflections are about $\approx 12°$. This value was derived from repeated spring generation and is set as desirable for other springs designed with this tool for testing purposes.

The number of pins and pin radius is not critical for this project, as it regards the outer connection interface, which does not influence the stiffness or deflection angle of a

spring. The design tool assumes a camshaft connection for the inner and outer linkage. However, in this project, a different connection mechanism is employed for the outer ring to minimize unwanted mechanical effects. This value is set to 1

The tip-to-cam Gap is wanted to be as low as possible, but to still maintain a fixed-free beam relation between the inner and outer linkage. The achievable manufacturing accuracy has an immense impact on these values. This value is set to 0.100 mm for the example spring. **The minimum Tip Radius** also has an impact on the stiffness curve, due to the limited manufacturing accuracy, the contact surface with the inner cam shaft will be rough and increase friction. For this project larger tip sizes are desired since the material and manufacturing process used make small parts more vulnerable to fatigue and failure. This value is set to 1.000 mm for the example spring.

The run time affects the tool's ability to find a suitable candidate for the desired properties. This tool uses generative methods and tries to find a design that matches the parameters requested. The **step size** for this project is set to 0.15 mm, which represents what could be expected as the accuracy of the 3D printers used in this project. A lower value means that more calculations are needed. This will increase the time required to find a solution, as does a higher value for the **number of control points**, a compromise of 180 seconds is set to still keep rapid prototyping viable.

Parameter	Unit	Value			
Profile-ID	-	09	Number of Pins	-	1.00
Material	-	PETG	Pin Radius	mm	2.00
Outer Radius	mm	50.00	Desired Deflection	deg	12.00
Spring Thickness	mm	10.00	Run Time	s	180.00
Spring Stiffness	Nm/rad	20.00	Step size	mm	0.150
Design Stress	MPa	25.52	Number of Control Points	-	3.000
Young's Modulus	GPa	1.47	Tip-to-cam Gap	mm	0.100
Root Radius	mm	45.00	Minimum Tip Radius	mm	1.000
Contact Radius	mm	20.00	Allowable Deflection	deg	12.00
Number of Flexures	-	8.00	Flexure Closeness	mm	15.13

Table 3.2: PETG ID9 Tool parameters

The output of the design tool consists of a detailed parameter set and a corresponding geometric model of the spring. Specifically, for each spring design, the tool provides a complete 2D profile as a set of splines. This profile can be exported as a DXF or SVG file and directly used for manufacturing workflows such as laser cutting or 3D printing. Each spring geometry is characterized by key output parameters: the number of flexures, their arc length, root thickness, fillet radius, beam width, and tip offset. Additionally, the tool calculates and reports mechanical characteristics including the torsional stiffness (in

Nm/rad), maximum allowable deflection before yield, and an estimate of the linearity of the torque-angle relationship.

3.3.2 Changes to Spring Design

As previously stated the outer cam shaft, provided by the design tool, introduces additional mechanical play and is therefore replaced to fit the connector "`Actuator to spring`". To improve force transmission, the outer radius is expanded by 5 mm, and six evenly distributed holes are added to the outer ring to ensure uniform load transfer through the spring. Additionally, the identification string will be marked on the outer ring for easy identification, by recessing it into the string design. The inner interface is largely unchanged; only the designed "Spring to drive shaft" connector is incorporated into the existing cam. Figure 3.7 the finalized spring design in the CAD software used.

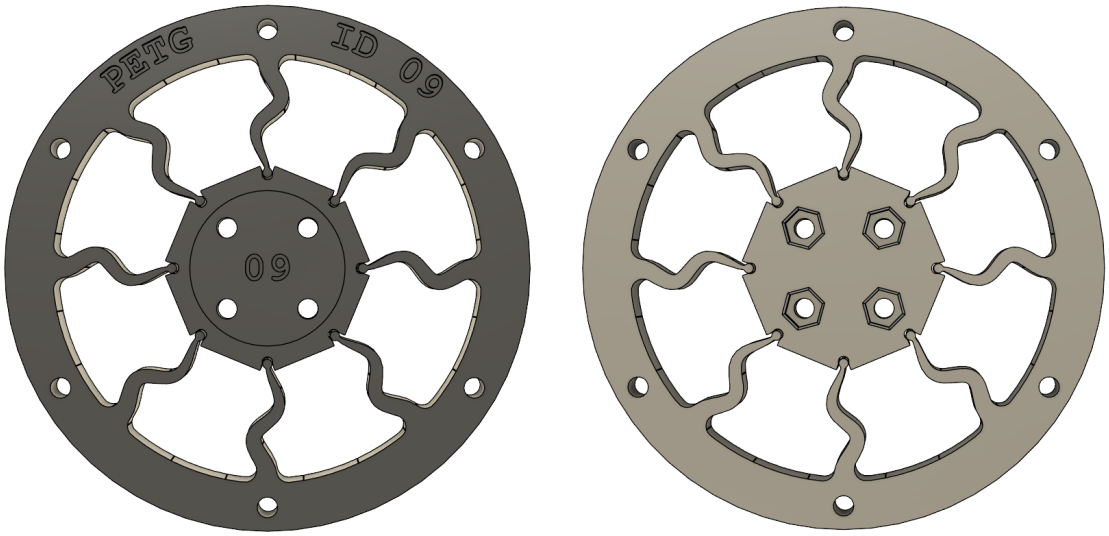

Figure 3.7: CAD of PETG ID09

3.3.3 Manufacturing

For optimal performance of 3D-printed springs, several print settings are selected. The infill is set to 100% to ensure maximum material density and strength. A layer height of 0.2 mm provides a good balance between printing precision and build time, ensuring consistent layer adhesion without sacrificing detail. It is essential to align the printed layers perpendicular to the primary stress direction, this means orienting the print so that the Z-axis corresponds to the axis of the spring. Additionally, using a concentric infill pattern helps align the internal material structure with the radial stress paths, thereby improving overall mechanical integrity.

In this work, springs have been successfully fabricated from PLA, PETG, and PC-ABS using the `Raise3D Pro3 Plus 3D` printer [23].

4 Software Design

This chapter details the development of the software framework used for the experimental evaluation of torsional springs and control algorithms on the actuator testbed. The implementation runs on a Raspberry Pi 5 (RPI5) under a Unix-based operating system (OS) and interfaces the actuator and sensors.

To enable precise data acquisition and control under real-time constraints, the software is structured around static and dynamic test modes. These programs implement deterministic timing behavior, core affinity, and synchronized multi-threading to ensure consistent performance. In both modes, the system acquires multiple different measurements and exports these to a text file.

4.1 Static Test

The static testing protocol is structured to collect real-time measurement data to assess the stiffness of a spring in static testbed configuration. The stiffness k of a torsional spring is defined as the ratio of the applied torque T to the resulting angular displacement θ.

$$k = \frac{T}{\theta}$$

To determine this ratio, the static test program deflects the spring from one end point to another in multiple cycles, reaching a predefined maximum deflection in both directions while capturing torque and angular displacement. Performing the deflection in this manner is necessary to capture a possible difference in the torque angle relationship during loading and unloading. This forms a loop in the torque displacement graph due to internal friction, material damping, and energy dissipation, this phenomenon is known as hysteresis. By determining the slope of this path, the local stiffness can be derived and the overall behavior of a spring throughout its deflection can be analyzed, providing insight into whether this behavior is linear.

However, to enable adaptation to different spring designs, this test works with user-defined parameters, such as the number of loops to perform, the maximum deflection angle, and the position update rate. In addition, an optional live visualization of measurements and export of this data is provided, making it possible to analyze the gathered data after a test is completed. This analysis is not part of this program.

4.1.1 System Overview

The static test system consists of a control system running the software that interfaces with the actuator via a CAN bus and the force-torque sensor (FTS) over EtherCAT. For optional live visualization, a UDP data stream is provided, which can be viewed using additional hardware running appropriate software; in this work, a Simulink model is used for live plotting. Figure 4.1 illustrates the connections for the static test software with the testbed in its static configuration.

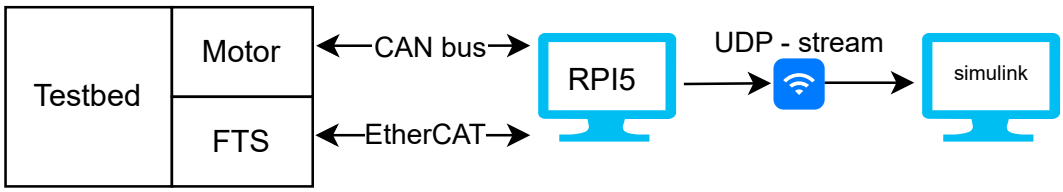

Figure 4.1: Static system overview

4.1.2 Software Design and Implementation

Since the host platform runs a UNIX OS, additional action must be taken to maintain a stable timing for communication and data acquisition, due to possible interference from OS scheduling. This is necessary to keep the temporal integrity of data in tact.

In order to help with real-time execution, the software configures the scheduling policy of the program using the POSIX function `sched_setscheduler()`. POSIX (Portable Operating System Interface) is a standardized set of APIs and system interfaces to ensure compatibility between Unix-like operating systems, providing support for process control, threading, memory management, and real-time scheduling. This function allows the process to run under the First-In, First-Out (FIFO) real-time scheduling policy, which ensures deterministic execution without time-sharing interference from lower-priority processes. In addition to configuring the real-time scheduling policy, the software further assigns the process execution to a specific CPU core using `sched_setaffinity()`. This ensures that the process runs exclusively on a designated core, reducing context-switching, and further strengthening a deterministic and isolated execution environment.

The RPI5's real-time clock is utilized to establish a nanosecond accurate high-precision timer, enabling cyclic measurements at precise intervals. This timer serves as a stable clock for the control algorithm, with its interrupts used as the system clock reference. Additionally, to make the communication more efficient and execution concurrently POSIX-Threads are utilized, to execute tasks in parallel. These threads will share memory, and therefore the `mutex synchronization` method is employed to avoid incomplete data readings and race conditions. This would lead to random crashes or incorrect results when two threads want to access the same memory at the same time. Figure 4.2 shows the general structure of the program in a flow chart.

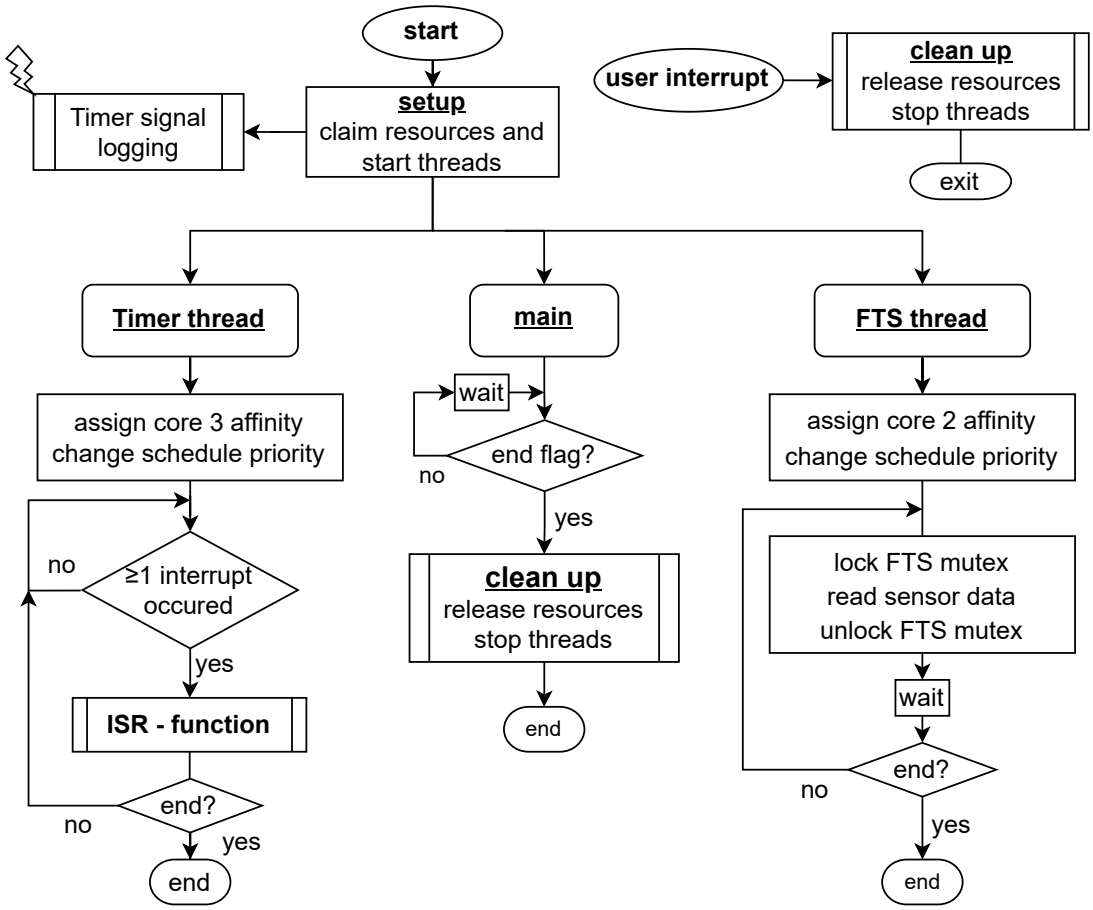

Figure 4.2: Static program structure

Setup

The software setup initializes all required interfaces and binds them to sockets for efficient management. It creates all the necessary threads and activates the timer. In addition, it logs debug messages for the setup progress to the terminal, providing the user with real-time status updates, and guides the user through a calibration process to ensure consistent zero-angle reference points. Before executing the actual test, the program asks the user for confirmation. Together with the setup debug messages, this precaution is intended to prevent potential damage to the test bench or test subject. Since the actuator has the capability to generate significant power to harm the user and the testbed.

FTS-Thread

This thread manages EtherCAT communication with the FTS, continuously acquiring torque measurements and storing them in shared memory. It operates at a higher frequency

than the timer thread, ensuring that the control algorithm either accesses the most recent torque data immediately or waits until the thread finished updating the value. This introduces less accurate timing but is negligible due to the higher update rate than the timer thread uses. The access of shared memory is handled by a `mutex`. This approach guarantees that the control loop always utilizes the most recent measurement while enhancing the system response and accuracy by running this task as a thread on a different core than the control loop. The update rate is set to 500μs.

Timer-Thread

This thread handles hardware timer interrupts by calling an interrupt service routine (ISR). In this way a system clock is provided for control tasks, which will also be referenced as a control clock in this work. This is done by reading how many interrupts have occurred since the last check. If at least one occurred, it calls the ISR. If none occurred, it waits until an interrupt is registered. This method ensures that an interrupt signal does not interfere with the current execution of the ISR, since it will only be able to check the counter again when the ISR-function is finished with its execution. Due to this implementation, the control clock will only be stable if the ISR does not take longer than the set timer interval. The Timer-Thread is executed with high schedule priority and it's own dedicated CPU core to minimize disruptions by the OS. This creates a real-time application with high confidence, but no guarantee. The timer period is set to 2ms, but is generally subject to user parameters.

ISR-Function

This function handles the control algorithm, which reads the measurements θ by polling the actuator and `T` from shared data. It captures timestamps, calculates a new position and commands it to the actuator as well as writes the collected data into the output file, as well as the UDP stream. This update is structured so that if a reading timeout occurs and the data is not updated, the data from the last cycle is still available. Figure 4.3 shows a flow diagram of this function.

4.1.3 Test Procedure and Data Handling

Before executing the static program and testing a spring, the user must edit the test parameters, shown in Table 4.2. Then compile, build, and execute the test with admin permissions due to the use of the POSIX API. After a successful setup, the user is asked if they want to proceed with the test. If the user confirms, the test starts and the live data can be viewed via the UDB data stream, any occurring errors will be printed to the terminal until the test finishes. When the test is finished gathered data is available as a text file stored in a dedicated output folder.

