Design, Modeling, and Control Strategies for Soft Robots

Brandon Jonathan Caasenbrood

The work described in this thesis was carried out at the Eindhoven University of Technology.

A catalogue record is available from the Eindhoven University of Technology Library.
ISBN: 123-45-678-9012-3

Typeset by the author using the pdf LaTeX documentation system.
Cover design: Name of cover designer
Reproduction: Name of printer || www.printerswebsite.nl

Design, Modeling, and Control Strategies for Soft Robots

Proefschrift

ter verkrijging van de graad van doctor aan de Technische Universiteit Eindhoven, op gezag van de rector magnificus prof.dr.ir. F.P.T. Baaijens, voor een commissie aangewezen door het College voor Promoties, in het openbaar te verdedigen op maandag ?? ?? 2023 om 16:00 uur

door

Brandon Jonathan Caasenbrood

geboren te Roermond

Dit proefschrift is goedgekeurd door de promotoren en de samenstelling van de promotiecommissie is als volgt:

voorzitter:	
promotor:	prof. dr. H. Nijmijer
co-promotor:	dr. A.Y. Pogromsky
leden:	prof. dr. J. den Toonder
	prof. dr. G. Krijnen (Twente University)
	dr. ir. J.T.B. Overvelde (AMOLF)
	prof. dr. A. Ollero (University Seville)

Het onderzoek dat in dit proefschrift wordt beschreven is uitgevoerd in overeenstemming met de TU/e Gedragscode Wetenschapsbeoefening.

Abstract

Design, Modeling, and Control Strategies for Soft Robots

In the past two decades, the field of soft robotics has sparked significant interest among many scientific disciplines. Contrary to rigid robots, soft robots explore soft materials that significantly enhance the robot's dexterity, enable a rich family of motion primitives, and enhance environmental robustness regarding contact and impact. Since its inception, soft robotics has exemplified its potential in diverse areas such as safe manipulation, adaptive grasping, exploration under environmental uncertainty, rehabilitation, and the bio-mimicry of many animals. By exploring the uncharted versatile nature of soft materials, soft robotics places the first steppingstones towards achieving biological performance in modern-day's robotics. This thesis aims to further the advances in soft robotics by addressing some of the open multi-disciplinary challenges within this young field of research.

Although soft materials harbor many advantages akin to biology, which are difficult to achieve for rigid robotics, it also roots many fundamental problems. First is the issue of soft robotic design. Traditional robotic design emphasizes high structural rigidity and weight minimization – a well-established practice in engineering. On the other hand, soft robotic design relishes minimal structural rigidity for motion, leading to complex, highly nonlinear relations between the input and output. Besides, distributed soft actuation, imparted by gravitational and inertial forces acting on the continuum elastic body, introduce joint mobilities that are in many cases uncontrollable nor aligned with the control objective, *e.g.*, precise grasping and manipulation. Since describing the underlying continuum mechanics and applying such mathematical theory to systematic design is challenging, a large number of soft robotic systems are still developed *ad hoc*.

Second, a direct duality of the previous challenge is dealing with the innate infinite-dimensionality from a control perspective – particularly with model-based feedback in mind. The transition from rigid to flexible has introduced a new control paradigm: the trade-off between precision and speed in a numerical setting. Not only is control theory for soft robotics in stages of inception, but deriving accurate and numerically efficient model-based controllers is challenging due to large nonlinear deformations of the soft robotic continuum.

In light of these challenges, this thesis proposes a set of systematic tools with theoretical and experimental applications for (*i*) the structural design and fabrication of continuum-deformable soft actuators optimized for user-defined joint motion, (*ii*) the development of efficient dynamic models of soft continuum manipulators, and (*iii*) applying such mathematical models to model-based controllers for a subclass of (pneumatic) soft continuum manipulators and soft grippers.

The first part of the thesis addresses the design problem by proposing novel computer-automated design algorithms for developing efficient soft actuators. These algorithms account for the underlying continuum mechanics described by a set of partial differential equations, which respect the aforementioned nonlinearities between the input and output motion. Tailoring a user-defined objective to a desired motion and control reachability, an implicit representation of the optimal soft material distribution can be found within a fixed design space. Several generative designs for a diverse subset of soft actuation morphologies are produced including, but not limited to, soft rotational actuators, soft artificial muscles, and soft grippers. In what follows, an optimal design for a soft robotic manipulator with an adaptive gripper is synthesized; and through Additive Manufacturing (AM) of printable flexible material, the sim-to-real boundary is passed. The proposed approach does not only accelerate design convergence but also builds upon the vast library of soft robot morphologies currently unexplored in literature.

The second part of the thesis addresses the question of modeling for control applicable to a class of soft robotic systems – most notably soft continuum manipulators. The thesis proposes a reduced-order modeling strategy for soft robotics, whose dynamics are derived through the differential geometric theory on spatial beams. Besides discussing earlier modeling strategies, the thesis also proposes a new strain-based parametrization approach that ensures the structural information and the underlying continuum mechanics are preserved when synthesizing the reduced-order beam models – a possible solution to the aforementioned control paradigm of precision vs. speed. To enhance numerical performance further, spatio-temporal integration schemes are also proposed that exploit the geometric structure of such soft beam models, resulting in real-time simulation with sufficient numerical precision purposefully tailored for control.

The third part of the thesis treats the development of model-based controllers that can be employed in various control scenarios akin to control for traditional rigid robotics, *e.g.*, inverse kinematics and motion planning, set-point stabilization, trajectory tracking, and multi-point grasping of objects. The stabilizing controller is rooted in an energy-based formulism, providing robustness even when faced with material uncertainties. The controller's effectiveness is demonstrated both in simulation and experiments for various soft robotic systems that share a resemblance to biology, *e.g.*, the elephant's trunk or the tentacle of an octopus.

The main contribution of the thesis is a collection of multi-disciplinary tools compressed into one general framework for the design, modeling, and control of a class of soft robots, ranging from the theoretical to the experimental domain.

Keywords: Soft Robots, Hyper-redundant Robots, Design Optimization, Continuum Mechanics, Reduced-order Modeling, Model-based Control, 3D Printing.

Samenvatting

In de afgelopen twee decennia heeft het veld van de zachte robotica significant interesse opgewekt binnen verschillende wetenschappelijke disciplines. In tegenstelling tot rigide robots verkennen zachte robots zachte materialen die de behendigheid van de robot aanzienlijk verbeteren, een rijke collectie van bewegingsprimitieven mogelijk maken en de omgevingsbestendigheid ten aanzien van contact en impact vergroten. Sinds de oprichting heeft de zachte robotica haar potentieel aangetoond in diverse gebieden zoals veilige manipulatie, adaptief grijpen, verkenning onder omgevingsonzekerheid, revalidatie en de biomimetica van vele dieren. Door de veelzijdige aard van zachte materialen te verkennen, legt de zachte robotica de eerste stappen naar het bereiken van biologische prestaties in de moderne robotica. Deze scriptie heeft als doel de vooruitgang in de zachte robotica verder te bevorderen door enkele van de open multidisciplinaire uitdagingen binnen dit jonge onderzoeksgebied aan te pakken.