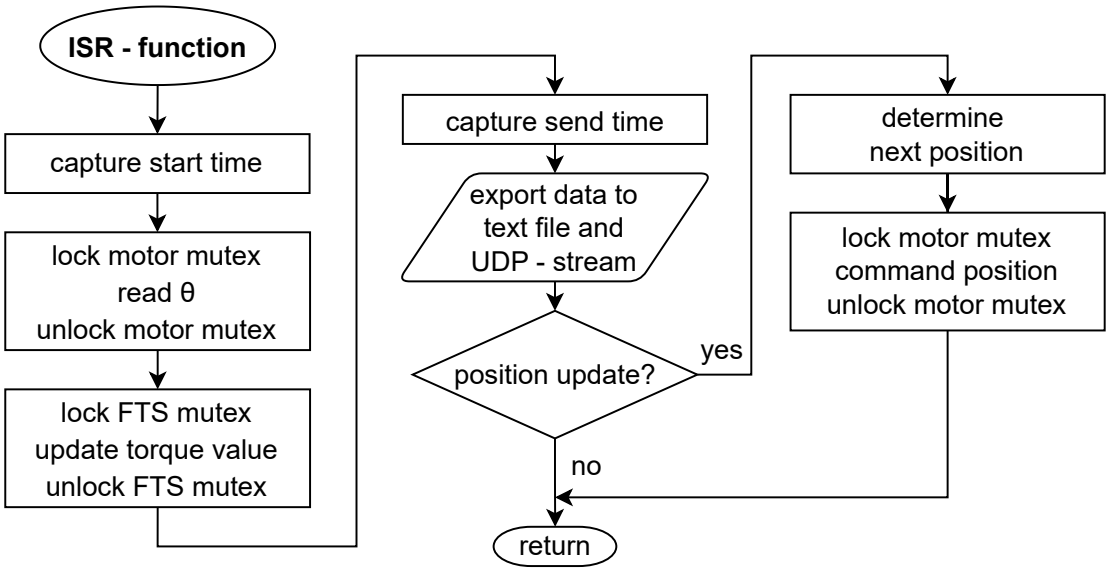

Figure 4.3: Static ISR function

Parameter	**Description**
Timer interrupt period	Representing the clock period
Number of cyclic loops	Total count of loops in the cycle
Angle step size $\Delta\theta$	Incremental rise in position per update
Position update rate	Time to wait before updating position
Maximal deflection angle $\lvert\theta_{\max}\rvert$	Absolute value of maximal deflection angle
Start deflection direction	Initial movement towards $-\theta_{\max}$ or $+\theta_{\max}$
Torque limit	Protection against excessive torque
UDP IP and port	Used for live data transmission

Table 4.1: User settings static test descriptions

Procedure/Algorithm

The position control algorithm uses the absolute position command `(A4)` of the actuator to move the output shaft to new positions. This method is chosen because it simplifies the control task by utilizing the actuator's onboard position controller. The position update is calculated within the ISR by adding the user-defined step size $\Delta\theta$ to the current position repeatedly. The algorithm starts with deflecting the spring to an initial position at $\theta_{\max}$ regarding the initial deflection direction. From this position on it performs the cyclic loops by repeatedly adding the step size to the current position until it reaches an end point defined as $\theta_{\max}$ and then changes direction. When the subsequent end point is reached,

one loop is completed. This cycle continues until all iterations are complete, after which the spring is returned to the start position established during setup calibration.

Furthermore, additional safety measures are taken to protect the sensor and springs, the measured torque will be compared to a set limit by the test parameters. If it exceeds this value, the test is terminated, releasing all resources and shutting of the actuator, essential releasing potential tension on the spring.

However, the recorded data is exported at every ISR call prior to each position update. The output produces a human-readable yet machine-processable table in a `.txt` file. At the start of each loop, a line is inserted that contains the `loop x` (with x as the loop index). When an endpoint is reached and the deflection direction changes, the line will be marked with `turnaround` to aid structured parsing later on. Each entry includes torque and angle measurements, along with two timestamps in microseconds, one marking the ISR start and another indicating the moment the data is written. The complete data set is saved in a `.txt` file using comma separated values (CSV), ensuring compatibility with standard post-processing tools.

4.2 Dynamic Test

The purpose of the dynamic test framework is to provide a reusable template for the evaluation of various control strategies or other dynamic tests for SEAs. This will allow a users to implement their own control algorithms while ensuring that all necessary measurements and derived data, such as velocity and acceleration, are readily available.

The dynamic test uses the dynamic testbed configuration. The removal of the FTS allows for free movement of the drive shaft and subsequently introduces the need for calibration for both angles measurements. This has been integrated into the guided setup procedure. Figure 4.4 illustrates the modified system for dynamic testing. Since the software architectures for dynamic and static tests share a similar structure, the following will focus on changes to the static software to facilitate the dynamic test software template.

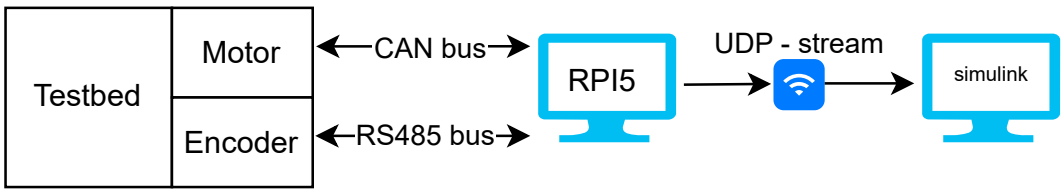

Figure 4.4: Dynamic system overview

4.2.1 Sensor Changes

Since the drive shaft now needs to move and torque measurements are no longer needed, the FTS is physically removed from the system and the software. Instead, an absolute encoder is introduced to provide the measurement of the link angle. The absolute encoder

is handled by a thread with high priority and performs measurements at higher rates than the control loop operates, similar to the FTS-thread. This introduces less accurate timing but is negligible due to the higher update rate than the timer thread uses. The sensor features a RS485 interface and makes a use of the HAT. The update rate is set to 900μs.

4.2.2 Test Changes

Since this testbed configuration allows the drive shaft to move freely, it is now necessary to calibrate the measured angles to set a zero angle for the link and actuator. This procedure has been incorporated into the setup part of the program, by interacting with the user via the terminal to confirm the calibration positions. In addition to providing safety features for the components and the user, a workspace limitation is introduced that limits the actuator output shaft angle to a user-defined range. If the defined relative angle space is exceeded, due to a current state or a pending command, the test is terminated. The actuator is deactivated and the output shaft is disengaged allowing free rotation. However, position updates are now handled via a function available to the user. These functions include output speed regulation. For testing purposes, an additional thread safe function has been implemented to facilitate user communication with the control loop. This function enables a user to set a new position via the terminal.

Furthermore, the control of SEAs may require link and actuator velocity as well as acceleration signals, which have been added to this template. These values are derived from the measurements and processed using a moving average filter over a user-defined window size to reduce sensor noise. Furthermore, the data export structure has been changed to log additional data. Since marking completed loops and direction changes is no longer necessary, the exported text data is stored as a single continuous CSV table. The UDP data stream has been extended to include additional signals and measurements.

The ISR function for dynamic testing is illustrated in the flow chart in Figure 4.5. The `Control Box` represents a placeholder for custom control algorithms provided by the user.

The test procedure is largely unchanged, although calibration is now more extensive and parameter are now moved to a dedicated file to improve readability and making them available to multiple programs. New parameters include:

Parameter	**Description**
workspace limit	symmetric limit around the calibration point
motor speed limit	maximum allowed actuator speed for control tasks (dps)
θ_{start}	initial calibration position

Table 4.2: Additional user parameters for dynamic tests

Since an elastic element and the actuators backlash can cause difficulties finding a suitable calibration point for the actuator, due to possible mechanical play, the setup routine references a hard-coded zero point θ_{start}. The user is asked to compare this value with

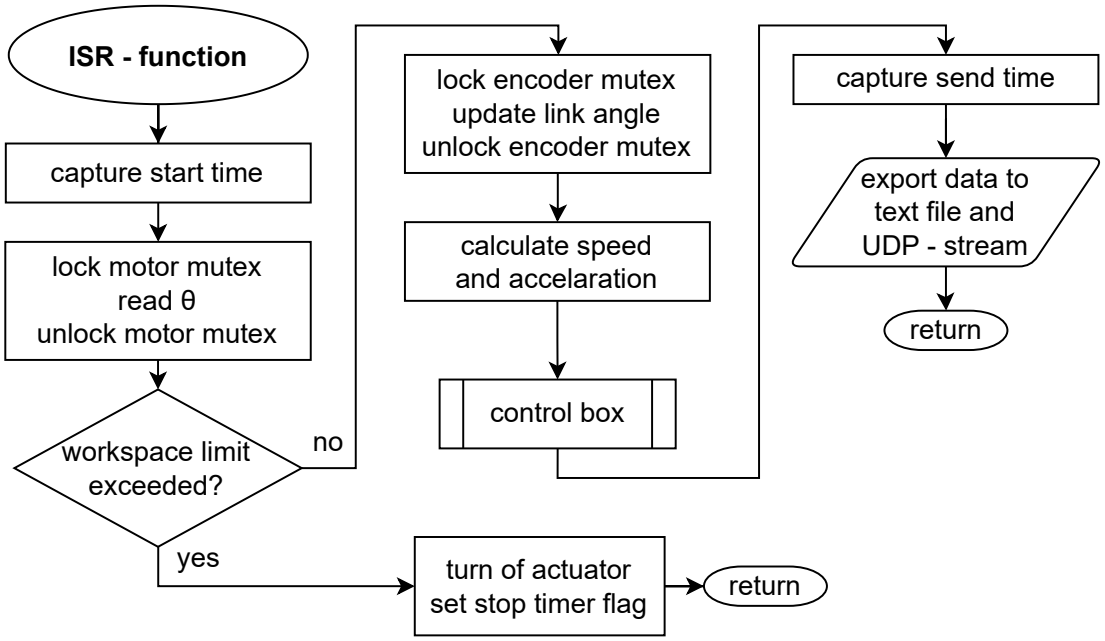

Figure 4.5: Dynamic ISR function

a current measurement to identify a possible miss alignment. If the provided zero point is confirmed by the user, then the actuator will move to this point to provide a rigid interface for link angle calibration. If not, the user aborts and changes this value in the parameters. It is important to calibrate these angles to reference a common zero point, since spring deflection will be calculated using both angles.

The software developed for this testbed is provided in [24].

5 Validation and Discussion

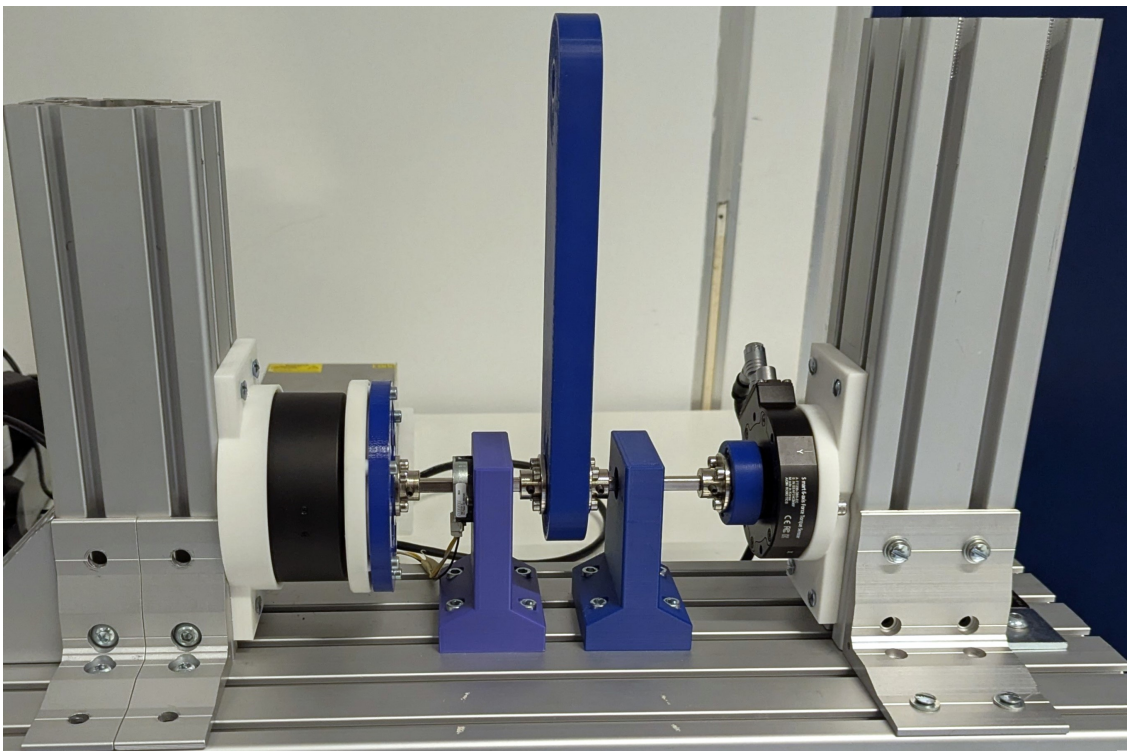

Figure 5.1: SEA Testbed in static configuration

5.1 Testbed Components

The testbed comprises several components which are validated for their suitability in the intended testbed setup. This section discusses core hardware elements such as actuators, sensors, and structural components and outlines the test procedures used to assess their performance and suitability for this project.

5.1.1 Actuator

To validate the **MyActuator-X8-20** used in this testbed, both its communication speed and dynamic response are analyzed. This evaluation is essential to determine the feasible control frequency and to assess whether the actuator can adequately respond to rapidly

changing position commands. This is an important requirement for dynamic tests, such as control tasks. For this purpose, additional software was developed.

Communication Speed

A high control frequency and therefore a high communication speed is desired because it influences the system reaction to high frequency oscillations of the elastic element. The actuators specifications suggest that communication can be performed with a CAN baud rate of 1 MHz, which would provide a good basis for a 1 kHz control frequency. A CAN-frame for these motors commands consists of about $\approx$ 120 bit therefore the one-way-time for one package on the bus would be $\approx$ 120 μs. However, the usage of a HAT to access the bus and the host module's OS introduce additional process delays. In addition, the process time of the actuator is unknown. Therefore, a measurement is taken to provide a better insight into the system behavior.

The timing test evaluates how long it takes to send a command and process the received response. This duration is measured by capturing a time stamp before and after a command is executed with high schedule priority and core affinity, as used in the static test program. This test is performed for all commands used for this testbed, including reading the angle position and commanding new positions.

Furthermore, a base-line reading was performed using an additional Raspberry Pi and an identical HAT for the CAN communication and the same message structure and protocol for messages. Both Raspberry Pi's are connected via a HAT to the same bus line and the test evaluation was performed 25 times in a row. The average duration for the two-way timing was 640.240 μs. This measurement shows that significant overhead is added to the transmission time. This baseline reading will prove to be useful in identifying possible sources of delays.

However, testing the timing with the actual actuator will be performed for various command used in this work. Furthermore, to get a better insight about timing variations of this communication, the software performs a more intensive test. This test evaluates an overall mean value and a corresponding standard deviation over 30 batches of 1000 command executions each. A more detailed report including the longest execution time and the number of executions that take longer than a set threshold time per batch is provided in Section B.2.1. Table 5.1 shows a summary of these tests. These tested commands include, reading the output angle `(0x92)`, commanding an absolute position `(0xA4)`, an alternative position tracking command `(0xA5)` and the subsequent exectution of `(0x92)` and `(0xA4)`, similar to how these commands are used in static and dynamic tests.

Command	Mean of Batch Means (μs)	Std. Dev. of Batch Means (μs)
read angle (`0x92`)	999.343	1.853
absolute position (`0xA4`)	1401.135	1.772
position tracking (`0xA5`)	1404.801	1.636
(`0x92`) + (`0xA4`)	2788.283	2.300

Table 5.1: Statistics actuator command execution time

The results show that a 1 kHz control frequency is not feasible with this system. Since it is necessary to command and read an angle within one clock period and the combination takes at least $\approx$ 2.8 ms to execute. Comparing these timings to the base-line result suggests that the internal processes of this actuator greatly delay the communication. In addition, when viewing the longest duration, either the OS or the actuator randomly causes a considerable delay, sometimes more than double the mean execution time. This introduces limitations for the control frequency, considering the additional processing time added by the software and its control algorithm. A stable control frequency could be reached with 250 Hz, i.e. a clock period 4 ms.

Dynamic Position Response

This experiment aims to assess position tracking to determine the suitability of the actuator for control tasks. This test commands a sinusoidal trajectory under varying control parameters, including the position update rate and sinusoidal frequency. The software used the dynamic test template with a sampling time of 4 ms, determined in the previous subsection. For the execution of this test, the actuator was disconnected from the testbed.

The commanded position is defined as

$$\theta_{\text{des}}(t) = A \cdot \sin(2\pi f t), \tag{5.1}$$

where $\theta_{\text{des}}(t)$ represents the desired angular position in degrees, `A` the amplitude, `f` is the sinusoidal excitation frequency and `t` the time. To get a smooth desired path, the discrete time points were generated by calculating the difference between time stamps. This setup enabled the evaluation of the dynamic performance of the actuator, including its tracking accuracy, delay, and general responsiveness to control inputs. The tested commands are described as the absolute position command `A4` and position tracking command `A5` by the actuators protocol. The difference of these command lies in their intended use. The actuator protocol describes command `A5` to be used for cases where the trajectory planning is realized by users in the control. Where command `A4` lacks a description of its intended use, but the context suggests that the onboard system performs trajectory planning or tracking.

By varying the position update rate and excitation frequency, the impact of different control timings on the actuator's ability to maintain precise positioning was examined.

The results of these tests provide insight into the actuator's feasibility for tasks requiring real-time control and dynamic motion tracking.

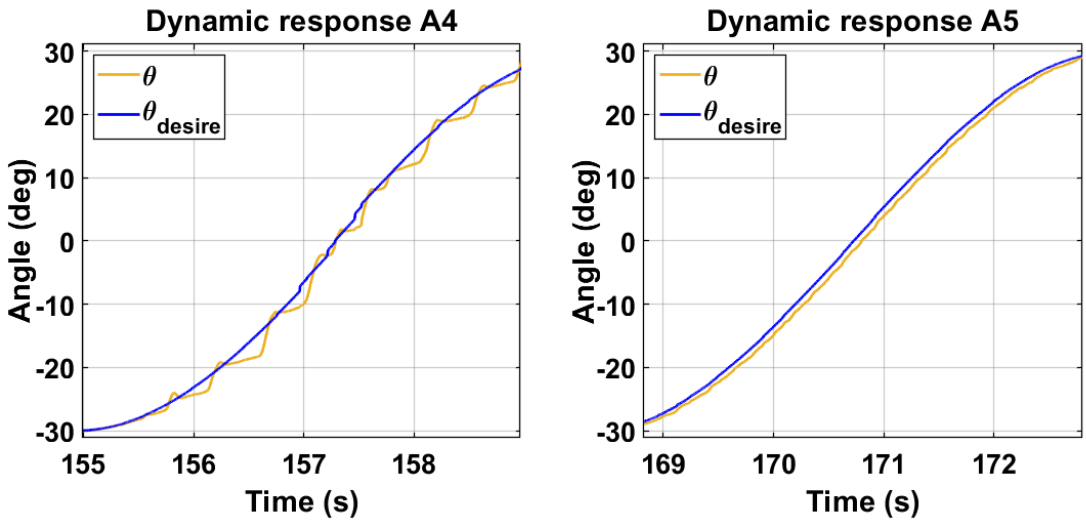

Figure 5.2: Dynamic response test: A = 30, f = 0.1 Hz, position update rate = 4 ms, command A4 (left), command A5 (right)

Figures 5.2 and 5.3 show the actuator response captured with a 4 ms sampling time for two different position update rates for two commands. Command `A4`, on the left side, shows notable tracking errors for high position update rates, manifesting as jerk movements, which becomes more accurate for lower update rates. Command `A5`, on the right side, shows a significantly better accuracy but introduces a significant constant phase delay for both update rates. This test was extended to additional control configurations, the results of which are summarized in Table 5.2, where the data sample time was set to 4 ms. This table provides a column for the expected maximum speed the output shaft will experience for the specific configuration, providing additional information to analyze the actuators behavior.

These tests show that with higher trajectory frequencies `f` the command `A4` fails to follow the path. The delay of Command `A5` remains the same for various configurations until the actuator eventually reaches its physical speed and acceleration limits. This delay is probably introduced by the internal actuator control. However, a lower update rate for command `A4` is not favorable for control tasks, and a delay of ≈ 75 ms for command `A5` imposes challenges when handling high-frequency oscillations of a spring. This leads to the conclusion that the actuator's dynamic behavior is not suitable for dynamic tests.

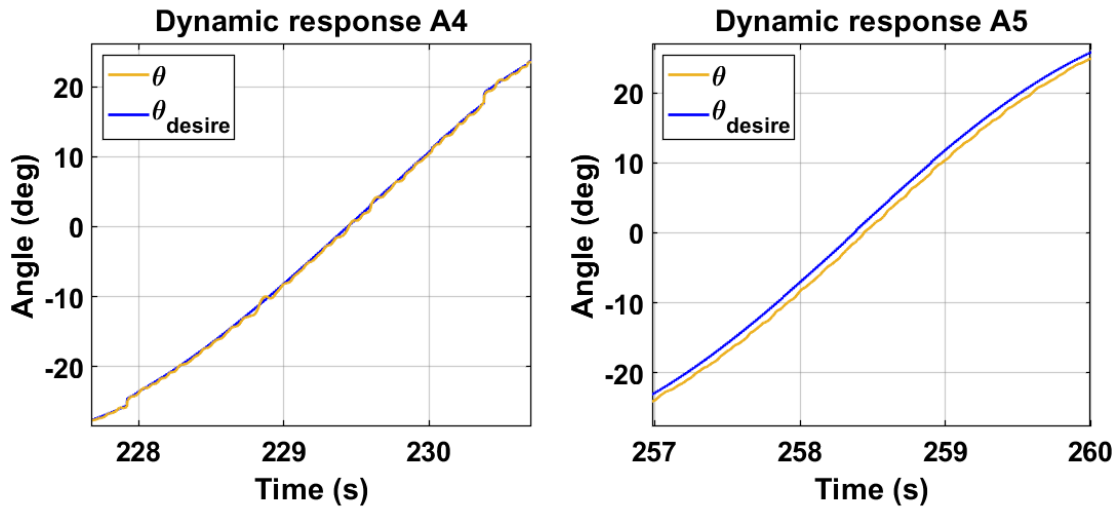

Figure 5.3: Dynamic response test: A = 30, f = 0.1 Hz, position update rate = 16 ms, command A4 (left), command A5 (right)

Command	f(Hz)	slope(dps)	update(ms)	tracking results
A5	0.100	18.85	4	75 ms delay
A4	0.100	18.85	4	regular jumps
A5	0.100	18.85	16	75 ms delay
A4	0.100	18.85	16	regular and less jumps
A5	1.000	188.50	4	75 ms delay
A4	1.000	188.50	4	over shoot and bad tracking
A5	1.000	188.50	16	75 ms delay
A4	1.000	188.50	16	no tracking, motor jerks
A5	2.000	376.99	4	exceeded speed limit of 300 dps, smaller amplitude
A5	3.000	565.49	4	cannot catch up, no tracking

Table 5.2: Actuator dynamic response

5.1.2 FTS

The FTS used in this testbed communicates through an EtherCAT interface and the manufacturer states that this sensor can sample force and torque values at a rate of 1 kHz. This performance is evaluated for this system to see if the polling rate as described in Section 4 can be reached and, therefore, if the chosen method is valid.

Communication Speed

The test procedure is similar to that used for the actuator communication speed, capturing timestamps before and after command execution, but for this setup there is no baseline test. The FTS is detached from the testbed, and the RPI5 is connected directly via an Ethernet cable. Since the EtherCAT interface operates with real-time capabilities, its expected communication latency should be minimal. However, actual performance may be affected by software overhead, data conversion, and system processing delays. This test consisted of executing the force torque reading command in 30 batches of 1000 executions each, recording both mean execution time and standard deviation.

The results are summarized in Table 5.3 and indicate the achievable sample rate. A more detailed report including the longest execution time and the number of executions that take longer than a set threshold time per batch is provided in Section B.2.2.

Statistic	**Force-Torque Reading**
Batch Size	1000
Iterations	30
Overall Mean of Batch Means (μs)	81.281
Standard Deviation of Batch Means (μs)	1.277

Table 5.3: Statistics FTS read command execution time

The test results show that the average sample time took ≈ 82 μs, considering the additional threshold metric, the reliable duration for this polling command is 95 μs. This duration is selected because only a limited number of commands exceed it and it suits the requirement of being faster than the control loop period, ensuring recent data availability.

5.1.3 Absolute Encoder

The absolute encoder used in this testbed, to capture the angle of the drive shaft and further the angle of the link, communicates through an RS485 interface, and the manufacturer states that the time before the encoder responds to a command, for the used Baud rate of $115,200$ $\frac{\text{bit}}{s}$, is 10.8 μss. Since this just provides an upper bound for the sample rate, and position tracking is essential for accurate control, the timing characteristic of the command execution duration is evaluated to determine the feasibility of real-time position sensing. Furthermore, to assess whether the polling rate requirement from Section 4 is achievable, considering the additional processing time caused by the host module's OS or the sensor's internal delay.

Communication Speed

To assess the communication latency of the encoder position polling, the sensor is mounted to the drive shaft and connected to RPI5 through the HATs RS485 interface. The test was performed to measure the duration of sending a position request and process

a response. The test procedure was identical to that used for the actuator and FTS, capturing timestamps before and after command execution, but without the baseline test.

The results are summarized in Table 5.4 and indicate the achievable sample rate. A more detailed report including the longest execution time and the number of executions that take longer than a set threshold time per batch is provided in Section B.2.3.

Statistic	**Position Reading**
Batch Size	1000
Iterations	30
Overall Mean of Batch Means (μs)	839.901
Standard Deviation of Batch Means (μs)	0.617

Table 5.4: Statistics encoder read command execution time

The test results show that the average sample time was ≈ 840 μs, and considering the additional threshold metric, the reliable duration for this polling command is 900 μs. This period is chosen because few commands surpass it, and is sufficient for the testbed since it satisfies the requirement to be faster than the control loop period to provide recent data. Therefore, this sensor can be used for control loops up to 1kHz.

5.1.4 Mechanical Components

In order to validate the mechanical concept and the assumption that connections to the drive shaft and the drive shaft itself are rigid under load, a unique test bed configuration is used. This configuration is shown in Figure 5.4 and shows the mechanical connections between the components but not the adapters. This testbed configuration makes use of two additional adapters as detailed above. Their purpose is to connect the actuator directly to the driveshaft via a flange and the spring to the FTS. Since the spring is directly connected to the FTS via an adapter, the measurement of actuator angle θ represents the deflection angle of the spring, drive shaft and adapters together. Similarly to the static configuration, although the spring is now further away from the point of measurement. However, 3D printed adapters are still assumed to be rigid parts. By analyzing the difference between θ and `q`, insights into the deflection of the short path from the actuator to the encoder can be obtained, thanks to the measurement point of the link or drive shaft position. This distance is about 30 mm long and for this connection to be rigid the difference of theses angles should be negligible under a load. With this setup, a static test is performed and Figure 5.5 shows the torque-angle displacement path for both angles and their difference.

The graph shows 3 loops, blue and orange are the displacement graph according to θ and `q` angles and in yellow the difference of θ to `q` and black arrows display the direction of the spring deflection. The yellow plot shows the difference and exhibits linear elastic behavior, but also shows angular jumps along the pathway.

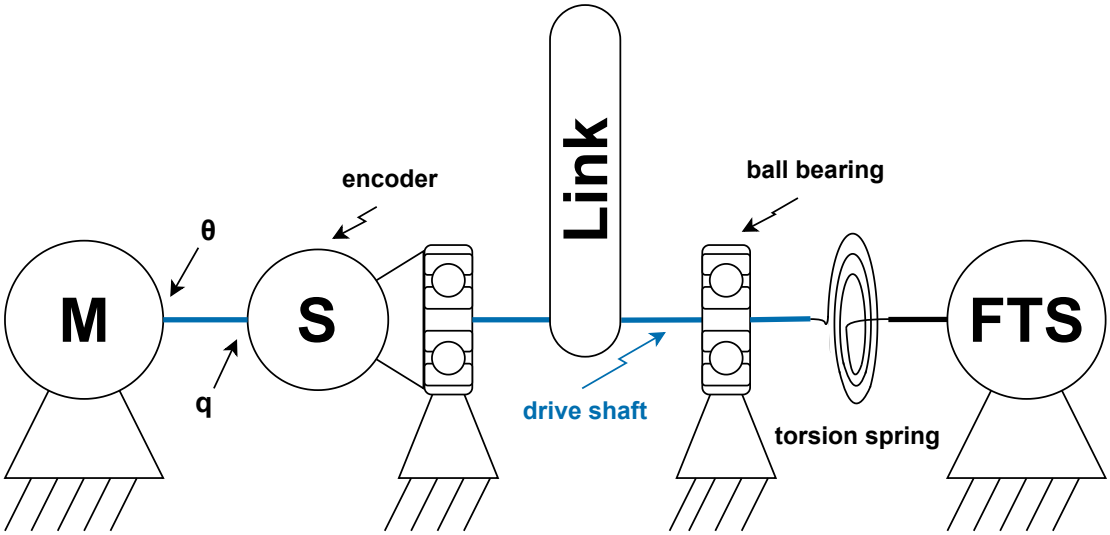

Figure 5.4: Unique testbed configuration utilized for shaft twist investigation

These jumps appear twice for each loading and deloading path. These distinct positions of sudden changes in the angular position of either q or θ along the deflection path suggest two different sources. For jumps on the deloading path and closer to the resting position, the actuator backlash is responsible, since a load change occurs and the difference is $\approx 0.14°$ high, matching the backlash angle specified by the manufacturer. The other jump is located at higher torque measurement $\approx 0.19°$ and occurs on a loading pathway. This could be due to mechanical play at the flange to drive shaft interface, i.e. caused by the manual fabrication process of the driveshaft groves. Both sudden changes cause a jump in the same direction of angle difference. This is due to the calculation of the difference from θ to q, a backlash change causes the actuator angle to increase while q is unchanged and a sudden change caused by the flange connection decreases the q angle while θ is unchanged.

However, the results also suggest elasticity of the drive shaft. In Section A.1.1 an approximation has been calculated for the examined section deflection. This approximation assumes a circular driveshaft profile and therefore underestimates the elastic behavior since the drive shaft has a flat side. The underestimated value of the angle of twist of the drive shaft is 0.17° under a torsional load of 3 Nm, representing the maximum torque of this test. The data suggest the angle of twist to be higher. When adjusting for angular jumps, the twist angle measures $\approx 0.33°$. Although the calculated difference and the approximate deflection of the analyzed section come close to the magnitude of the sensor accuracy, it showed a distinct reproducible behavior. In addition, all springs designed and used in this project have been measured and analyzed with this configuration, further underlining the above-mentioned but also showing variability of the flange connection, which could be caused by misalignment and screw tightening in manual assembly.

This analysis leads to the conclusion that the chosen interface method lacks mechanical

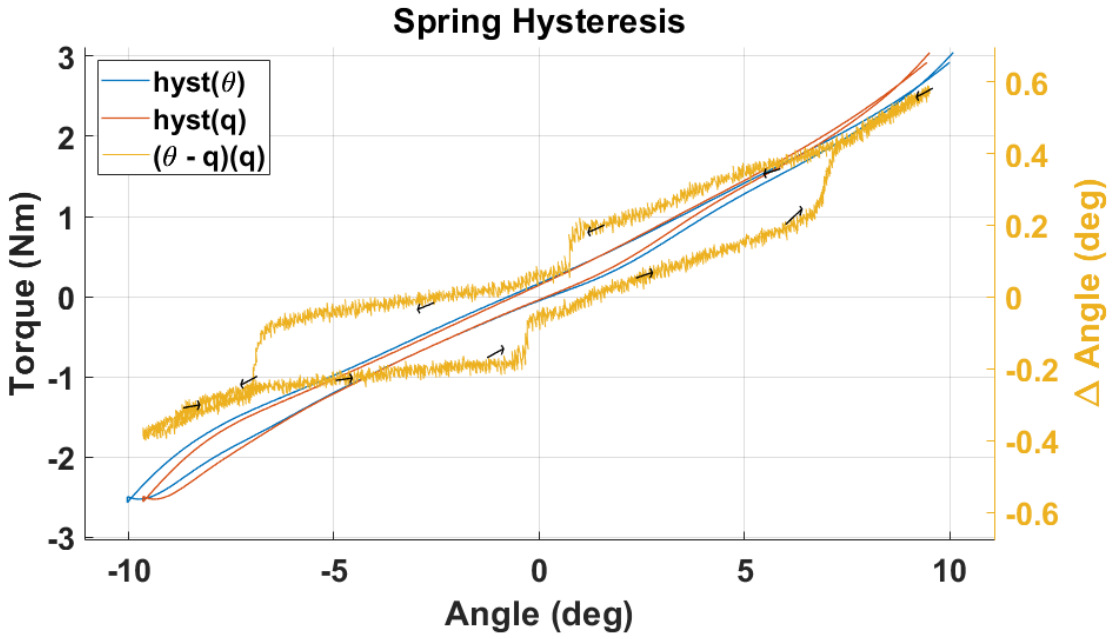

Figure 5.5: Difference of angle measurements in spring hysteresis

stability and that the drive shaft exhibits considerable elasticity. Therefore, the assumption of a rigid drive shaft does not hold. As determined in Section A.1.1, the estimated torsional stiffness value of the drive shaft is approximately 201.05 $\frac{\text{Nm}}{\text{rad}}$. Consequently, the springs tested in this configuration experience a different deflection angle than the actuator angle measurement θ suggests. Since the measured deflection includes the combined deformation of both the spring and the drive shaft, the testbed effectively captures the stiffness of a series combination of the two components. With additional measurements of the drive shaft stiffness, the actual spring stiffness could be deduced due to the series connection. This correction could yield more accurate results, assuming linear behavior of both components.

The same effect is also present in the static testbed configuration used to determine stiffness. Although shaft rotation is restricted, a movement of the link is viable pointing to the matter discussed above. An approximation of the maximum angle of twist under the actuator's torque limit of 10 Nm is performed in Section A.1.1. This approximation assumes a steel shaft with a circular cross section. The drive shaft would twist for about 2.85°, significantly compromising the measurement. However, in this setup, the link angle measurement is captured closer to the spring, providing an opportunity to correct the measured deflection angle.

5.2 Real-Time Software System

This section evaluates whether the software for static and dynamic tests meets the intended real-time requirements. Ensuring real-time capability is critical for both data integrity and control tasks. The evaluation is based on the analysis of the time difference between the timestamps captured at the start of each ISR function call. These timestamps are saved and exported to a text file. The difference between these timestamps is calculated and visualized in a graph, providing a clear representation of timing consistency, but also if the ISR execution duration exceeds the timer interval. Additionally, a mean duration and the corresponding standard deviation of these time differences are derived and displayed in the graph to quantify variations in timing precision. Any deviation from the set timer period would indicate potential interference from the operating system or communication delays, which affects the reliability of the real-time system and further control tasks or data integrity.