Hoewel zachte materialen, zoals systemen in de biologie, veel voordelen hebben, die soms moeilijk te bereiken zijn voor rigide robotica, brengt het ook veel fundamentele problemen met zich mee. Het eerste probleem is het ontwerp van zachte robots. Traditioneel robotica-ontwerp legt de nadruk op hoge structurele stijfheid en gewichtsminimalisatie - een goed doordachte discipline in de engineering. Aan de andere kant houdt het ontwerp van zachte robots van minimale structurele stijfheid voor beweging, wat leidt tot complexe, zeer niet-lineaire relaties tussen input en output. Bovendien leiden gedistribueerde zachte activering, toegepast door zwaartekracht- en traagheidskrachten die op het continu elastische lichaam werken, tot gewrichtsmobiliteiten die in veel gevallen niet te controleren zijn of niet zijn afgestemd op de controle-doelstelling, zoals nauwkeurig grijpen en manipulatie. Omdat het beschrijven van de onderliggende continue mechanica en het toepassen van dergelijke wiskundige theorie op systematisch ontwerp uitdagend is, worden nog steeds een groot aantal zachte robotsystemen *ad hoc* ontwikkeld.

Ten tweede vormt de directe dualiteit van de vorige uitdaging het omgaan met de intrinsieke oneindige-dimensionaliteit vanuit een controleperspectief - met name met modelgebaseerde feedback in gedachten. De overgang van rigide naar flexibel heeft een nieuw controleparadigma geïntroduceerd: de afweging tussen precisie

en snelheid in een numerieke omgeving. Niet alleen bevindt de controletheorie voor zachte robotica zich in de beginfase, maar het afleiden van nauwkeurige en numeriek efficiënte modelgebaseerde controllers is uitdagend vanwege de grote niet-lineaire vervormingen van de zachte roboticacontinuüm.

Gezien deze uitdagingen stelt deze scriptie een reeks systematische tools voor met theoretische en experimentele toepassingen voor (*i*) het structurele ontwerp en de fabricage van continuüm-deformeerbare zachte actuators geoptimaliseerd voor door de gebruiker gedefinieerde gezamenlijke beweging, (*ii*) de ontwikkeling van efficiënte dynamische modellen voor zachte continuüm manipulatoren, en (*iii*) het toepassen van wiskundige modellen op modelgebaseerde controllers voor een subklasse van (pneumatische) zachte continuüm manipulatoren en zachte grijpers.

Het eerste deel van deze scriptie richt zich op het ontwerpprobleem door het voorstellen van nieuwe computer-geautomatiseerde ontwerpalgoritmen voor de ontwikkeling van efficiënte zachte actuators. Deze algoritmen houden rekening met de onderliggende continue mechanica die wordt beschreven door een set partiële differentiaalvergelijkingen, die de eerder genoemde niet-lineariteiten tussen de input en output beweging respecteren. Door een door de gebruiker gedefinieerd doel aan te passen aan een gewenste beweging en controlebereik, kan een impliciete representatie van de optimale zachte materiaalverdeling worden gevonden binnen een vast ontwerpruimte. Verschillende generatieve ontwerpen voor een diverse subset van zachte actuator-morfologieën worden geproduceerd, waaronder, maar niet beperkt tot, zachte rotatie-actuators, zachte kunstmatige spieren en zachte grijpers. Vervolgens wordt een optimaal ontwerp voor een zachte robotmanipulator met een adaptieve grijper gesynthetiseerd. Door middel van Additive Manufacturing (AM) van printbaar flexibel materiaal wordt de grens tussen simulatie en realiteit overschreden. De voorgestelde aanpak versnelt niet alleen het convergeren van het ontwerp, maar bouwt ook voort op de enorme bibliotheek van zachte robot-morfologieën die momenteel onontdekt zijn in de literatuur.

Het tweede deel van de scriptie richt zich op de modellering voor controle die van toepassing is op een klasse van zachte robotica-systemen - met name zachte continuüm manipulatoren. De scriptie stelt een modelleerstrategie voor met gereduceerde orde voor zachte robotica, waarvan de dynamica worden afgeleid door middel van differentiële geometrische theorie op ruimtelijke balken. Naast het bespreken van eerdere modelleerstrategieën stelt de scriptie ook een nieuwe spanninggebaseerde parameterisatiebenadering voor die ervoor zorgt dat de structurele informatie en de onderliggende continue mechanica behouden blijven bij het synthetiseren van de gereduceerde balkmodellen - een mogelijke oplossing voor het eerder genoemde controleparadigma van precisie versus snelheid. Om de numerieke prestaties verder te verbeteren, worden ook spatio-temporele integratieschema's

voorgesteld die de geometrische structuur van dergelijke zachte balkmodellen benutten, wat resulteert in real-time simulatie met voldoende numerieke precisie die specifiek is afgestemd op controle.

Het derde deel van de scriptie behandelt de ontwikkeling van op modellen gebaseerde controllers die kunnen worden gebruikt in verschillende controle scenario's vergelijkbaar met de controle voor traditionele rigide robotica, bijvoorbeeld inverse kinematica en bewegingsplanning, set-point stabilisatie, traject volgen en multi-point grijpen van objecten. De stabiliserende controller is geworteld in een op energie gebaseerde formulering, die robuustheid biedt, zelfs wanneer er sprake is van materiaalonzekerheden. De effectiviteit van de controller wordt zowel in simulatie als in experimenten aangetoond voor verschillende zachte robotica-systemen die een gelijkenis vertonen met de biologie, bijvoorbeeld de slurf van een olifant of de tentakel van een octopus.

De belangrijkste bijdrage van de scriptie is een verzameling multidisciplinaire tools gecomprimeerd in één algemeen framework voor het ontwerp, de modellering en de controle van een klasse van zachte robots, variërend van het theoretische tot het experimentele domein.

Trefwoorden: Trefwoord 1, Trefwoord 2, Trefwoord 3, ...

Societal summary

Soft robotics is an emerging field that focuses on the design, development, and implementation of robots made from soft, flexible materials. Unlike traditional rigid robots, soft robots are able to mimic the movements and behaviors of living organisms, making them well-suited for a wide range of applications in fields such as healthcare, manufacturing, and exploration.

One of the main benefits of soft robotics is their ability to interact with humans and delicate objects in a safe and gentle way. Soft robots can be designed to be compliant and adaptable, allowing them to work in close proximity to people without causing harm. They can also be used in healthcare applications, such as wearable devices and prosthetics, where their flexibility and ability to conform to the human body can provide greater comfort and functionality than traditional rigid devices. Another advantage of soft robotics is their ability to operate in unstructured and unpredictable environments, such as disaster zones or outer space. Soft robots can change shape and adapt to their surroundings, making them more versatile and capable of performing a wide range of tasks. For example, soft robots can be designed to crawl through small spaces, squeeze through tight openings, and manipulate objects with greater precision than rigid robots.

However, there are also challenges associated with the development and implementation of soft robotics. One major challenge is the complexity of designing and controlling soft robots, which often require sophisticated algorithms and advanced materials. Additionally, there is a need for more research into the long-term durability and reliability of soft robots, particularly in harsh or extreme environments.

Despite these challenges, the potential benefits of soft robotics are significant, and the field continues to grow and evolve. As more researchers and engineers develop new materials, technologies, and applications for soft robots, we can expect to see these versatile and adaptable machines play an increasingly important role in our lives and society.

Nomenclature

Vector and matrix notations

x	Scalar notation
$\boldsymbol{x}$	Vector notation
$\boldsymbol{X}$	Matrix notation
$\boldsymbol{\mathcal{X}}$	Tensor notation
$\mathcal{Q}$	Manifold
$T_{\mathcal{Q}}$	Tangent space of $\mathcal{Q}$

Set notations

$\emptyset$	Empty set
$\mathbb{R}$	Set of real numbers
$\mathbb{R}^n$	n-dimensional Euclidean space
$\mathbb{R}_{>0}$	Strictly positive reals
$\mathbb{R}_{\geq 0}$	Positive reals
$\mathbb{N}$	Set of natural numbers
$\mathbb{T}$	Compact time domain
$\mathbb{X}$	Compact set in $\mathbb{R}$ (*i.e.*, line segment)
$\mathbb{V}$	Compact set in $\mathbb{R}^2$ or $\mathbb{R}^3$ (*i.e.*, volume)

Groups

id	Identity group
SO(3)	Lie group of rotations on $\mathbb{R}^3$ (*i.e.*, special orthonormal matrices)
SE(3)	Lie group of homogeneous transformations on $\mathbb{R}^3$
se(3)	Lie algebra of SO(3)
so(3)	Lie algebra of SE(3)

Vector- and matrix operations

$(\cdot)^\top$	Transpose
$\dot{(\cdot)}$	First time derivative
$\ddot{(\cdot)}$	Second time derivative
$\hat{(\cdot)}, (\cdot)^\wedge$	Isomorphism from $\mathbb{R}^6 \to \mathrm{se}(3)$
$\check{(\cdot)}, (\cdot)^\vee$	Isomorphism from $\mathrm{se}(3) \to \mathbb{R}^6$
$(\cdot)^\circ$	Reference or rest configuration
$(\cdot)^\star$	Optimal solution
$(\cdot)^{-1}$	Square matrix inverse
$(\cdot)^\dagger$	Moore-Penrose pseudo inverse
$(\cdot)^+$	Generalized matrix inverse
$(\cdot)^\perp$	Annihilator

Operators and letter-like symbols

δ	Variation of a field
∂	Boundary of a set
int	Interior of a set
$\sup_t$	Supremum over continuous time t
dim	Dimension of vector
trace	Trace of matrix
diag	Diagonal of matrix
blkdiag	Block diagonal of matrices
$\mathrm{Ad}_{(\cdot)}$	Adjoint action on Lie group
$\mathrm{ad}_{(\cdot)}$	Adjoint action on Lie algebra
$\|\cdot\|$	Euclidean norm
$\|\cdot\|_X$	Matrix norm
$\|\cdot\|_{\mathrm{rms}}$	Root-mean-square norm
$\lfloor\cdot\rfloor_n$	Floor operator
$\lceil\cdot\rceil_n$	Ceiling operator

Acronyms

CoM	Center of mass
FEM	Finite element method (or model)
MDE	Matrix differential equation

ODE	Ordinary differential equation
PDE	Partial differential equation
PMDE	Partial matrix differential equation
PneuNet	Pneumatic network
SRM	Soft robotic manipulator
TopoOpt	Topology Optimization

Contents

I

Introduction

1

Soft Robotics – a new perspective on biomimicry

1.1 Soft robots: what are they?

Biological systems have long served as a source of inspiration for roboticists in their pursuit of developing more robust and capable machines. One might marvel at the ease with which we interact with a diverse array of objects in our daily lives. In contrast, conventional (rigid) robots require precise knowledge of an object's weight, shape, and orientation in order to interact with it in a safe and reliable manner. However, Nature's methodologies have been adopted and translated into a new robotics discipline known as "*soft robotics*". Unlike rigid robots, these machines are fashioned from softer materials characterized by low elasticity and enveloping properties. An examination of common materials employed in robotics reveals a conspicuous gap in comparison to natural materials and their corresponding elastic moduli, as illustrated in Figure 1.1. The absence of material diversity in robotics is believed to be a crucial missing element that could enable modern machines to achieve bio-analogous performance.

In engineering, it is common to use the Young's modulus as a measure of elasticity. While limited to homogenous materials subject to small deformations, it can still be applied to classify rigidity in robotics. To aid the reader's understanding, we have included a spectrum of different materials in Figure 1.1. An observation of this spectrum reveals that biological organisms are primarily composed of low-modulus materials in the range of 10^4 to 10^7 Pa, such as muscle and skin tissue, with rigid materials, such as bone, being much less prevalent. In contrast, classic

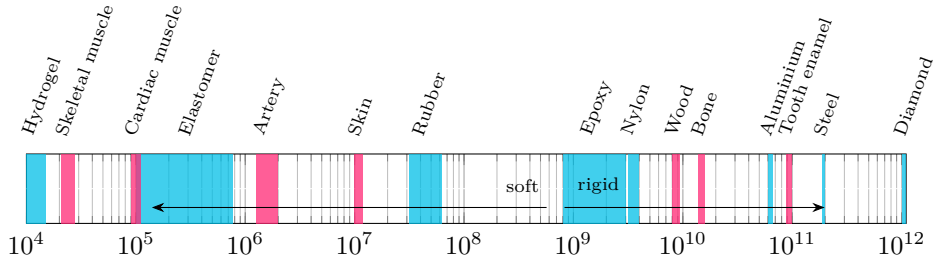


Figure 1.1. Young's modulus spectrum in (Pa) of rigid and soft materials, where (—) are the organic (*i.e.*, biological) materials and (—) inorganic materials.