5.2.1 Static Test Analysis

The static test data is obtained from a static test, capturing the torque-deflection path, where the timer period was set to 2ms. In this test, the software executed a position update every 40ms. These parameters were chosen to provide higher-resolution data, even though the subsequent execution of a measurement and position update exceeds the timer period for these cycles. Since this specific test with its parameters was not intended to capture a dynamic response of a spring, the temporal inaccuracy in subsequent periods is negligible. Figure 5.6 visualizes the difference in static ISR start timestamps, highlighting the periodicity and any potential variations, including a zoomed in view.

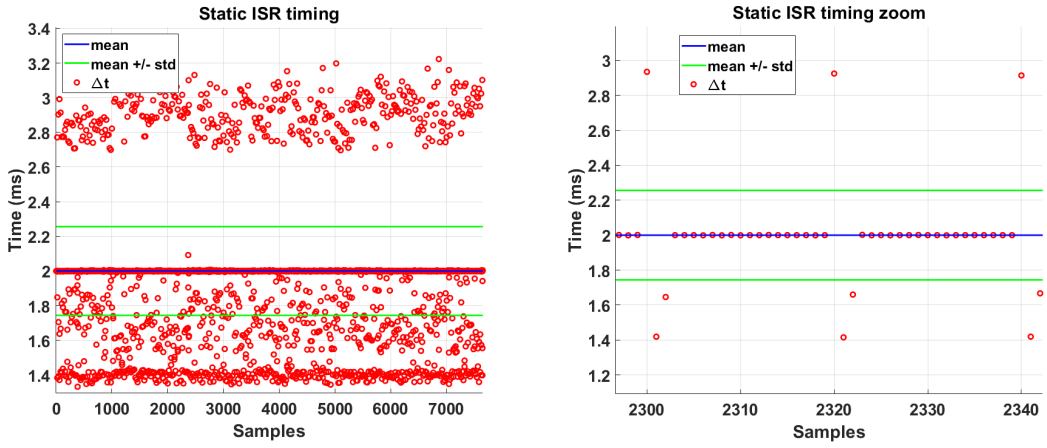

Figure 5.6: Static ISR timing, overall(left), zoom(right)

The graph visualizes the expected behavior of periodical interferences placing the time delay in the range of the expected ≈ 2.8 ms when communicating twice with the actuator in one ISR cycle. Furthermore, this duration is extended even further, due to OS interference

or process times of the algorithm. The subsequent shorter time difference, visible in the zoomed graph, follows due to the timer interrupt architecture; since self-interruption is prohibited, the ISR will execute as soon as possible after the current execution yields.

This test result shows that a 2 ms timer period or control clock is not an optimal interval but produces adequate data for a static test that does not evaluate dynamic loading performances. This also means that dynamic tasks will not be able to perform correctly at this clock rate, since the timing requirements for these tasks are higher.

5.2.2 Dynamic Test Analysis

The dynamic test data is collected from the sinusoidal test, discussed in Section 5.1.1. This test is relevant for evaluating control performance as it involves continuously updating actuator positions while reading the angle position using the dynamic test software. Although the testbed components were not mechanically linked, this does not have an impact on the evaluation. The test software was extended to feature additional capabilities to change the update rate while performing this test. This was implemented in a way which could interrupt the control algorithm and therefore introduce delays when changing the parameters. The parameter change is performed with user interaction via the terminal and includes the adaptation of the position update rate but not the timer period. The test data included changes from 4 ms to 16 ms, while the control period was set to 4 ms. The data collected is analyzed using the same method utilized in the previous section and shown in Figure 5.7 similar to the static test, including a zoomed in view.

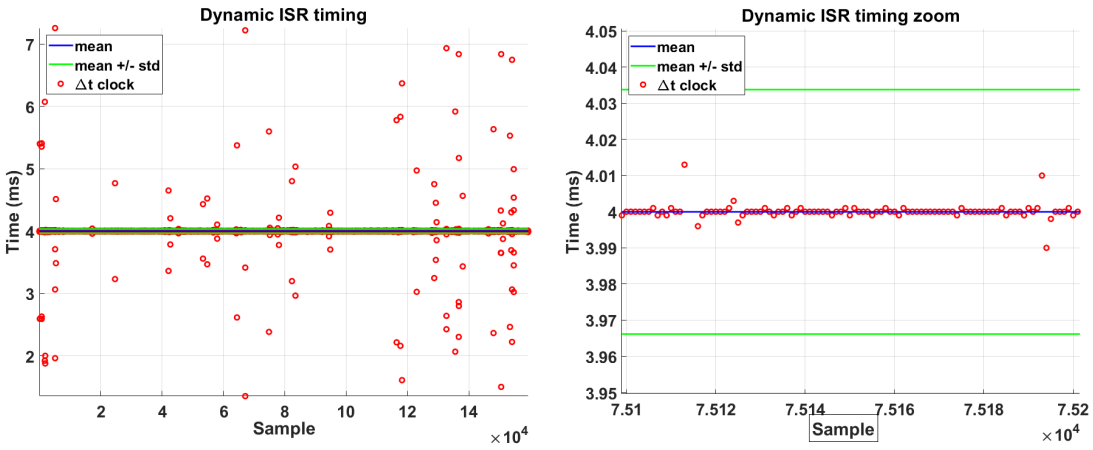

Figure 5.7: Dynamic timing analysis

This graph shows that the timer consistency holds up and just a few data points show delays that are higher than before. The ratio of delayed cycles compared to regular cycles is minimal and seems to be caused by processing the user interaction. However, a closer look at the graphs reveals a slight deviation from the control clock, of magnitude 20μs, and is considered insignificant for control tasks using a `ms` clock. Therefore, this test

suggests that the chosen timer period is well suited to handle control tasks, although this cannot be guaranteed since the control box is left for a user to implement and has the potential to introduce significant processing times, adding to timing consistencies.

5.3 Static Spring Results

In this project three springs, each of a different material, have been designed and tested to evaluate whether it is possible to use the above-mentioned tool [1] to generate linear torsion springs using 3D-printing. Therefore, these springs are assessed with the static testbed configuration and software to analyze their stiffness. Each spring was designed with the same target stiffness of $20\frac{\mathrm{Nm}}{\mathrm{rad}}$, the design parameters are mostly the same, besides their material parameters, and differ slightly in their `Tip-to-Cam Gap` value, which will have an impact on mechanical play. Figure 5.8 lists these springs and their material. A detailed list of all design parameters is added in the Appendix B.3.

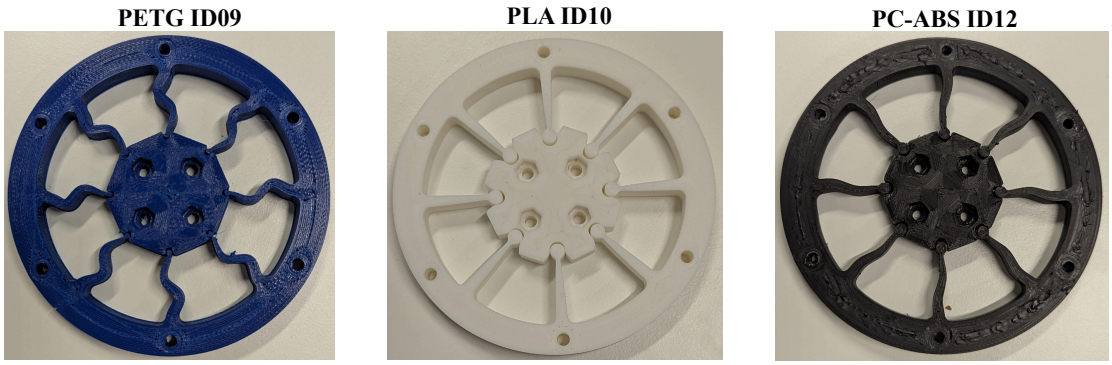

Figure 5.8: Tested springs

5.3.1 Test Parameters

Each spring is tested using the same parameters for the static test, which are detailed in Table 5.5. These settings were selected to approach, but not exceed, the common maximum allowable deflection the tested springs, ensuring consistent and meaningful comparisons. The test procedure increments the deflection angle in small steps of $\Delta\theta = 0.1°$ up to a maximum deflection of $\theta_{\mathrm{max}} = 10°$. The system updates the target position every 40 ms, while samples are recorded at a higher resolution of 2 ms. To account for variability and enable statistical robustness, five repetitions (loops) of the entire deflection sequence are conducted for each spring. This configuration is not meant to stress test a spring, but rather to get an idea of how the design tool and the spring design performs for the chosen materials and design.

Parameter	Unit	Value
Max deflection $\theta_{\max}$	deg	10.0
Angle step size $\Delta\theta$	deg	0.1
Position update rate	ms	40
Sampling time	ms	2
Number of loops	-	5

Table 5.5: Stiffness test parameter

5.3.2 Static Data Processing

This section details the analysis of the data generated with the static testbed configuration from static tests to determine static spring characteristics, including local stiffness and hysteresis losses.

Local Stiffness

The test procedure results in multiple loops of measurement of torque and angular displacement. In this work, five loops have been recorded. Due to sensor noise and mechanical effects, the raw data contains fluctuations that would distort the derivative-based evaluation of stiffness. To address this, the data is first smoothed using a `Gaussian Process Regression (GPR)`. This nonparametric regression method provides a continuous and smooth estimate of the measured signal by modeling it as a distribution over functions, conditioned on the observed data points. Inherently, regression can lead to overfitting or oversmoothing, potentially obscuring data features that are part of a spring's behavior, such as sudden changes in stiffness or torque spikes. In addition to the smoothed Torque signal $T_\mu(\theta)$, GPR provides a standard deviation $\sigma_T(\theta)$ for each predicted point. This value σ_T represents the model's predictive uncertainty and is later used for propagation of the measurement uncertainty in the stiffness estimation. A higher value of σ indicates regions of greater uncertainty due to a low number of data points or noisy data.

This method allows to predict the smoothed $T_\mu(\theta)$ values at a common angular data set over all loops, enabling direct comparison and mean calculations. To ensure that the hysteresis behavior is not artificially removed during regression, the measurement loops are split into two segments: the upper and lower halves of the hysteresis curve, and each half-cycle is treated independently at the same set of angles.

Following regression, the local stiffness is calculated for each half-loop by taking the discrete derivative of the smoothed torque–angle relationship using a common angular dataset, referred to as θ_{grid}. The local stiffness is calculated as

$$k(\theta_{i+\frac{1}{2}}) = \frac{\Delta T}{\Delta\theta} = \frac{T_\mu(\theta_{\text{grid},\,i+1}) - T_\mu(\theta_{\text{grid},\,i})}{\theta_{\text{grid},\,i+1} - \theta_{\text{grid},\,i}}. \tag{5.2}$$

The resulting stiffness value $k(\theta_{i+\frac{1}{2}})$ is interpreted to be at the midpoint between two neighboring angle values, $\theta_{i+\frac{1}{2}} = \frac{1}{2}(\theta_i + \theta_{i+1})$. This midpoint assignment is necessary, as the numerical derivative is inherently defined over intervals between points. The procedure is applied to each half of every measurement loop, resulting in a set of loop-wise stiffness curves.

In a final step, the average stiffness curve is calculated for each hysteresis half by computing the point-wise mean of the individual curves. This averaging provides a representation of the spring behavior while reducing the influence of outliers and noise. It is important to note that the calculation of the average stiffness will obscure certain information about the response of a spring to repeated loading cycles, such as softening. Hence, the average stiffness should be viewed together with the smoothed or even raw loop data.

Hysteresis Losses

To further characterize the energy dissipation in the elastic element, hysteresis losses are estimated from the mean hysteresis loop. This is done by numerically integrating the upper and lower halves of the torque–deflection curve and computing the enclosed area between them.

5.3.3 Accuracy/Uncertainty Propagation for Local Stiffness

Since detailed information on sensor accuracy and error bounds are not provided by the manufacturers, a precise error propagation analysis is not possible. Therefore, the following uncertainty analysis represents an approximation based on the assumed signal behavior. In order to approximate the accuracy of the described data processing method and the resulting stiffness calculation, uncertainty propagation is applied, as both torque and angle measurements are subject to noise. Static directional errors such as backlash at the actuator output shaft are not considered, since they cancel out due to calculating a derivative. It is assumed that the uncertainty for each torque value is provided by the standard deviation $\sigma_T(\theta)$ predicted by the Gaussian Process Regression.

The actuator manufacturer does not provide an uncertainty metric for the angular measurement. Therefore, the uncertainty is estimated from a dynamic test in which the actuator moved and held various positions within the workspace while connected to a spring and a weight-loaded link. The captured angular measurements are analyzed by considering the positions where the system was at rest. Within one window the noise is recorded as the highest difference to this windows mean angular position. The average noise over all positions is assumed to be a sufficient estimate for the following uncertainty propagation. The deeper investigation of noise for this calculations is considered to be out of scope for this work and thus not addressed in this thesis. However, details of this estimate are provided in Section A.2. The resulting angular uncertainty $\delta\theta = 0.017177°$ ($2.9980 \cdot 10^{-4}$ rad) is used for subsequent uncertainty propagation, which is performed according to Gauss' law.

The propagated stiffness uncertainty $\sigma_k(\theta_{i+\frac{1}{2}})$ is calculated using the following formula

$$\sigma_k(\theta_{i+\frac{1}{2}}) = \sqrt{\left(\frac{\sigma_{T_i}}{\Delta\theta}\right)^2 + \left(\frac{\sigma_{T_{i+1}}}{\Delta\theta}\right)^2 + 2\left(\frac{\Delta T \cdot \delta\theta}{(\Delta\theta)^2}\right)^2}, \tag{5.3}$$

where

$\Delta T = T_\mu(\theta_{\text{grid},\,i+1}) - T_\mu(\theta_{\text{grid},\,i})$	...	is the difference of the smoothed torque values,
$\Delta\theta_{=}\theta_{\text{grid},i+1} - \theta_{\text{grid},i}$	...	is the spacing of the interpolation grid,
$\sigma_{T_n} = \sigma_{T_n}(\theta_{\text{grid},\,n})$	...	is the predictive uncertainty of torque from GPR,
$\delta\theta$	...	is the experimentally found uncertainty in the angle measurement.

As a final step, the stiffness and associated uncertainty are calculated separately for each half-loop and then averaged across all repetitions. Assuming the measurements are uncorrelated, the uncertainty of the mean stiffness $\sigma_{k,\text{mean}}$ is computed as

$$\sigma_{k,\text{mean}} = \frac{1}{n} \cdot \sqrt{\sum_{i=1}^{n} \sigma_{k_i}^2}, \tag{5.4}$$

where n is the number of loops for the respective half of the hysteresis cycle.

This process provides an estimated accuracy or uncertainty of the spring's local stiffness behavior by propagating errors for each point of the interpolation grid. This process depends on the number of cyclic loops performed during the measurement. Equation 5.3 also shows that this accuracy calculation is highly dependent on the spacing of the interpolation grid $\Delta\theta$ in relation to the measured stiffness value ΔT and the uncertainty in angle measurement $\delta\theta$.

Therefore, it is not possible to provide a general uncertainty for stiffness calculations. It is also worth noting that the only user-changeable variable is the spacing of the interpolation grid, which indirectly affects both the resolution and the effective range of testable springs. Specifically, the grid spacing determines over which angular interval the stiffness is calculated, and thus influences the impact of sensor noise on the result.