robotics predominantly rely on hard materials such as metals and hard plastics. Furthermore, it is worth noting that materials which undergo repeated deformation during motion possess correspondingly low elastic moduli, as opposed to the use of rigid materials in classic robotics. The concept of exploring low-elasticity materials, referred to as *"soft materials"*, has fostered a new direction in robotics aimed at unifying robots and biology. We propose the following description before proceeding:

> ***Soft materials*** *are a class of homogenous materials with a Young's modulus (i.e., the modulus of elasticity) typically lower than* $E \leq 10^9$ Pa. *Following, the word softness refers to the collection of mechanical properties that are often associated with these low moduli materials.*

Now, although the words *"soft"* and *"robotics"* have a clear definition independently, the collocation of the two sparked many vivid discussions and new idealogies within the robotics community for the past two decades. Throughout its young academic life, several definition have been coined. Throughout its relatively brief academic existence, various definitions have been proposed. Initially, soft robotics referred to robots with variable joint stiffness [4] or artificial compliance achieved through control [3]. The term was also used to underline the shift from rigid-linked robots to *"bio-inspired continuum robots are inherently compliant and that exhibit large strains in normal operations"* [226]. Paraphrasing the work of Robison et al. [172]: *"soft robotic manipulators are continuum robots made of soft materials that undergo continuous elastic deformation and produce motion through the generation of a smooth backbone curve"*. Alternatively, a broader definition was coined in a review by Kim et al. ([118], 2013) simply referring to soft-bodied robots as *"an analogy to soft-bodied animals"*. A concise (but generic)

definition was proposed by Laschi et al. [125], as soft robots being *"any robot built by soft materials"*. Rus et al. [177] defined soft robots in terms of their structural elasticity: *"Systems that are capable of autonomous behavior, and that are primarily composed of materials with moduli in the range of that of soft biological materials"*.

The ongoing debate regarding the precise terminology for soft robotics may never reach a definitive conclusion. However, it is crucial to establish consistent terminology for the purpose of this thesis. Previous definitions coined by the scientific community have placed great emphasis on the natural motion that arises from soft materials with high similarities to nature. Our definition is based on the historical development of soft robotics and current scientific trends in literature, as discussed in Chapter 2, with particular focus on design and control. Given the interdisciplinary nature of the field, the terms used in this thesis may differ from those used in existing literature. Nonetheless, we will consistently refer to the following when discussing "*soft robotics*":

> ***Soft robotics*** *is a subclass of robotics with purposefully designed compliant actuators embedded into their mechanical structure whose goal is to endow the robot with natural (or biological) motion or compliance.*

This definition is adopted from the work of Della Santina et al. [57], but modified to highlight the importance of soft materials to mimic biological motion – also referred to as *"bio-mimicry"*. The ambition of closely mimicking biological creatures is perhaps not commonly associated with the field of robotics, given its important role in the automation industry, yet the inception of robotics can originally be found in bio-mimicry when regarding its rich history.

1.2 Soft materials in robotics: the pros and cons

1.3 Research disciplines within soft robotics

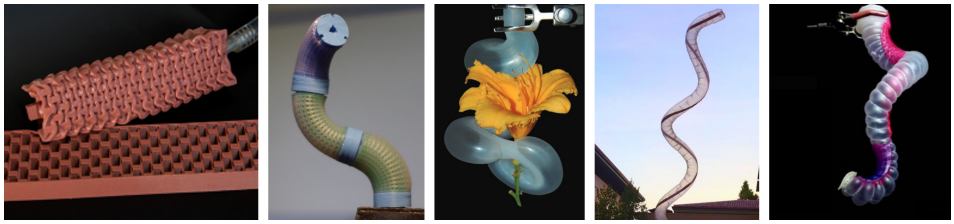
In this section, we will examine several unresolved research issues in the field of soft robotics. These topics primarily concern the design and control of soft robots. It should be emphasized that while these problems may seem distinct, they are often interconnected. For instance, the structural design of a soft robot influences its workspace and consequently affects the feasibility of *a-prioiri* defined control objectives. This section aims to underscore such scientific interconnections and demonstrate the essential components required to address common paradigms.

1.3.1 Tailoring design of fluidic soft actuation

As the name soft robots arises from its use of soft materials, it follows that design and fabrication using soft materials play a huge role in their technological development. Contrary to rigid robots, many soft robots explore whole body movement rather than localized regions undergoing motion called *"joints"*. In classic robotics, robots are composed of a countable number of rigid links and joints [52, 154, 201], either arranged in series or parallel. Together they span a workable range of motion called the *workspace* [201]. Focussing on rigid manipulators, whose base is often structurally fixated, their workspace can be obtained through a system of kinematics, often derived through a set of geometrical equalities. Rigid manipulators often have a bounded workspace (assuming actuation limits). In robotic locomotion, similar kinematic descriptions can be obtained for the legs and feet, with the exception of an additional free-floating base. In these cases, however, the workspace is of less interest, rather the different possibility of *"gait cycles"* that arise from the link-joint configuration and actuator dynamics determine system's success for locomotion.

Returning to soft robots, the definitions such as joints, workspace and gait cycles also apply here. Yet, the high flexibility allows for many non-restricted joint displacements which make deriving closed-form mathematical descriptions challenging. The shape of workspace and locomotion patterns are majorly influenced by geometry of the soft actuator, its flexibility modes, and how forces are transferred with the continuum soft body. Controlling the motion within soft actuation – so to speak reducing parasitic mobility and tailoring motion based on structural geometry – is an active topic in soft robotics research for decades.

Engineering principles in fluidic soft actuators. In the past decade, researchers have developed various techniques of exploiting the high-elasticity of soft materials for *controllable* actuation. One key development, similar working principles to the pneumatic muscle groups (see [93, 151] or Figure 2.3), are Soft Pneumatic Actuators (SPAs). A few examples are shown in Figure 1.2. SPAs undergo similar mechanics akin to McKibben actuators [93] or Morin actuators [151], yet they envelop a diverse collection of motion besides uniaxial. Examples include: contraction and elongation [247], axial growth [87], bending [73, 135, 152], helical and twisting, and a hybridization of all the aforementioned motions [119]. An example of soft actuators capable of contraction is the Vacuum-Actuated Muscle-inspired Pneumatic (VAMP) structures by Yang et al. (2016, [247]). Their work proposes a tailored geometrical structure embedded into a soft elastomer medium that is highly sensitive towards buckling. When subjected to a sufficiently large negative differential pressure, the internal structure undergoes a (reversible) mechanically instable leading to uniaxial contraction, we see in Figure 1.2. Their work

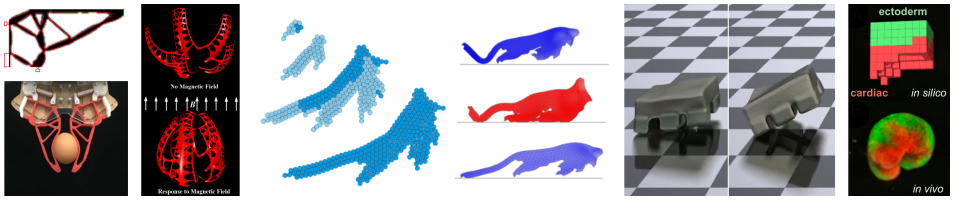


(a) (b) (c) (d) (e)

Figure 1.2. Various examples of continuum-bodied joint motions in modern soft actuation. (a) Soft actuator undergoing contraction by Yang et al. [247]. (b) Set of serial-chain of bending soft actuator by Cianchetti et al. [48, 49] (c) Soft tentacle composed of twisting soft actuators. (d) Vine-inspired soft actuators capable of growth by Hawkes et al. [87]. (e) Soft manipulator composed of hybrid bending and twisting soft actuators through laminate materials by Kim et al. [119].

is inspired by a similar buckling behavior of patterned elastomer [23, 153, 186] subjected to axial loads. These muscle-inspired vacuum soft actuators are fast, produce a stable, repeatable motion; and more importantly, explore structural geometry to reduce parasitic motion. An example of soft bending actuators is the FLIP-FLOP system [48]. Akin the ORM system [75], it has three pressure chambers embedded into a soft cylindrical-shaped elastomer. To prevent ballooning, inextensible rings are placed orthogonal to the deformable backbone. It design is also reminiscent of [203, 204]. Hawkes et al. (2017, [87]) developed a soft manipulator inspired by the growing behavior of vines. Kim et al. (2019, [119]) used laminates that adhere to the volumetrically expanding soft body as to govern the motion trajectory through bending and twisting.

Exploring optimization and evolutionary algorithms. Besides design through engineering principles, optimization in soft robotics is slowly gaining momentum recent years. Wang et al. (2020, [231]) used a topology optimization to find the optimal design for a cable-driven soft gripper (Figure 1.3a). Similarly, Tian et al. (2020, [219]) explored topology optimization for ferro-magnetic soft grippers. Besides soft grippers, evolutionary design algorithms are also employed for soft mobile like crawler and swimmers. Joachimczak et al. (2015, [105, 106]) explored evolutionary search algorithm with the purpose of automatically designing complex morphologies and controllers of multi-cellular, soft-bodied robots (Figure 1.3c). Hu et al. (2019, [97]) used a differential physics simulator called `DiffTachi` that efficiently computes gradient information for each simulation timestep. The gradient information can then be fed into neural network controllers to solve, for instance, the appropriate gait-cycles required in the locomotion of soft-bodied crawlers (Figure 1.3d).



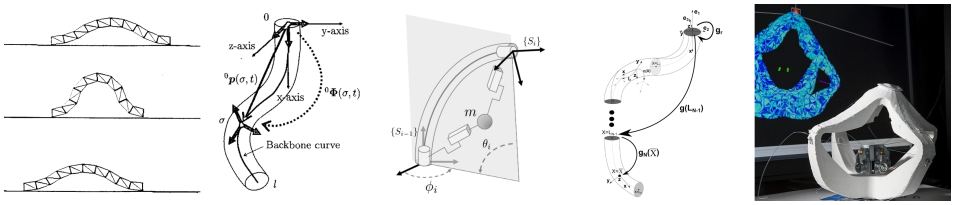
(a) (b) (c) (d) (e)

Figure 1.3. Optimization for design and motion of soft robots. (a) Soft gripper by Wang et al. [231]. (b) Topology optimization for ferromagnetic grippers by Tian et al. [219]. (c) Evolutionary algorithms for multicellular soft-bodied robots by Joachimczak et al. [105, 106]. (d) `DiffTachi` result for soft crawler by Hu et al. [97]. Voxel-based optimization for xenobots [122].

1.3.2 Gaining performance through modelling and control

As the inherent properties in soft materials bring forth many benefits, *e.g.*, adaptability, hyper-redundancy, and passivity w.r.t. the environment; it too hinders the progress in model-based controllers. Earlier, we have touched upon these subject with the rise of kinematic and dynamic models for hyper-redundant robotics in the late 80's – early 90's. Chirikjian et al. (1992, [42]) provided a kinematic framework for hyper-redundant manipulators with application to motion planning (see Figure 1.4a). Here the elastic backbone is approximated using a modal formulation. Such modeling framework are one-to-one transferable to soft continuum manipulators. Mochiyama et al. (1998, [146, 148]) extended this work to a dynamics formulation – even providing Lyapunov-based control strategies for shape regulations (Figure 1.4b). However, both modeling frameworks were computationally inefficient – lacking transferability to real-time control. The root problem stems from the fact that soft continuum robots, belonging in their exact formulation to the field of continuum mechanics, lead to infinite-dimensional models often expressed as Partial Differential Equations (PDEs). Rigid multi-body systems, like robot manipulators or mobile robots on the other hand, have convenient Ordinary Differential Equation (ODEs) structures as they are rooted in Lagrangian or Newtonian mechanical principles. Rigid-body models were (and still are) fast computationally and their literature on controller design is vast and well-established [52, 154, 201]. The computational issues in early continuum robots may be reflected by its literature gap between the 1990's to 2010's.

A: Modern control-oriented models for soft robots. In the past decade, significant steps have been made to address the issues of infinite dimensionality [58]. The key is to formulate a finite-dimensional approximation of the soft robot's dynamic such that they can be written as standard ODEs. Reduced-Order Models



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

Figure 1.4. Popular modeling strategies for soft robotics. (a) Hyper-redundant modeling description through tensegrity by Chirikjian [42, 46]). (b) Analytical continuum beam description by Mochiyama [147, 148]). (c) Augmented rigid-body model subjected to PCC kinematics by Katzschmann et al. [110]. (d) Geometric Cosserat model subjected to PCS kinematics by Renda et al. [170]. (e) Diamond-shaped soft robot manipulator controlled using the FEM-based `SOFA` software by Duriez et al. [62] and related [51, 82].

– ROM for short – have paved the path of model-based controller for soft robots whose reduced formulations are both tractable and precise. In the years shortly after its academic boom, many different assumptions and model approximations have come forth to address the issue of control.