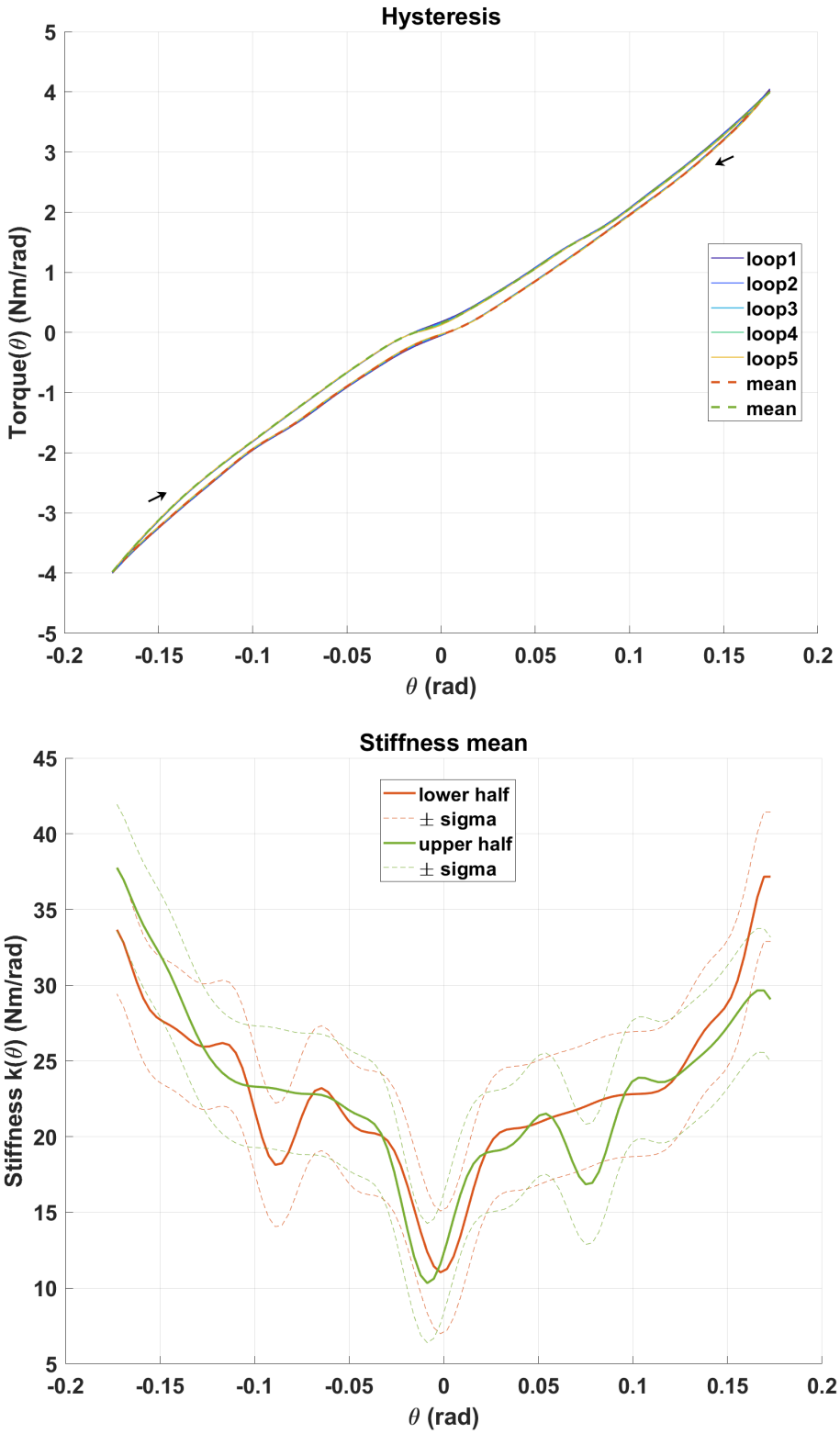

Figure 5.9: Hysteresis and stiffness of spring PETG ID09

5.3.4 Stiffness Results and Conclusion

The data obtained from the testbed was processed as described earlier, with a grid angle increment of $\Delta\theta_{\text{grid}} = 0.1°$. This increment was selected to improve accuracy by maintaining a ratio of $\approx 1 : 10$ for the angular component of error propagation, considering the noise present in angular measurements. This choice achieves a suitable trade-off between resolution and precision for local stiffness across five measurement loops. A different trade-off could be made when higher resolution is desired by increasing number of loops recorded.

Figures 5.9, 5.10 and 5.11 in this section show two graphs, first the hysteresis curve which displays all loops captured and the mean loop for each half cycle. Arrows indicate the starting point and direction of loading. In the bottom graphs the mean stiffness over all loops of each half with the resulting $\sigma_{\text{k,mean}}$ is shown.

Table 5.6 summarizes the results for all the springs discussed in this work. An additional metric has been calculated for each spring the approximate maximum twist angle of the shaft. Since the approximate stiffness of the shaft is around $200\frac{\text{Nm}}{\text{rad}}$, the impact on these springs is minimal, as these springs are softer. Furthermore, each spring showed a mechanical play in their CAM interface. To quantify and provide a comparable metric for this behavior, this work defines and measures mechanical play. Specifically, mechanical play is determined by identifying the angular positions on either side of the load change point where the torque differs by 0.1 Nm and calculating the sum of these angular distances. This definition ensures a consistent and reproducible evaluation method for different spring samples.

All tested springs exhibit an inconsistency in their stiffness curve along the loading pathways between one and two `Nm` torque, visible as two small local minima or a change in hysteresis width. This regular and consistent occurrence is caused by a sudden change in angular displacement due to mechanical play in the flange connection when the system overcomes friction in the interface. This underlines the influence of manual manipulation of the drive shaft. However, since all loops of each spring lay sufficiently on top of each other, it can be concluded that the designed springs and selected materials exhibit consistent results under changing loads.

Moreover, within the stiffness mean graphs, each loop exhibits pronounced local minima close to the zero angle as a result of the combined mechanical play of the spring and actuator. This can also be seen in the Hysteresis path as a plateau between load changes.

Furthermore, the mechanical play causes local stiffness minima that are angularly shifted between the upper and lower half. This indicates hysteresis losses due to internal friction, viscoelastic material behavior, and possible interface imperfections between printed spring layers. A higher shift between these peaks implies a greater energy loss. Furthermore, the angular width of these minima could serve as an alternative method to determine the mechanical play of the springs.

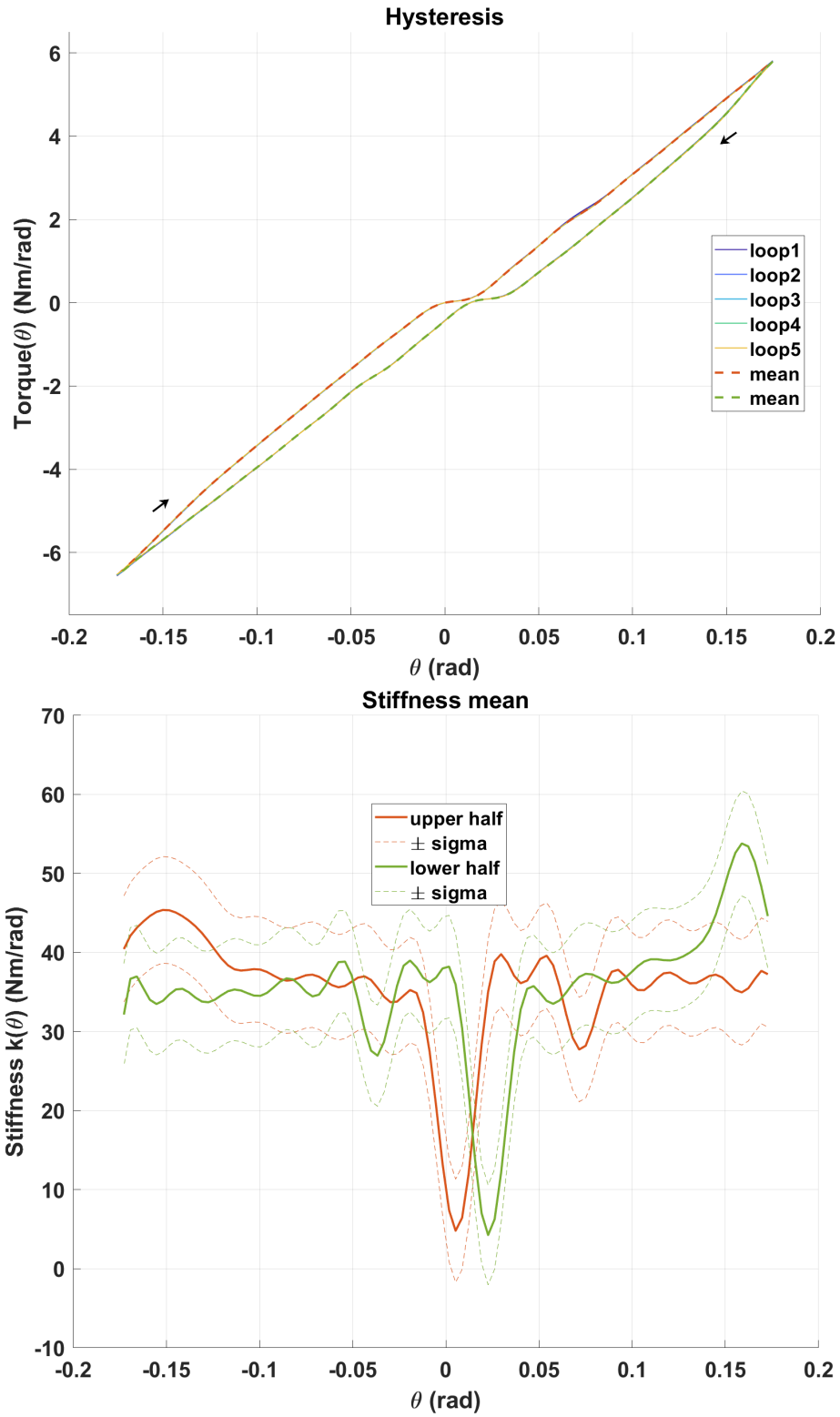

Figure 5.10: Hysteresis and stiffness of spring PLA ID10

The spring PETG ID09, shown in Figure 5.9, starts close to the general target stiffness of $20\frac{\text{Nm}}{\text{rad}}$ but exhibits noticeable nonlinearity, becoming stiffer toward the maximum deflection. This is observed as a non-linear hysteresis path and a linear rising stiffness curve. However, mechanical play was not documented for this spring, as the plateau was minimal and almost not noticeable. The matching dip observed in the stiffness plot was less pronounced compared to other springs. Both halves showed similar behavior during loading and unloading, but upon closer investigation displayed differences in stiffness values at the end points of each curve and along the path. This possibly indicates that this particular printing method or material introduced directional dependencies, giving this spring a possible favorable deflection direction.

The spring PLA ID10, shown in Figure 5.10, resulted in approximately twice the target stiffness. This material exhibited the highest energy losses among all tested springs, about 6 times higher compared to the other springs. This losses could have increased due to larger camshaft tips, resulting in greater surface friction at this interface. This spring shows mostly linear behavior and the symmetry between the loading and unloading pathways is more consistent compared to the spring PETG ID09, although there were slight deviations, which could indicate a favorable direction for this spring as well.

The spring PC-ABS ID12, shown in Figure 5.11, came closest to the target stiffness and exhibited mostly linear behavior and aside from mechanical play this spring had low hysteresis losses. However, the stiffness curve showed noticeable sharp ripples, hinting at rough surface conditions. Interestingly, these ripples appeared predominantly in one loading direction, suggesting a possible influence of the printing direction of the outer perimeters. The material `PC-ABS`([25]) demonstrated superior performance compared to the others and should be further investigated.

Parameter	PETG ID09	PLA ID10	PC-ABS ID12
Stiffness ($\frac{\text{Nm}}{\text{rad}}$)	$\approx 20 - 35$	≈ 36	≈ 20
Hysteresis Loss ($\text{Nm} \cdot \text{rad}$)	0.0438	0.283	0.0523
Cam Play (rad | °)	0	0.021 | 1.2	0.0314 | 1.8
Max Torque (Nm)	4	6.63	3.3
Max Shaft Twist (rad | °)	0.0198 | 1.139	0.0329 | 1.889	0.0164 | 0.854
Tip-to-cam Gap (mm)	0.1	0.1	0.2
Minimum Tip Radius (mm)	1	2.5	2
Comment	nonlinear	most linear	mostly linear

Table 5.6: Static test results of springs

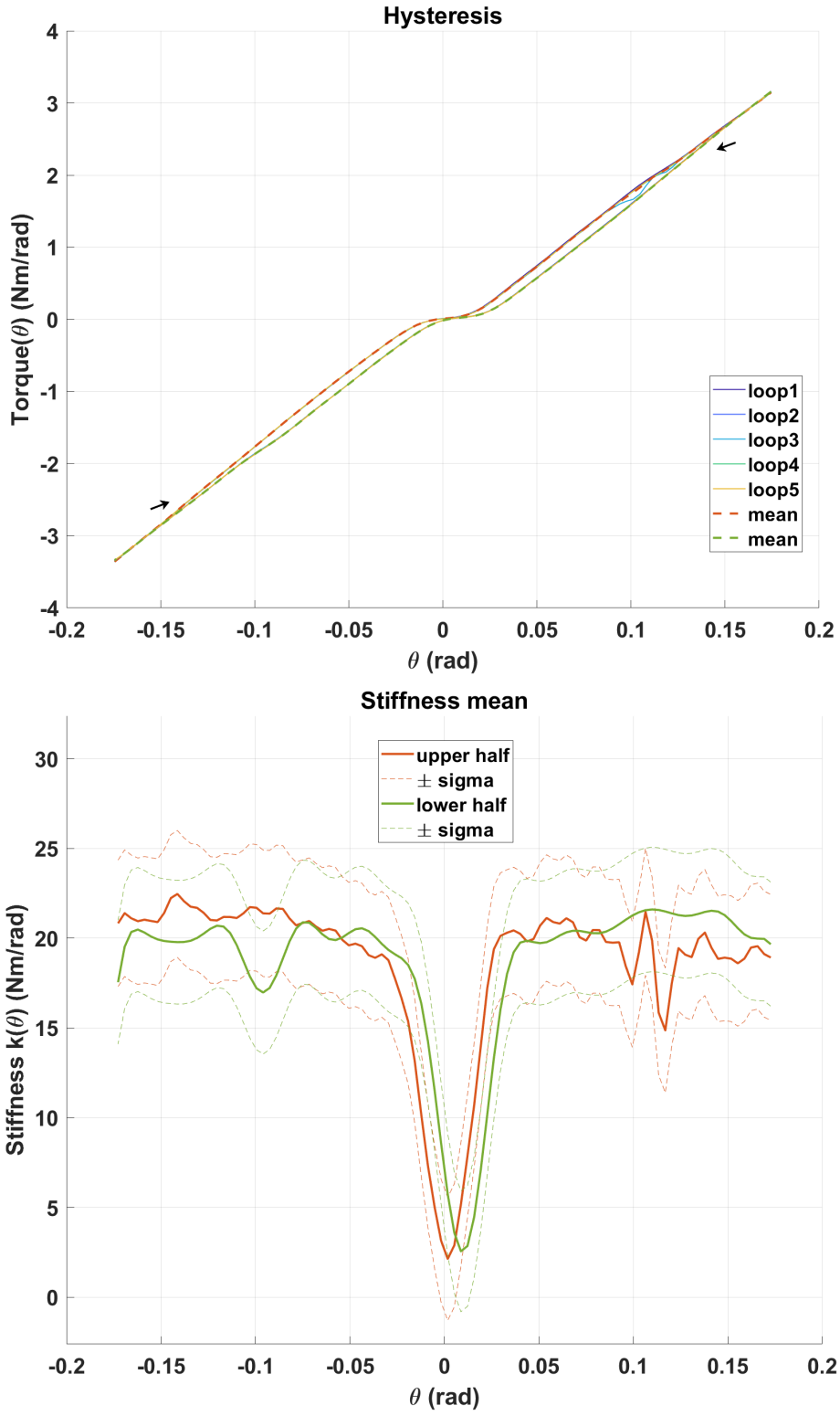

Figure 5.11: Hysteresis and stiffness of spring PC-ABS ID12

This analysis demonstrates that it is feasible to fabricate linear springs using the spring design tool from [1] in combination with 3D printing, although minor modifications to the original designs are required. Furthermore, the target stiffness of $20\frac{\text{Nm}}{\text{rad}}$ was only reached with spring design `PC-ABS ID12`. PC-ABS appears to be a promising candidate for applications that prioritize energy recovery and accuracy design. The selection of a spring material is beyond the scope of this thesis, and a detailed analysis is subject to future work. Moreover, the observed mechanical play in the cam connection of the springs constitutes an undesirable effect, which could potentially be mitigated through further design adaptations and print settings adjustments. However, such modifications may influence the stiffness characteristics and energy dissipation of the springs. These aspects need further investigation in future work.

5.4 Gravity Compensation

In order to position the link and its load accurately with a controller, a mathematical model is required. Therefore, the gravitational impact on the system is modeled to predict the position of the link due to spring deflection. This model is evaluated in the dynamic testbed configuration by commanding the actuator to discrete positions in 10° increments and comparing the model-predicted link positions with actual link angle measurements. Figure 5.12 illustrates the real-world setup with the chosen coordinate system and additionally a sketch indicating angular polarity.

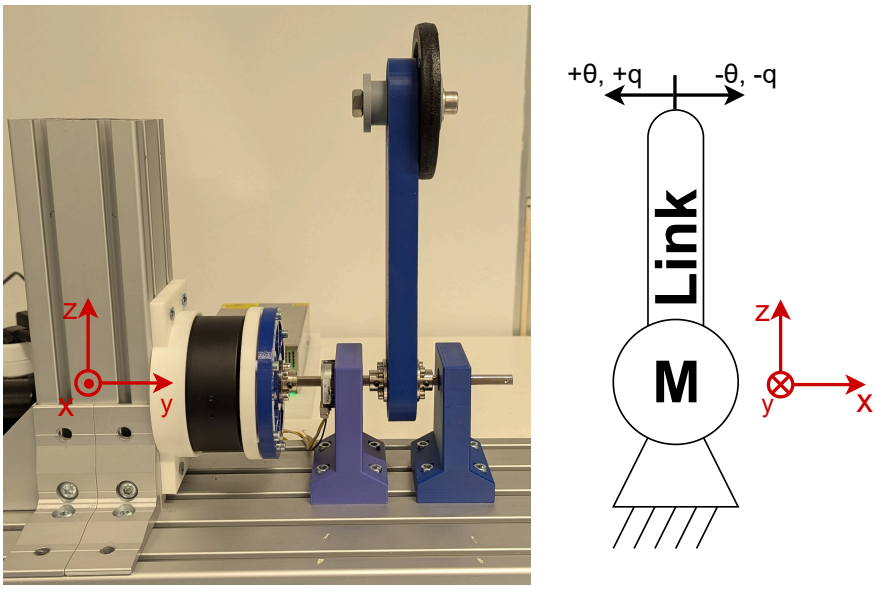

Figure 5.12: Gravity-compensation test setup with coordinate system

5.4.1 Model

The link and the added weight are assumed to be bodies with a homogeneous mass distribution and symmetric shape. This enables approximation by solely considering their centers of mass. With the distance to these centers, the gravitational pull on the link applies a torque to the springs via the drive shaft. Thus, the gravitational torque $T_g(q)$ concerning the angular polarity in the coordinate system of this setup, can be expressed as

$$T_g(q) = -(m_{\text{link}} \cdot l_{\text{link}} + m_{\text{load}} \cdot l_{\text{load}}) \cdot g \cdot \sin(q), \tag{5.5}$$

where

q	...	is the link angle measured from the upright position,
$m_{\text{link}} = 219$ g	...	is the link mass,
$l_{\text{link}} = 100$ mm	...	is the distance to link mass point,
$m_{\text{load}} = 500$ g	...	is the load mass,
$l_{\text{load}} = 200$ mm	...	is the distance to weight mass point,
$g = 9.81$ m/s^2	...	is the gravitational acceleration.