Remark 1.1 The reduced-order formulations for soft robotics are primarily applicable to slender soft robots, leading to a focus on soft robot manipulators in control-oriented studies. This approach is well-motivated given that many soft robots have one dominant physical dimension compared to the other two [58]. As such, this thesis primarily focuses on the modeling and control of soft manipulators rather than a broader scope.

Focussing on soft robot manipulators first, a popular choice of finite-dimensional reduction is the so-called *Piecewise Constant Curvature* (PCC) soft beam model. The PCC modeling approach is by far the most adopted in the soft robotics community [235]. As the name implies, the soft robot is modelled as an elastically deformable beam with all strains but curvature neglected. Examples of this approximation include [59, 65, 70, 109, 110, 135, 176, 235]. Highlighting a few, Katzschmann et al. (2018, [109]) proposed to connect the PCC formulation to an augmented rigid robot dynamical model with parallel elastic actuation (see Figure 1.4c). A similar approach was proposed earlier by Falkenhanh et al. (2015, [65]) and applied to Festo's bionic arm [85]. Although such lumped models may seem like a major over-simplification, the proposed model allows sufficient speed and accuracy such that model-based feedback is applicable. This formulation was also employed later in an adaptive sliding mode control scheme [112] for the SoPra

soft arm [225]. Following, Renda et al. (2018, [170]) extended the PCC model to *Piecewise Constant Strain* (PCS) formulation. This formulation allowed for all strain if and only if considered spatially piece-wise constant (Figure 1.4d). Rooted in SE(3) geometry of the Cosserat approach [188], it provided a closer relation with the rigid body geometry of the traditional robotics. The formulation also extends to soft manipulators with fluidic actuation [171, 220] The PCS model was later employed to feedforward controllers by Thuruthel et al. (2018, [218]) using model-based policy learning algorithms. To improve efficiency, a recurrent neural network was trained using an offline PCS model. Grazioso et al. (2019, [84]) explored a similar path of geometric Cosserat beams using helical strain functions. Nevertheless, Constant Strain (CS) models have severe limitations. They often do not originate from continuum mechanics and thus are only applicable in restrictive settings. Although computationally performance might surpass continuous models, due to intrinsic kinematic restrictions, they are unable to capture important continuum phenomena, like buckling, environmental interaction, or wave propagation.

In response to its limitation, many researchers continued their search for efficient and more generalizable alternative. Della Santina et al. (2020, [60]) proposed a polynomial description to described the continuum dynamics, a description analogous to [42]. In their work, they expressed the curvature function of the soft robot in terms of a standard polynomial bases. Not only can an exact infinite dimensional formulation of the problem be obtained (in theory), truncation at any level is changed easily. The technique is also widely used for flexible-link robot manipulators to capture small vibrations, see DeLuca et al. (2016, [56]). Della Santina et al. also showed that PCC-rooted assumptions as control output produce a minimum phase system [60] – a fundamental stepping-stone for nonlinear control. Following, Boyer et al. (2021, [27]) extended upon their prior Cosserat models [169, 170], and presented a tractable and generalizable beam model for slender soft manipulators. A similar approach to [60] was followed but all strains are discretized using a finite set of strain basis functions. Renda et al. (2020, [169]) improved computation by introducing a two-stage Gauss quadrature [250] to derive the Magnus expansion [86]. Other examples of the Cosserat beam descriptions, but more focussed on the continuum mechanics rather than control, is the work Gazzola et al. (2018, [77]). Their work allowed for efficient Cosserat beam models suitable for self-collision, thus providing various simulation for twirling and coiling of beams under increasing torsional loads.

Another popular alternative, better suited for general soft robotic systems like soft mobile robots, are reduced-order finite element models [50, 51, 62, 82, 111, 214, 215, 224, 243, 252] or neural-network trained using offline FEM simulations

[66]. Starting from high-order FEM data (*e.g.*, state dimensions of the order 10k) that capture the whole workspace spanned by the network of soft actuators, Proper Orthogonal Decomposition (POD) techniques are employed to drastically to reduce the state dimension of the soft robot model. These techniques can even retain external loads (*e.g.*, contact and friction) with precision as long as they are included in the offline data set [82]. By far, this FEM-driven method has shown the most success in the experimental control regime.

Soft robot simulation and programming environments. The rapid development of soft robotic models in recent years have also increased the demand for (open-access) software packages. Especially since many of the aforementioned ROM models require an advanced level of mathematical understanding. In an attempt to help the soft robotics community, many researchers have provided open-source, documented simulators interwoven in their soft robotics research. A popular FEM-based software on soft robotic modeling and control is `SOFA` by Duriez et al. (2013, [51, 62]). `DiffTachi` by Hu et al. [96, 97] explores differential simulations to produce soft machine capable of locomotion. Bern et al. (2022, [19, 20]) developed `SoftIK`, a software for soft deformable plushy robots. Among beam or rod-based models there exist many options. Examples included `Elastica` (or `pyElastica` [211]) by Gazzola et al. [77, 251], `TMTDyn` by Sadati et al. [178], `SimSOFT` by Grazioso et al. [84], and `SoRoSim` by Mathew et al. [138] based on the work of Renda et al. [169] and Boyer et al. [27]. The `Sorotoki` toolkit by Caasenbrood et al. [33] – open-source software package presented as a part of this thesis – explores a combination of FEM models and soft beam models. The toolkit written in Matlab aims to bridge the gaps between design, modeling, and control of (hyper-elastic) soft robots.

Closing the loop in soft robotics. Following the many developments in computational efficiency of reduced-order models (and accordingly the advances in soft sensing), academic research in model-based or model-free control for soft robots is significantly growing since early 2019. Note that feedforward controllers for soft manipulators have been proposed years prior, *e.g.*, [64, 65, 181, 217].

By far, experimental validation of PCC model has been used more intensively than other models. In Della Santina et al. (2019. [59], the augmented rigid-body PCC model is used to design a closed-loop controller for a continuous soft manipulator, presenting two architectures designed for dynamic trajectory tracking and surface following. Prior work is provided here [110]. A similar approach was followed by Milana et al. (2021, [**?**]) and applied to an artificial soft cilia. They showed that soft bending actuators could mimic the asymmetric motion of the cilia through model-based control. Cao et al. (2021, [37]) explored a reduced analytical model [233], somewhat equivalent to a linear pendulum model

(apart from quadratic terms in the potential force); to develop robust tracking controllers without velocity observers. Their controller was tested experimentally on a soft PneuNet actuator. Wang et al. (2022, [234]) developed a computed torque controller (see [201]) using the augmented rigid-body PCC model and applied it to a soft Honeycomb Pneumatic Network Arm [104]. Franco et al. (2020, [70, 71]) used a port-Hamiltonian modeling framework akin to rigid-body PCC model (*i.e.*, three-link pendulum) and applied such principles to energy-shaping controllers. The performance of their controller was assessed via simulations and via experiments on two soft continuum prototypes. On a side note, Franco et al. (2022, [68]) also developed an energy-shaping control law together with nonlinear observers for the control of soft growing robots [87] with pneumatic actuation subject to the (ideal) gas laws.