With this model, the deflection of a spring Δq, due to the gravitational pull, can be calculated as

$$\Delta\, q = \frac{T_g(q)}{k}, \tag{5.6}$$

where k is the linear spring stiffness. This enables the compensation of angular displacement due to gravity.

5.4.2 Spring

For this evaluation a previous print of the spring `PTEG ID09` was used (cf. Fig. 5.14). This spring shares its design with the spring discussed in Section 5.3 but was printed under different conditions. For this reason, it was not included in the previous discussion and underlines the manufacturing deviations typical for 3D-Printing. This spring was analyzed using the same procedure as previously discussed.

The validation of gravity compensation is performed in the linear stiffness region of this spring from 0° to −90°. Since the spring shows changing stiffness values for this region, the maximum torque experienced by the spring in this test is taken into account for stiffness approximation. Using Equation 5.5 and substituting the given values, the max torque caused by the gravitational pull at −90° is

$$T_g(-90°) = -(0.219 \cdot 0.1 + 0.500 \cdot 0.2) \cdot 9.81 \cdot \sin(-90°) = 1.196\text{Nm}. \tag{5.7}$$

This result is used to find the maximum deflection of the spring in the torque deflection relationship(cf.5.14 hysteresis). The maximum deflection for equilibrium positions will be −0.075 rad. Analyzing the stiffness graph (cf.5.14 Stiffness mean) for the interval from −0.075 to 0 rad, the stiffness of the spring is linearly approximated with 14.6 $\frac{\text{Nm}}{\text{rad}}$.

5.4.3 Validation

The actuator was positioned with the dynamic test software from 0° to −90° and the actuator angle and the link angle were recorded. This data was analyzed, and the predicted link position was calculated as

$$q_{\text{predict}} = \theta + \frac{T_g(q)}{k}, \tag{5.8}$$

where θ is the motor position.

With this equation, the difference between the link position and the predicted link position q_{predict} is calculated as

$$q_{\text{err}} = q_{\text{predict}} - q \ , \tag{5.9}$$

and displayed in Figure 5.13.

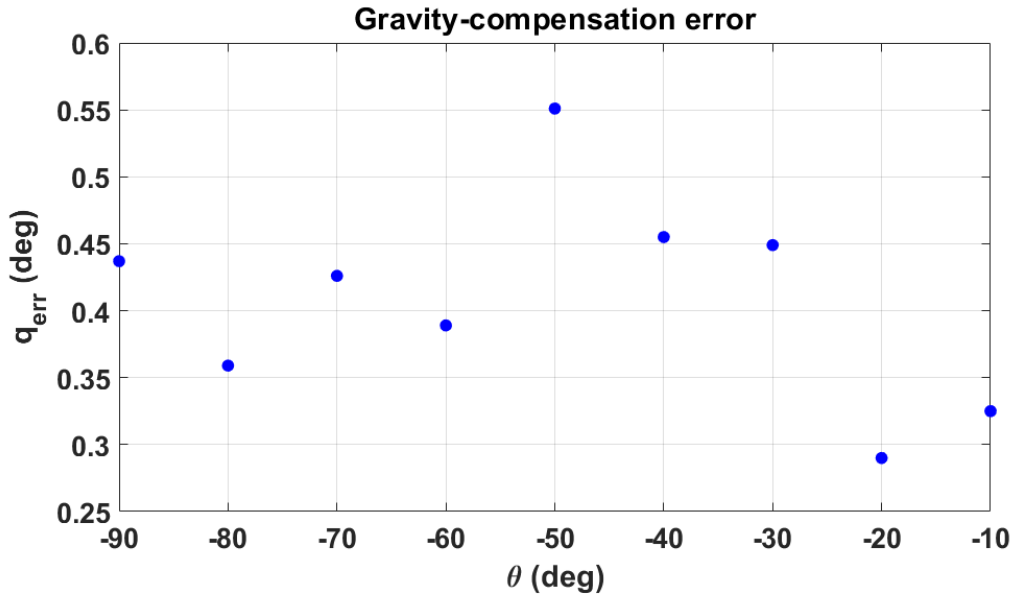

Figure 5.13: Gravity-compensation error for various actuator positions

The gravity compensation test demonstrated that the simplified stiffness model is capable of predicting the link position under gravitational loading with reasonable accuracy. Despite small deviations—primarily attributed to local stiffness variations and manufacturing tolerances—the maximum observed position error $q_{\text{err}} \approx 0.5°$ at $\theta = -50°$, which is within the acceptable range given the sensor resolution and system uncertainty.

As a result, it can be concluded that the linear stiffness model, combined with a gravitational torque estimation, is sufficiently accurate to serve as a basis for future control implementations. This finding supports the potential of using model-based compensation strategies in robotic joints equipped with compliant elements.

5.5 Damping Test

5.5.1 Test Procedure

To assess the dynamic damping properties of the springs, a manual excitation test was performed. Each spring was manually deflected while rigidly connected to the base of the testbed via a dedicated adapter and subsequently released. Allowing a spring to oscillate freely around its upright equilibrium position. The link angle `q` was recorded using the absolute encoder at a sampling rate of 1 kHz. Since the actuator is not used for this test to avoid it's damping effect on the system the software was able to perform at this sampling time.

This test was performed for the three springs previously characterized in Section 5.3 in two inertia configurations. Once for I_0 consisting of only the link and I_{500} with an additional weight of 500 g to the link. Inertia was estimated by approximating the link as a cylindrical rod and the added weight as a hollow cylinder. The moment of inertia with respect to the rotational axis of the drive shaft was calculated using the parallel axis theorem. The resulting values are $I_0 = 3 \cdot 10^{-3}\,\mathrm{kg\,m^2}$ and $I_{500} = 24.4 \cdot 10^{-3}\,\mathrm{kg\,m^2}$.

The dynamics of the freely oscillating link in the damping test can be approximated, assuming linear spring stiffness k, and purely viscous damping d and can be written as equation of motion like

$$I\ddot{q}(t) + D\dot{q} + k(q - q_0) + T_g(q) = 0, \tag{5.10}$$

where I is the rotational inertia of the link, q its angular position, q_0 the spring's rest position, and $T_g(q)$ the gravitationally Torque due to the links position.

The observed free oscillations of the deflected link(cf. Fig. 5.15) can be described as a damped rotational system. The solution of this equation for damped oscillations is a damped cosine function of the form

$$q(t) = Ae^{-\lambda t}\cos(\omega_d t + \phi), \tag{5.11}$$

where A is the initial amplitude of the free oscillation, λ is the decay constant, ω_d the damped natural frequency, and ϕ a phase shift. In this work, the damping behavior is evaluated by fitting only the exponential envelope of the oscillation, i.e.,

$$q_{\mathrm{fit}}(t) = Ae^{-\lambda t}, \tag{5.12}$$

which represents the hull curve of the decaying amplitude. This fitting approach isolates the decay constant λ, providing a direct measure of how quickly the system loses energy due to damping. In this approximation we neglect the influence of gravity since the amplitude envelope is primarily governed by viscous damping. Additionally, the oscillation period is not measured as the difference between local minima.

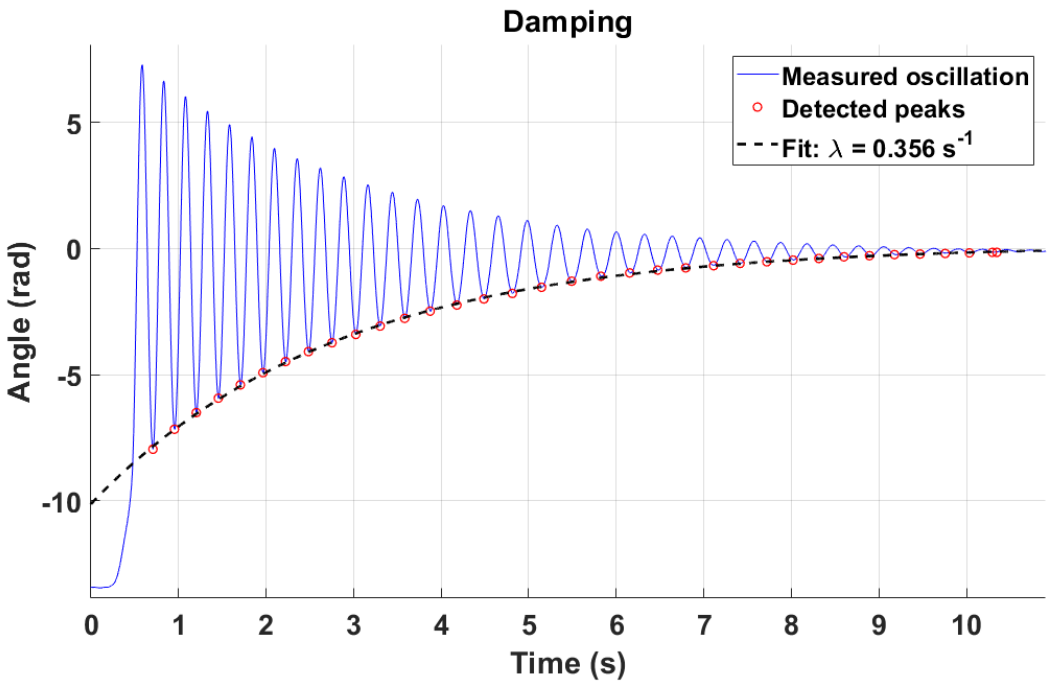

Figure 5.15: Oscillatory response of the mass-spring system incorporating the spring `PC-ABS ID12` with link inertia $I_{500} = 24.4 \cdot 10^{-3}\,\mathrm{kg\,m^2}$

5.5.2 Analysis and Conclusion

The extracted exponential decay rates λ were compared for the springs tested. This comparison provides insights into material differences in dynamic damping behavior.

Spring ID	Inertia	$\boldsymbol{\lambda}$ **(s^{-1})**	$\boldsymbol{R^2}$	**Oscillation Period (ms)**
PETG ID09	I_0	3.778	0.993	88.363
PLA ID10	I_0	3.116	0.997	74.918
PC-ABS ID12	I_0	1.175	0.997	114.837
PETG ID09	I_{500}	1.359	0.970	247.036
PLA ID10	I_{500}	1.014	0.998	207.684
PC-ABS ID12	I_{500}	0.356	0.999	283.447

Table 5.7: Damping test results for I_0 and $I_5$00 with additional weight.

The results (Table 5.7) reveal that the PC-ABS spring exhibits the lowest damping (lowest $\lambda_0 = 1.175\ \mathrm{s^{-1}}$ and $\lambda_{500} = 0.356\ \mathrm{s^{-1}}$) in both inertia configurations, indicating superior

energy preservation over the other springs. In addition, the oscillation period reflects the effective stiffness-to-inertia ratio of the system, since a softer spring or a heavier link results in a longer period. Where the softest spring PC-ABS also showed the longest periods for both configurations $T_0 = 114.837$ Hz and $T_{500} = 283.447$ Hz.

In contrast, PLA and PETG show significantly higher decay rates, suggesting higher internal damping losses. This observation supports the static findings discussed in Section 5.3, where PC-ABS demonstrated not only a linear stiffness profile but also low hysteresis losses, making it a strong candidate for cyclic robotic applications. Where low damping is advantageous, such as for walking robots like *Bruce*, where restoring energy in each moving cycle is beneficial.

However, the PETG spring shows unexpectedly high damping and nonlinearity, diverging from expectations based on its design. Closer inspection revealed visible mechanical damage to the flexure arms, likely introduced by while handling the spring. This mechanical degradation likely led to additional energy losses and reduced dynamic performance. Therefore, the dynamic results for PETG are like not only caused by material behavior and must be interpreted with caution.

These findings highlight the importance of fabrication quality in evaluating material damping. While material selection plays a critical role, defects or irregularities in printed geometry can drastically alter dynamic response. Thus, further testing with a defect-free PETG spring is necessary to validate the material's suitability.

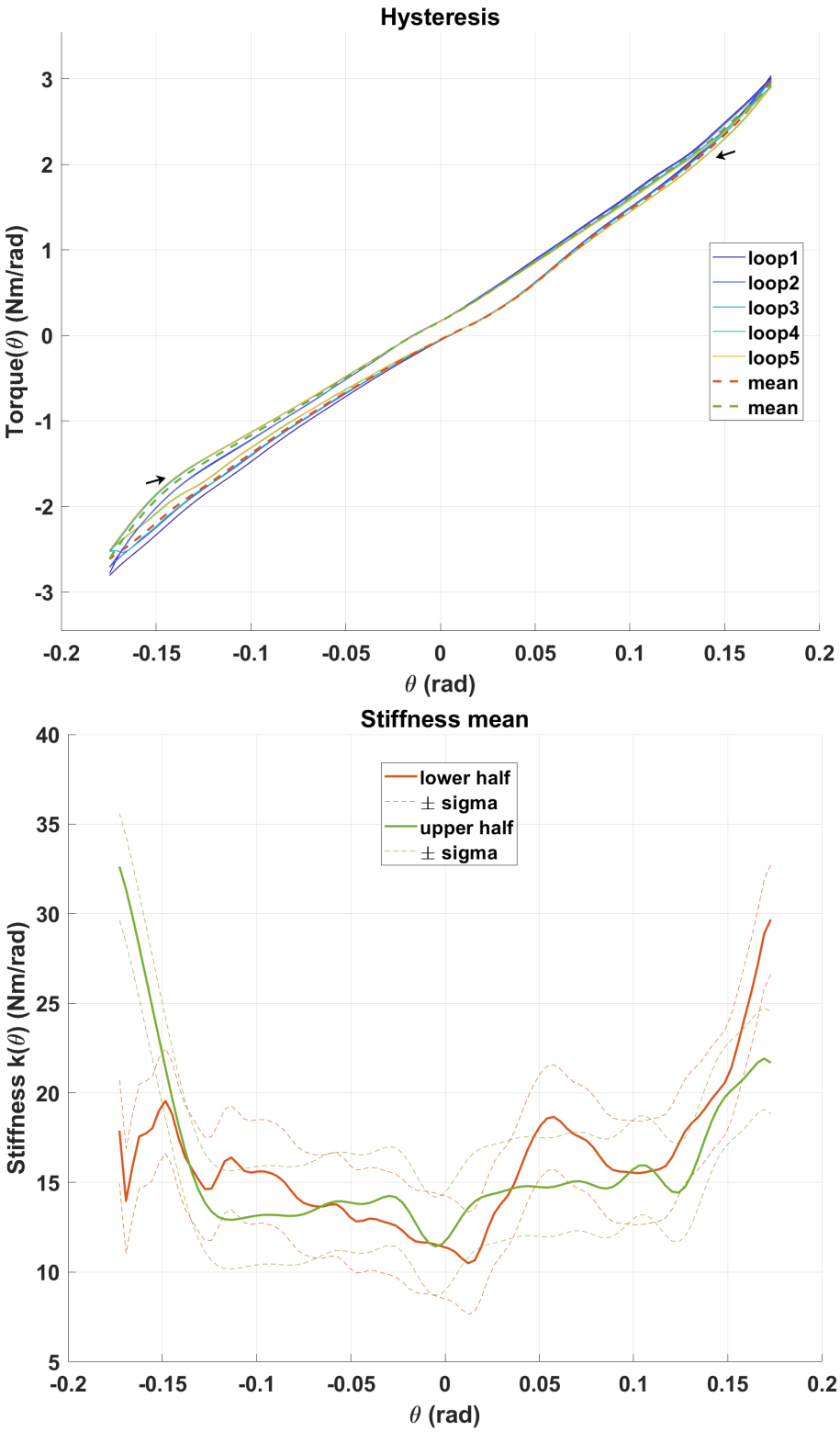

Figure 5.14: Hysteresis and stiffness of spring PETG ID09 utilized for gravity-compensation validation

6 Testbed Limitations

The developed testbed enables the evaluation of 3D-printed torsion springs; however, its performance is subject to several inherent limitations. These limitations arise from hardware constraints, software architecture, measurement uncertainties, and the mechanical structure of the testbed itself. This chapter summarizes the key restrictions identified during validation.