As mentioned previously, the PCC model has significant limitation that impose questions on the useability, dexterity, and robustness of their control derivation. Although majorly focussed on simulation, higher-order dynamics have been used for the development of feedback controller in soft manipulators. Della Santina et al. ([58]) developed swing-up controllers for a soft pendulum modelled by the affine curvature models (*i.e.*, a polynomial curvature model [60] of order $k = 2$). Their approach mirrors the path of classic control problem of inverted pendulums in the 90s and early 2000s [159, 187, 199, 200]. Later, Weerakoon et al. (2021, [236]) extended upon their work by introducing a revolute base. A common control problem here is under-actuation [154, 201, 209] – implying that not all control actions can be realized to steer the configuration space to a desired position. In the work of Borja et al. (2022, [26]) developed a general control framework that can stabilize soft manipulators based on potential energy shaping based on the affine curvature model. Their work showed that some linear matrix inequalities can be derived based on the gradient of the potential energy related to the passive and active states, such that local stability can be proven. In laymen terms, elasticity must dominate the forces resulting from the gravity in the underactuated states for (potential) energy-shaping controllers (and mostly-like others) to work.

1.4 Research objectives and contributions

In this section, the research objectives and contributions are specified. The research is divided into three unique branches each related to a specific subproblem within the field of soft robotics: (I) design synthesis of soft actuators, (II) modeling for control of soft robot manipulators, and (III) the development of software aimed to support the soft robotics community. For each research objective, a number of contributions of the thesis are presented.

A: Design synthesis of soft actuators. As presented by abundance of literature on soft robotic design, either rooted in engineering principles or through optimization, achieving an optimal structural geometry that fully accounts for hyper-redundancies in soft materials is no easy feat. Unlike their rigid counterparts and many biological systems for that matter, the kinematics are inherently encoded in the topological structure of the soft materials and where actuation is presented in the system. This implies the workspace cannot be characterized analytically in closed from through joint motions stemming from one point (unlike joints in rigid robotics), see Figure 1.5a. Futhermore, any external inputs will result in parasitic motion, which are mainly induced by distributed continuum deformations that are antagonistic to the input. These parasitic motions – or better phrased *"passive joint displacements"* – arise from the redundant flexibility of the system that are often unaccounted for during design. As such, they should be kept at as minimum as they lead to imprecisions in the mechanical operation and lost of mechanical efficiency (*e.g.*, the balloon effect [54]). Futhermore, there exist many elasticity moduli for various options of soft material, ranging from soft gels to hard rubber [177]. It is therefore of paramount importance that the nonlinear behaviors of soft materials, both in hyper-elasticity and nonlinear geometrical deformation, is understood and accounted for during the design process. This deepens the research problem by imposing questions on optimality regarding material choice.

(Optimality in soft material design). Our first research objective therefore focusses on the design. The main design principles for any compliant mechanical device can be described rigorously through continuum mechanics. In the thesis we focus on harmonizing the underlying continuum theory with the automated design of efficient soft actuators with user-defined objectives in mind (Figure 1.5b). Rooted in continuum mechanics [95, 117] (and its subsequent discretization through finite elements), optimization algorithms are employed that aim to seek an optimal layout of soft materials such that user-specified objective functions are minimized (Figure 1.5c). Such problems are often referred to an *inverse design problem* – solving shape by knowing deformation, rather than solving for deformation based on the shape. The study of combining continuum mechanics

Figure 1.5. Schematic illustration of the inverse design problem in soft robotics. (a) Bending behavior of PneuNet actuator under linearly increasing pressure. (b) Desired end-effector position ($\star$) outside the robot's workspace, changing the input is not sufficient will not improve objective – the workspace is *a-priori* encoded into the soft topology. (c) Solution to inverse design problem by finding a soft topology that contains ($\star$) in its workspace.

and free-form optimization in compliant structures is well-established field called *topology optimization* [17]. Yet, such practices are not easily transferred to soft actuation, since soft materials introduce hyper-elasticity and nonlinear geometric deformations. Besides, fluidic or pneumatic actuation complicates the optimization, as these loads become both design and state-dependent. This brings us to the first contribution:

> **Contribution I.** *Development of efficient algorithms, applicable to the general design of fluidic soft actuators, solving the inverse design problem: Given a desired motion and input, what is accordingly the optimal soft material distribution within a design domain to realize such joint motion?*

(Fabrication through Additive Manufacturing). Following these automated algorithms, many variations of soft actuation system with various joint mobility can be developed with relative ease. Yet, being of simulation origin, this raises the questions on the transferability of the numerical studies to practice. Hence, our second research objective therefore focusses on the transferring simulation design to feasible, functional soft actuators. Through the recent advances in Additive Manufacturing, many complex three-dimensional geometries can be fabricated with relative ease and effort.

> **Contribution II.** *Fabrication of an array of computer-optimized pneumatic soft actuators through Additive Manufacturing methods, whose collective assembly can be explored for soft robotic manipulation.*

B: Modeling for control of soft manipulators. Besides design, modeling for control is another important topic in soft robotics. In particular with the aim of achieving biological performance in modern soft robotic systems, models must be both accurate and fast. However, the infinite-dimensionality that are inherent to these soft continuum robots present some challenges. Augmented rigid-body models, like [70, 110**?**], offer exceptional computation speeds but do not

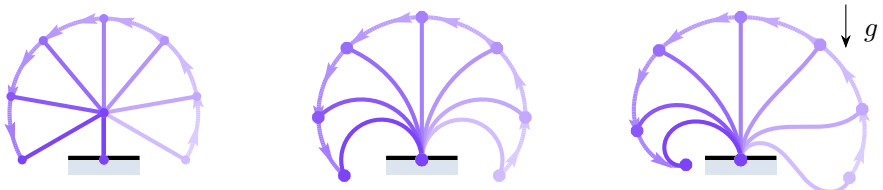


Figure 1.6. Stages of kinematic complexity in (soft) manipulators. (a) Standard 1-DOF rigid manipulator with analytic workspace. (b) Ideal soft manipulator deformed under uniform curvature (*i.e.*, PCC model). Workspace can be analytically expressed for non-complexity cases, *e.g.*, stiffness dominates gravity or homogenous deformation. (c) Truly under-actuated soft manipulator starting from initial (gravity-balanced) condition. Workspace can often not be derived analytically and depends highly on the initial conditions and actuation constraints.

respect the fundamental continuum mechanics, therefore invoking strict operation constraints on any soft system, *i.e.*, small deformation as to limit hyper-elastic nonlinearities; or slow actuation to prevent inertial mismatching.