6.1 Hardware Limitations

The hardware used for this testbed features the following limitations:

- **The actuator** provides a actuation torque up to 10 Nm, possesses a backlash of 0.167° and a angular resolution of 0.01°. The output shaft angle sensor noise was experimentally found to be 0.017177°.
- **The FTS** provides torque sensing up to 10 Nm with a resolution of 0.015 Nm and relative accuracy of 1%
- **The link-side absolute encoder** provides an resolution of a resolution of 14 bit (0.022°) with an accuracy of 0.2°.
- **The drive shaft** shows an estimated stiffness of 201.05 $\frac{\text{Nm}}{\text{rad}}$, leading to a torsional compliance of up to 2.85° at maximum actuator torque. It is mounted at a height of 80 mm relative to the base. With an effective length of about 150 mm.
- **The link** mounted on the drive shaft has a total length of 260 mm and provides a mechanism to connect weights at a distance of 200 mm relative to the main axes.

For dynamic testbed configurations, where the link can move freely, the workspace is limited by the physical architecture of the Testbed. Since the link is longer than the drive shaft height it can collide with the base. This limits the workspace for the link and was empirically found to be about 100° in both directions, measured from an upright link position.

6.2 Software Limitations

The static softwareis designed to run tasks on user parameters. These parameters have limits due to internal software processes but also physical limits due to component limits. The control frequency for tests is largely limited by the actuators communication

speed and the need to control the position and read the angle within one cycle. If consistent equidistant sampling is desired, for example, a control or damping test, the control frequency is limited to 250 Hz or lower, i.e. 4 ms period or higher. If consistent equidistant sampling is not important, like for a static test, a control frequency of 500 Hz or lower, i.e. 2 ms or higher.

The maximum deflection angle is limited by the angle range of the actuator, which is $\pm \text{INT32}_{\text{max}}/100$. This range is greater than 360°.

For static tests, the number of cyclic loops is limited by the numeric representation of $\text{INT32}_{\text{max}}$, although more loops also mean more data to save.

The dynamic softwareprovides parameters to set the workspace angle for protective measure, as mentioned above, this should be set to values equal to or lower than 100°. For tests requiring consistent equidistant sampling, the control frequency is limited to 250 Hz or lower, i.e. 4 ms period or higher. The maximum achievable motor shaft speed is 300 dps, as higher values lead to internal actuator errors.

6.3 Accuracy and Resolution

When analyzing the accuracy of the spring stiffness calculations, the hardware limitations mentioned above are only partially considered.

Static directional errors, such as motor backlash, are effectively canceled out due to the derivative-based calculation of a local stiffness. Instead, the main contributors to uncertainty are the predictive standard deviations provided by the Gaussian Process Regression (GPR) used for data smoothing and the finite spacing of the interpolation grid used for numerical differentiation. However, for dynamic tests, the motor backlash and the accuracy of the link encoder must be considered.

Furthermore, the uncertainty in the estimated stiffness is critically dependent on the torque measurement noise captured by the GPR model, the selected angular grid resolution and the inherent noise-related uncertainty in the actuator angle sensor. No single general uncertainty value can be provided for the stiffness measurements. Instead, the local uncertainty varies along the deflection range and is strongly influenced by the chosen data processing parameters, such as the interpolation grid θ_{grid}.

For the springs discussed in this work, the interpolation grid interval was set to 0.1°. This does not represent a fix value. Higher accuracy can also be achieved by increasing the number of cyclic loops.

6.4 Spring Test Range

The combination of actuator torque limits, angular interpolation resolution and driveshaft compliance defines the stiffness range for spring testing.

The theoretical maximum testable stiffness

$$k_{\max} = \frac{T_{\max}}{\Delta\theta_{grid}} = \frac{10 \text{ Nm}}{0.00175 \text{ rad}} \approx 5714 \frac{\text{Nm}}{\text{rad}}, \tag{6.1}$$

is limited by the actuation torque 10 Nm and the interpolation grid, an angular resolution of 0.1°. This maximum could be reached with this testbed if the drive shaft did not show compliance. Taking into account the current state of the testbed and the compliance of the drive shaft, springs softer than 201.05 $\frac{\text{Nm}}{\text{rad}}$ will produce sufficient results. A ratio of 1 : 10 for the spring to the shaft would yield small deviations. Therefore, springs up to 20 $\frac{\text{Nm}}{\text{rad}}$ can be tested.

The minimal testable stiffness

$$k_{\min} = \frac{T_{\min}}{\Delta\theta_{grid}} = \frac{0.015 \text{ Nm}}{0.00175 \text{ rad}} \approx 8.6 \frac{\text{Nm}}{\text{rad}}, \tag{6.2}$$

is limited by the FTS resolution of 0.015 Nm and the interpolation grid.

7 Conclusion & Future Work

7.1 Final Conclusion

This thesis presented the design, implementation, and validation of a low-cost modular testbed to evaluate torsion springs in SEAs, targeting applications in robotic joints. The testbed was realized with a flexible architecture that supports both static and dynamic testing modes, allowing systematic spring characterization and control development.

With the selected components, a real-time capable software package was created for static and dynamic tests that enabled torque-deflection measurements, revealing linearity trends and material-dependent damping. Three different linear torsion springs were designed for PLA, PETG, and PC-ABS using the tool from [1] and manufactured using FDM 3D printing. Spring stiffness was estimated by GPR, and a custom propagation model provided uncertainty bounds for local stiffness calculations. Dynamic damping was estimated by fitting a decay function to recorded spring oscillation peaks. Among the materials tested, PC-ABS showed promising results with stiffness values matching design the and low hysteresis and static damping. Despite the introduction of mechanical play for most printed springs, we demonstrated the feasibility of designing and manufacturing linear torsional springs using the tool from [1].

The software package developed in this work also includes a test template for validation of control strategies. Upon validating the utilized actuator, we unfortunately discovered that it is not suitable for control tasks because of the low achievable sampling rates in the actuator's regulation mode and the large delays in tracking modes.

Furthermore, analysis of this testbed revealed critical mechanical limitations. The drive shaft exhibited measurable compliance, approximated as a linear torsion spring with a stiffness of $\approx 200\frac{\mathrm{Nm}}{\mathrm{rad}}$, invalidating the initial assumptions of rigidity and compromising accurate interpretation of stiffness. The angular error made by the compliant driveshaft could be corrected. Since the angular position of the assumed rigid shaft is recoded, it can be used to correct the spring deflection value. However, a maximum measurable stiffness range of $\approx 5714\frac{\mathrm{Nm}}{\mathrm{rad}}$ and a minimum of $\approx 8.6\frac{\mathrm{Nm}}{\mathrm{rad}}$ was established based on actuator torque and analytic resolution.

In summary, the feasibility of manufacturing torsional springs via additive manufacturing was successfully demonstrated and the modular testbed proved effective for the evaluation of springs. The hardware constraints restrict some dynamic applications, due to the high communication delay and bad position tracking of the utilized actuator. The motor needs to be replaced by a different model with a maximum polling duration of 400 μs per command. This would allow the testbed to operate at a control frequency of 1 kHz.

Furthermore, a more accurate tracking behavior is needed to perform meaningful control tests. However, the real-time software together with the RPI5 provides the opportunity for better performance. This system establishes a basis for future SEA research and spring development.

7.2 Future Work

Based on the results and limitations discussed throughout this work, several promising directions are suggested for future work to further improve the testbed and spring evaluation.

Mechanical Interface

The assumption of a rigid drive shaft proved invalid, as evidenced by measurable torsional compliance from the shaft and mechanical play in the current shaft-flange interface. This issue compromises the accuracy of deflection angle measurements, especially under higher loads. Redesign of the drive shaft and its connection is necessary to over come this challenge. Future versions should consider a circular shaft cross section and a precision clamping interface to minimize backlash and angular slip. Commercially available components such as shaft collars or clamping rings [26] in combination with linear shafts [27] are likely to resolve the observed interface play without significant redesign, given their compatibility with current interface geometries. Figure 7.1 shows these products. Similar components could be obtained from [28].

Figure 7.1: Clamping ring and linear shaft

Shaft Deflection

In the static test configuration, the assumption of a rigid drive shaft leads to a deflection angle estimation based solely on the actuator angle θ. However, as shown in Section 5.1.4 and Figure 5.5, the drive shaft exhibits measurable torsional compliance, causing the measured angle θ to include not only the spring deformation but also the shaft's elastic twist. This systematic error can be corrected by incorporating the encoder measurement q located near the spring shaft connection. Since q reflects the output angle of the spring, the corrected spring deflection angle can be expressed as

$$\Delta\theta_{\text{spring}} = \theta - q. \tag{7.1}$$

This approach isolates the deformation attributed solely to the spring by subtracting the shaft's twist from the actuator-side measurement, but also introduces the link side encoders uncertainty to the measurement. However, the results of this adaption seem promising and do not require any redesign of the testbed only adjustment in the test software and data processing.

Real-Time System

The software infrastructure could benefit from migrating to a real-time Linux kernel. The current implementation is limited by the Raspberry Pi OS's scheduling behavior, resulting in occasional timing jitter leading to inconsistent control loop execution. A deterministic kernel would provide more reliable timing guarantees, potentially enabling greater stability for higher control frequencies and better temporal data integrity.

Actuator

The actuator used in this testbed demonstrated inadequate dynamic performance. High communication latency and poor tracking accuracy limited its use in control tasks. Replacement with an actuator that offers lower latency, faster command execution as well as better closed-loop position control is essential to test control strategies under realistic dynamic loads.

Testing

The current tests should be expanded to include long-duration fatigue tests to evaluate the reliability and repeatability of 3D-printed springs under cyclic loading. Such testing would provide insight into degradation mechanisms such as creep, plastic deformation, and delamination, which are particularly relevant for FDM-printed materials.

Spring Material and Design

In addition, future investigations should systematically explore alternative materials and printing parameters or methods as well as higher target stiffness for the spring. Adjustments in print orientation, infill patterns, or the use of composite or reinforced filaments could enhance the energy recovery capabilities and linear stiffness of the springs. These changes may also reduce hysteresis losses and improve dynamic performance. Furthermore, changes to the existing design to eliminate mechanical interference in the cam connection are vital for integration in robotic joints. However, alternative design strategies distinct from the tool from [1] should also be investigated.

Together, these improvements offer a clear path forward to increase the accuracy, robustness, and versatility of the developed testbed and to low-cost application of FDM-manufactured elastic components in compliant robotic systems.

A Calculations

A.1 Driveshaft Twist

To Approximate angular elastic deformation of the driven shaft under a given load, we calculate the angle of twist as

$$\theta = \frac{Tl}{GJ}, \tag{A.1}$$

Where where T is the applied Torque, l is the drive shaft lengthen for the considered section, G is the shear modulus of the shaft material and J is the polar moment of inertia.

For the drive shaft used in this testbed the given data is

- **Drive shaft diameter:** $d = 8$ mm,
- **Shear Modulus (Standard Steel):** $G = 75$ GPa.

The polar moment of inertia assuming a circular cross section is approximated as

$$J = \frac{\pi}{32} d^4 = 4.021 \times 10^{-10} \mathrm{m}^4. \tag{A.2}$$

A.1.1 Twist Estimation of Measured Sections

For the short section of $l = 30$ mm from the spring interface to the encoder measurement point in the unique testbed configuration from Section 5.1.4 the angle of twist is calculated. For the test performed in this section the drive shaft was under a load of $T = 3$ Nm. Substituting the values for Equation A.1 the expected angular twist is

$$\theta = \frac{3 \times 0.03}{75 \times 10^9 \times 4.021 \times 10^{-10}}$$

$$\theta = 0.00298 \text{ rad} = 0.171^\circ$$

Furthermore, the total drive shaft twist for the maximum actuation torque of $T = 10$ Nm and the effective drive shaft length from spring interface to FTS interface of $l = 150$ mm is calculated. Substituting the values for Equation A.1 the expected total angular twist is

$$\theta = \frac{10 \times 0.15}{75 \times 10^9 \times 4.021 \times 10^{-10}}$$

$$\theta = 0.0497 \text{ rad} = 2.85^\circ.$$

This represents an approximation, since the drive shaft geometry differs from the approximation. The actual polar moment will be smaller and the resulting twist angles will be higher.

A.1.2 Drive Shaft Stiffness Estimation

To estimate the stiffness of the drive shaft, the following calculation is performed for the effective drive shaft length of $l = 150$ mm. Linear stiffness is given by:

$$k_\theta = \frac{T}{\theta} \tag{A.3}$$

and the angle of twist is given by:

$$\theta = \frac{Tl}{GJ} \tag{A.4}$$

By rearranging the twist equation, we get:

$$k = \frac{T}{\theta} = \frac{GJ}{l} \tag{A.5}$$

$$k = \frac{75 \times 10^9 \times 4.021 \times 10^{-10}}{0.15} = 201.05 \ \frac{\text{Nm}}{\text{rad}}$$

This result provides an estimate for the rotational stiffness of the total drive shaft section from the spring interface to the force-torque sensor. Note that this calculation assumes a solid circular profile. Since the actual shaft geometry includes a flat side, the true stiffness will be slightly softer.

A.2 Estimation of Angular Measurement Uncertainty

To quantify the angular uncertainty used in the error propagation of the stiffness calculation, a static test was analyzed. The data originates from the gravity compensation test in Section 5.4. In this test, the actuator was commanded to hold fixed positions from 0° to -90° in -10° increments, while connected to a spring and a weight-loaded link. The system was allowed to reach a static equilibrium at each commanded position to only capture noise.

The measured angular values $\theta(t)$ were evaluated at each static position using short time intervals. For every interval, the mean $\bar{\theta}_i$ were calculated. Additionally, the maximum deviation from the mean was recorded per position:

$$\delta\theta_i = \max\left|\theta_i(t) - \bar{\theta}_i\right| \tag{A.6}$$

This allowed quantification of an estimate of the random noise. The final angular uncertainty used in stiffness error propagation is based on the mean of all observed maximum deviations

$$\delta\theta = \frac{1}{n}\sum_{i=1}^{n}\delta\theta_i. \tag{A.7}$$

The resulting value was

$$\delta\theta = 0.017177^\circ = 2.998 \cdot 10^{-4}\ \text{rad}.$$

B Data

B.1 Component Specifications

Actuator

Parameter	Unit	Value
Gear ratio	-	1:6
Input Voltage	V	48
Rated Speed	RPM	190
Rated Torque	N.m	10
Rated Power	W	200
Rated Current	A	5.2
Peak Torque	N.m	20
Peak Current	A	10.5
Back-Drive Torque	N.m	0.4
Backlash	Arcmin | °	10 | 0.167
Axial Payload	N	985
Inertia	$Kg \cdot cm^2$	20
Encoder res in/out	bit	16 / 14
Communication	-	CAN:1M
Torque(Current) Loop	Hz	15k
Speed Loop	Hz	5k
Position Loop	Hz	1k

Table B.1: Motor: MyActuator X8-20 (RMD-X8-Pro-H 1:6) specifications [11].

Force Torque Sensor

Parameter	Unit	Value
Operating voltage	VDC	12-24
Nominal torque range (M_{XYZN})	Nm	15
Limit torque (M_{XYZL})	Nm	20
Resolution	Nm	0.015
Accuracy	%	1
Sample rate	Hz	1000
Interfaces		EtherCAT

Table B.2: Force/Torque Sensor AFT200-D80-EC specifications [13].

Absolute Encoder

Parameter	Unit	Value
Power Supply (VDD)	3.8 - 5.5 V	
Start-up Time	ms	200
Current Consumption	mA	16
Resolution	bit	14
Accuracy	degrees	0.2
Type	-	single-turn
Absolute Position Update Rate	μs	100
Rotational Speed	RPM	4,000
shaft size	mm	2-8
Protocol	-	RS485
com-speed	bps	115,200
time before encoder responds	μs	10.8

Table B.3: Absolute rotary encoder AMT212F-V specifications [14]

Drive Shaft and Flange

Parameter	Unit	Value
Flat sided drive Shaft Specifications		
Material	-	Nickel-plated steel
length	mm	≈ 170
Dimensions	mm	8/7
Flange Specifications		
Material	-	hardened steel
Inner Diameter	mm	8
Total Height	mm	13
Outer Cylinder Diameter	mm	16
Cylinder Height	mm	10
Base Diameter	mm	32
Base Height	mm	3
Mounting Hole Spacing	mm	24
Clamping Screws	-	$2 \times$ M4

Table B.4: Drive shaft [15] and flange [16] specifications.

Raspberry Pi 5

Feature	Specification
Processor	Broadcom BCM2712 2.4GHz quad-core 64-bit Arm Cortex-A76 CPU, with cryptography extensions 512KB L2 cache per core, 2MB shared L3 cache
Memory	LPDDR4X-4267 SDRAM (2GB, 4GB, 8GB, 16GB)
Wi-Fi	Dual-band 802.11ac Wi-Fi®
Ethernet	Gigabit Ethernet, PoE+ support (via separate HAT)
Power	5V/5A DC via USB-C, Power Delivery supported
GPIO	40-pin Raspberry Pi standard header
RTC	Real-time clock, powered from external battery

Table B.5: Raspberry Pi 5 specifications [20]

COM-HAT

Parameter	Unit	Value
Expanded Interface	-	2-Ch RS485 + 1-Ch CAN
Communication Bus	-	2 x SPI
CAN Baud Rate	Mbps	≤ 1
RS485 Baud Rate	bps	300 - 921600
Operating Voltage	V	3.3
Dimensions	mm	65×56.5

Table B.6: Waveshares CAN/RS HAT specifications [21]

B.2 Timing Measurements

B.2.1 Actuator

duration (μs)
782.947
625.298
632.724
620.131
591.705
621.984
603.631
627.613
581.761
676.595
651.910
660.206
644.169
607.576
649.131
619.780
677.039
617.446
639.428
602.150
625.095
699.687
621.816
643.187
683.002
Mean 640,240

Table B.7: Timing data baseline test

Table B.8: Response duration reading motor angle (0x92) command

Batch	Mean (μs)	Std Dev. (μs)	t >1200 μs	t_{max} (μs)
1	999.774	17.201	2	1306.084
2	1001.353	42.863	5	1998.988
3	998.647	15.263	1	1271.158
4	998.718	5.694	0	1081.078
5	998.279	7.629	0	1100.691
6	1006.034	161.425	6	5973.166
7	999.818	9.712	0	1195.822
8	998.206	8.589	0	1080.042
9	997.610	8.376	0	1097.858
10	998.516	6.268	0	1105.505
11	1001.463	41.375	4	2102.990
12	999.766	11.099	1	1235.360
13	998.102	7.669	0	1079.171
14	997.913	8.715	0	1083.931
15	998.117	7.081	0	1085.375
16	1002.507	76.633	4	2814.783
17	998.909	18.991	1	1481.235
18	997.913	6.907	0	1083.209
19	997.915	7.402	0	1080.764
20	998.513	5.908	0	1082.524
21	1001.712	43.459	4	2087.248
22	999.310	19.300	2	1510.607
23	998.527	4.935	0	1024.504
24	997.868	6.001	0	1024.263
25	997.350	8.447	0	1084.116
26	1002.104	45.128	6	2104.545
27	999.093	10.649	1	1209.193
28	999.533	4.361	0	1098.024
29	998.407	6.620	0	1094.208
30	998.308	7.170	0	1085.857

Table B.9: Response duration absolut position (0xA4) command

Batch	Mean (μs)	Std Dev. (μs)	t >1600 μs	t_{max} (μs)
1	1404.960	98.226	4	4404.418
2	1400.865	17.932	0	1578.032
3	1400.257	15.971	0	1593.069
4	1400.263	17.053	1	1621.588
5	1402.457	37.319	4	2145.661
6	1400.254	15.821	0	1564.957
7	1399.673	14.706	0	1528.400
8	1401.657	25.775	1	1921.394
9	1400.257	15.723	0	1553.771
10	1399.852	13.137	0	1458.935
11	1400.242	15.687	0	1569.846
12	1405.461	109.523	4	4646.628
13	1400.279	15.985	0	1587.105
14	1400.055	14.439	0	1529.807
15	1400.855	17.463	0	1575.031
16	1400.236	17.227	1	1616.125
17	1399.882	13.999	0	1520.899
18	1400.033	14.242	0	1529.326
19	1402.663	39.532	4	2210.143
20	1400.020	15.840	0	1564.512
21	1399.898	14.799	0	1559.771
22	1400.482	17.572	1	1606.087
23	1406.657	132.548	4	5494.430
24	1400.233	15.725	0	1560.456
25	1399.879	14.506	0	1533.621
26	1402.454	33.311	3	2103.250
27	1401.261	23.890	1	1911.892
28	1400.246	15.994	0	1539.510
29	1399.854	14.741	0	1578.845
30	1402.859	39.795	2	2238.051

Table B.10: Response duration postion tracking (0xA5) command

Batch	Mean (μs)	Std Dev. (μs)	t >1600 μs	t_{max} (μs)
1	1407.406	77.642	3	3770.707
2	1406.024	60.162	3	3222.002
3	1406.049	36.284	3	2152.890
4	1405.358	20.011	3	1746.080
5	1403.588	11.272	0	1455.236
6	1403.872	11.684	0	1474.626
7	1406.857	63.818	3	2962.733
8	1404.927	17.005	0	1587.203
9	1404.001	15.312	1	1603.001
10	1407.056	58.935	3	2967.474
11	1404.541	20.017	3	1721.856
12	1403.184	10.709	0	1457.514
13	1403.668	12.629	0	1580.296
14	1409.323	122.338	3	5190.549
15	1404.131	19.584	1	1711.596
16	1403.400	12.683	0	1594.944
17	1403.457	12.682	0	1564.016
18	1407.198	36.737	4	2141.351
19	1403.521	14.986	0	1576.146
20	1403.459	13.316	0	1573.146
21	1405.794	36.631	1	2424.805
22	1404.921	17.117	1	1607.277
23	1402.677	9.953	0	1476.365
24	1402.854	11.742	0	1580.813
25	1406.330	24.673	1	1604.165
26	1403.382	11.182	0	1517.292
27	1403.278	13.145	0	1579.480
28	1406.186	61.318	2	3249.166
29	1403.984	12.552	0	1558.145
30	1403.594	11.069	0	1459.142

Table B.11: Response duration commands (0x92) + (0xA4)

Batch	Mean (μs)	Std Dev. (μs)	t >4000 μs	t_{max} (μs)
1	2792.526	101.794	2	5299.157
2	2791.091	43.064	0	3612.070
3	2787.678	28.263	0	2956.186
4	2793.598	130.063	1	6714.463
5	2788.031	27.785	0	2925.612
6	2789.233	42.621	0	3664.941
7	2784.644	27.847	0	2846.684
8	2790.474	115.627	1	6227.120
9	2791.193	138.354	2	5740.387
10	2786.370	27.947	0	3025.781
11	2788.045	33.755	0	3267.155
12	2785.902	29.726	0	2937.056
13	2786.831	47.627	0	3503.492
14	2784.641	29.520	0	2934.519
15	2788.307	50.469	0	3671.329
16	2786.175	28.396	0	2922.574
17	2789.486	65.971	1	4365.639
18	2788.839	137.438	1	6990.151
19	2786.831	30.814	0	3124.763
20	2786.825	38.175	0	3427.657
21	2788.237	31.719	0	3222.506
22	2788.262	38.431	0	3462.492
23	2785.569	28.898	0	2936.518
24	2790.640	48.887	0	3454.899
25	2787.098	31.582	0	3095.114
26	2787.826	42.228	0	3273.284
27	2792.569	121.535	1	6316.675
28	2787.254	34.743	0	3297.210
29	2788.747	46.481	0	3606.365
30	2785.573	31.849	0	3135.226

B.2.2 FTS

Table B.12: Response duration read torque measurement command

Batch	Mean (μs)	Std Dev (μs)	t >95 μs	t_{max} (μs)
1	79.636	0.921	1	106.226
2	79.657	1.057	1	96.429
3	79.610	0.671	0	91.744
4	79.585	0.769	0	87.651
5	79.598	0.677	1	95.670
6	79.563	0.563	0	85.392
7	79.497	0.615	0	87.336
8	79.469	0.674	0	87.021
9	79.463	0.493	0	84.984
10	79.482	0.497	0	86.429
11	81.247	1.577	0	94.392
12	82.581	0.972	1	95.503
13	83.358	6.060	28	203.933
14	80.909	1.089	2	98.577
15	82.300	1.282	0	93.892
16	82.304	1.029	0	94.466
17	82.281	0.996	1	95.429
18	82.182	0.917	0	91.243
19	82.184	0.844	0	92.170
20	82.139	0.793	0	94.244
21	82.205	0.972	0	92.484
22	82.161	0.967	0	93.873
23	82.068	0.868	0	90.670
24	82.108	1.120	1	95.170
25	82.103	0.602	0	90.651
26	82.401	2.849	13	118.337
27	81.879	1.252	0	93.799
28	82.027	1.048	1	98.263
29	82.258	0.947	1	96.522
30	82.161	0.791	0	91.114

B.2.3 Absolute encoder

Table B.13: Response duration reading position command

Batch	Mean (μs)	Std Dev (μs)	t >900 μs	t_{max} (μs)
1	841.712	10.435	5	993.447
2	839.851	6.574	0	890.596
3	840.418	7.042	1	904.873
4	839.647	5.722	0	862.429
5	839.435	5.821	0	866.373
6	840.442	7.222	1	911.614
7	839.193	6.211	1	918.818
8	839.436	5.913	1	907.929
9	839.276	5.781	0	898.966
10	840.029	5.764	1	901.114
11	839.539	5.098	0	860.280
12	840.076	7.910	3	951.947
13	839.578	6.128	0	895.281
14	840.545	5.947	0	860.836
15	839.282	5.650	0	870.725
16	839.879	5.566	1	900.891
17	839.662	5.326	0	868.596
18	840.377	6.264	0	874.688
19	840.826	7.035	1	902.632
20	841.057	5.482	0	881.095
21	839.774	5.370	0	866.706
22	839.053	5.379	0	861.725
23	839.290	5.060	0	853.725
24	840.095	8.615	4	952.577
25	839.422	6.511	1	926.559
26	839.965	4.909	0	858.817
27	839.124	5.200	0	854.891
28	840.554	6.066	0	864.725
29	840.050	6.816	1	905.096
30	839.453	6.722	1	907.947

B.3 Torsion Spring Design Parameter

Parameter	**Unit**	**Value**		
Profile-ID	-	09	10	12
Material	-	PETG	PLA	PC-ABS
Outer Radius	mm	50.00	50.00	50.00
Spring Thickness	mm	10.00	10.00	10.00
Spring Stiffness	Nm/rad	20.00	20.00	20.00
Design Stress	MPa	25.52	42.39	31.12
Young's Modulus	GPa	1.47	1.15	1.77
Root Radius	mm	45.00	45.00	45.00
Contact Radius	mm	20.00	20.00	20.00
Number of Flexures	-	8.00	8.00	8.00
Number of Pins	-	1.00	1.00	1.00
Pin Radius	mm	2.00	2.50	2.00
Desired Deflection	deg	12.00	21.57	12.00
Run Time	s	180.00	180.00	180.00
Step size	mm	0.150	0.150	0.150
Number of Control Points	-	3.000	3.000	3.000
Tip-to-cam Gap	mm	0.100	0.100	0.200
Minimum Tip Radius	mm	1.000	2.500	2.000
Allowable Deflection	deg	12.00	21.57	12.00
Flexure Closeness	mm	15.13	15.12	15.17

Table B.14: Design tool parameters of discussed springs

Bibliography

[1] Z. Bons, G. C. Thomas, L. Mooney, and E. J. Rouse, "An energy-dense two-part torsion spring architecture and design tool," in *IEEE/ASME Transactions on Mechatronics*, 2023.

[2] 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, 1995, 399–406 vol.1.

[3] B. Vanderborght *et al.*, "Variable impedance actuators: A review," in *Robotics and Autonomous Systems*, vol. 61, 2013, pp. 1601–1614.

[4] Westwood Robotics, *Bruce – The Biped Robot*, Accessed: 2025-05-03, 2023. [Online]. Available: `https://www.westwoodrobotics.io/bruce/`.

[5] O. S. Al-Dahiree, R. A. R. Ghazilla, M. O. Tokhi, H. J. Yap, and M. Gul, "Design and characterization of a low-cost and efficient torsional spring for es-rsea," *Sensors*, vol. 23, no. 7, 2023, ISSN: 1424-8220. [Online]. Available: `https://www.mdpi.com/1424-8220/23/7/3705`.

[6] S. Arumugom, S. Muthuraman, and V. Ponselvan, "Modeling and application of series elastic actuators for force control multi legged robots," *CoRR*, vol. abs/0912.3956, 2009. arXiv: `0912.3956`. [Online]. Available: `http://arxiv.org/abs/0912.3956`.

[7] E. Garcia, J. C. Arevalo, G. Munoz, and P. Gonzalez-de-Santos, "Combining series elastic actuation and magneto-rheological damping for the control of agile locomotion," in *International Conference on Robotics and Automation (ICRA)*, 2011, pp. 5147–5152.

[8] S. Wolf *et al.*, "Variable stiffness actuators: Review on design and components," in *IEEE/ASME Transactions on Mechatronics*, vol. 21, 2016, pp. 2418–2430.

[9] D. Robinson, J. Pratt, D. Paluska, and G. Pratt, "Series elastic actuator development for a biomimetic walking robot," in *1999 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (Cat. No.99TH8399)*, 1999, pp. 561–568.

[10] A. Sutrisno and D. J. Braun, *High-energy-density 3d-printed composite springs for lightweight and energy-efficient compliant robots*, 2022. arXiv: `2211.09245 [cs.RO]`. [Online]. Available: `https://arxiv.org/abs/2211.09245`.

[11] MyActuator, *X8-20 actuator specifications*, Accessed: Feb-2025, 2025. [Online]. Available: `https://www.myactuator.com/x8-20-details`.

[12] A. Manual, *Aft200-d80-ec e-manual*, Accessed: Feb-2025, 2025. [Online]. Available: `https://emanual.oopy.io/aft200-d80-ec`.

[13] A. specs, *Aft200-d80-ec specs*, Accessed: Feb-2025, 2025. [Online]. Available: https://www.aidinrobotics.co.kr/en/smart-6-axis-f-t-sensor.

[14] S. Devices, *Amt212f-v absolute modular rotary encoder - technical specifications*, Accessed: Feb-2025, 2025. [Online]. Available: https://www.sameskydevices.com/product/motion-and-control/rotary-encoders/absolute/modular/amt212f-v#technical.

[15] Beschlagtechnik24, *Profilstange l 1000 mm - drehstangenschlösser*, Accessed: Feb-2025, 2025. [Online]. Available: https://www.beschlagtechnik24.de/Schliesssysteme/Mechanische-Schloesser/Drehstangenschloesser/Profilstange--L--1000-mm.html.

[16] A. Germany, *Flanschkupplung 8mm - sourcing map*, Accessed: Feb-2025, 2025. [Online]. Available: https://www.amazon.de/sourcing-map-Flansch-Wellenkupplung-Motorverbinder/dp/B086M7KB8H.

[17] 3DJake, *Threaded inserts - 50 piece set*, Accessed: Feb-2025, 2025. [Online]. Available: https://www.3djake.com/3djake/threaded-inserts-50-piece-set?sai=9420.

[18] A. Germany, *Honlena kugellager 608zz - shielded rillenkugellager*, Accessed: Feb-2025, 2025. [Online]. Available: https://www.kugellager-express.de/miniature-deep-groove-ball-bearing-608-zz-8x22x7-mm.

[19] item24, *Aluminium profiles - profile technology*, Accessed: Feb-2025, 2025. [Online]. Available: https://www.item24.com/en-us/profile-technology/aluminium-profiles.

[20] R. P. Foundation, *Raspberry pi 5 - product specifications*, Accessed: Feb-2025, 2025. [Online]. Available: https://www.raspberrypi.com/products/raspberry-pi-5/.

[21] Waveshare, *Rs485 can hat (b) - waveshare wiki*, Accessed: Feb-2025, 2025. [Online]. Available: https://www.waveshare.com/wiki/RS485_CAN_HAT_(B).

[22] Raise3D, *Raise3D Premium PETG Technical Data Sheet*, Accessed: Feb-2025, 2019. [Online]. Available: https://s2.raise3d.com/public/media/2019/07/Raise3d_Premium_PETG_TDS_V3.pdf.

[23] Raise3D Technologies Inc., *Raise3D Pro3 Plus 3D Printer*, https://www.raise3d.com/de/products/raise3d-pro3-plus-3d-printer/, Accessed: 2025-05-06, 2025.

[24] P. Pably, *Sea-testbed software: Control and evaluation framework for series elastic actuator experiments*, https://github.com/ppably/SEA-Testbed, Accessed: 2025-05-01, 2025.

[25] Polymaker, *Polymaker PC-ABS – Technical Data Sheet*, Accessed: Feb-2025, 2021. [Online]. Available: https://3d.nice-cdn.com/upload/file/Polymaker_PC_ABS_TDS_V5.1.pdf.

[26] MISUMI, *Shaft collars - standard type*, Accessed: 2025-05-01, 2025. [Online]. Available: https://uk.misumi-ec.com/vona2/mech/M0100000000/M0103000000/.

[27] MISUMI, *Linear shafts - standard round type*, Accessed: 2025-05-01, 2025. [Online]. Available: `https://uk.misumi-ec.com/vona2/mech/M0100000000/M0101000000/`.

[28] McMaster-Carr, *Mcmaster-carr supply company online catalog*, Accessed: 2025-05-01, 2025. [Online]. Available: `https://www.mcmaster.com/`.

Eidesstattliche Erklärung

Hiermit erkläre ich, dass die vorliegende Arbeit ohne unzulässige Hilfe Dritter und ohne Benutzung anderer als der angegebenen Hilfsmittel angefertigt wurde. Die aus anderen Quellen oder indirekt übernommenen Daten und Konzepte sind unter Angabe der Quelle gekennzeichnet. Die Arbeit wurde bisher weder im In- noch im Ausland in gleicher oder in ähnlicher Form in anderen Prüfungsverfahren vorgelegt.

Wien, Mai 2025

Patrick Pably