(Accurate control-oriented models through PCC condition). Our second research objective therefore focusses on the reduced-order modeling, aiming to balance precision and speed for control. By building upon the original works of Chirikjian et al. [42] and Mochiyama et al. [145] in the 90's, the thesis presents a dynamic modeling formulation for soft manipulators that respects its continuum nature. To address the issue of infinite-dimensionality, a reduced-order modeling strategy for soft robot manipulators is proposed, whose mathematical framework is based on the differential geometric theory of spatial curves. Such framework allows for easy transferability to classical ROM models in soft robotics, like the PCC strain [60, 65, 110] (see Figure 1.6b). However, the thesis proposes two improvements that are essential development of model-based controllers. Inspired by the success of FEM-based models in soft robotics [62, 82], we bridge the gap between the PCC model and the underlying continuum mechanics by matching the quasi-static behavior to a Finite Element Model (FEM). The numerical FEM approaches used are analogous to Contribution I. Second, to enhance computational efficiency, we propose a reduced-order integration scheme using Matrix Differential Equations (MDEs) to compute the spatio-temporal dynamics in real-time. The proposed reduced-order soft beam model is tested rigorously in simulation; however, for practical control tests demand experimental rigorousness. As such, a

3-DOF soft robot manipulator is developed *ad-hoc* through Additive Manufacturing. The system is loosely inspired by the *Orm* soft manipulator from the early 60's [75]. The proposed dynamic model is tested under various experimental conditions. For example, natural oscillations, forced inputs, and under tip-disturbances of various inertial mass. Various performance measures are introduced to compare the proposed model objectively with respect to linear elastic alternatives. This brings us to our third contribution:

Contribution III. *Development of fast, efficient, and accurate dynamic models for soft manipulators composed of hyper-elastic soft materials that are directly applicable to classical control theory akin to rigid robotics.*

(Models beyond the PCC). As mentioned in Section 1.3.2, the piecewise continuity inhibits many kinematic redundancies – and therefore the hyper-redundant nature of these soft robotic systems cannot be explored to its full potential. For example, any large nonlinear deflection due to gravity (see Figure 1.6c) cannot be captured using these constant descriptions. Also, the study of (true) underactuation through such kinematically lumped models is (nearly) impossible as the modeling framework infers homogenous strain and actuation. Herein lies the third research objective that focusses majorly on relaxing the PCC condition, to closely pursue its true infinite-dimensionality. Adopting the prior modeling strategies of differential curves, spatially-varying strain fields are considered and approximated (closely) through sets of orthogonal shape functions rather than prior piecewise representations. Building upon prior models presented in the thesis, the underactuated and hyper-redundant soft system can be casted into a port-Hamiltonian framework [160, 182]. This allows for classic controller design through energy-shaping by modifying the closed-loop potential energy of the system – a well-known practice in classic robotics [159, 160, 182]. Exploring the hyper-redundancy in soft robotics, more advanced control objectives can be realized such as shape regulation and full-body grasping of rigid objects. However, spatial discretization in these model-based controller play a crucial part in their dexterity to achieve various control tasks. Within this context, the fourth contribution reads:

Contribution IV. *Investigation of spatial discretization in low-order energy-shaping controllers applied to high-order soft robotic models with a focus on closed-loop performance in shape and full-body grasping control.*

(Exploiting geometry for reduction). Contribution IV provides a stable mod-

eling platform for a variety of possible shape functions tailored to unique joint mobilities in soft robotic systems. Yet, many works in modeling literature choose such functions in *ad-hoc* fashion, *e.g.*, polynomial bases [27, 43, 60]. Within this context, the thesis explores a (geometric) modal decomposition approach. Similar to the eigenmode analysis in continuum mechanical systems, geometric strain modes are extracted from higher-order (volumetric) FEM models and used to construct optimal soft beam models. The approach leads to fast, accurate, and generic low-dimensional models that encode the geometric features and elasticity of the true soft body into a new strain parameterization - we call a Geometry-Informed Variable Strain (GIVS) basis. A merit benefit of the approach can be naturally expanded to identify the hyper-elastic material parameters and the actuation map of the reduced beam model. Robustness of the technique is investigated experimentally for several soft robotic systems, including PneuNets, soft grippers, and soft manipulators. In context of robustness, the nonlinearities with increasing actuation frequency, under environmental contact modeled by signed distance functions, and multi-input pneumatic actuation are also considered. The thesis continues with a qualitative comparison between existing strategies, demonstrating that our approach can improve accuracy and speed compared traditional techniques. Our fifth contribution therefore reads:

> **Contribution V.** *A novel method for finite-dimensional model reduction in soft manipulators that explores the mechanical interconnection between structural geometry and flexibility modes of the soft manipulator body.*

C: Software development. Finally, the thesis collects all theoretical material into one unifying soft robotics software package called `Sorotoki` – short for SOft RObotics TOolKIt. The software aims to bridge the gaps between different disciplines of soft robotics, applicable to design, modeling and control. Using a minimal programming framework, complex problems can be coded using minimal lines of code. The toolkit is heavily interwoven with Contributions I to V of the thesis, and is publicly available at https://github.com/BJCaasenbrood/SorotokiCode [33]. The last contribution therefore reads:

> **Contribution VI.** *Developement of a versatile, user-friendly, open-source software called* `Sorotoki` *that envelops the presented theory on design, modeling and control of the thesis into one coherent Matlab toolkit.*

1.5 Outline of the thesis

This thesis discusses the design, modeling and control of soft robotic systems. Including this introductory materials, the thesis consists of seven chapters.

Chapter 2 presents the design algorithm for soft actuators that aim to solve the inverse design problem. The chapter start with a brief introduction into continuum mechanics applied to three-dimensional deformation of hyper-elastic materials, followed by its numerical implementation using finite elements. From here, numerical optimization procedures are introduced that solve the inverse design within the context of (fluidic) soft actuation. Chapter 3 follows with the second objective of the thesis, namely modeling for control. Instead of volumetric soft robotic models, lower-dimensional soft beam models are introduced that are tailored for fast and accurate model-based controllers. The chapter focusses primarily on PCC soft beam models. Chapter 4 address the limitation of the PCC model, and instead extends upon it. The chapter formulates a finite-dimensional port-Hamiltonian modeling approach for soft beams, where spatial shape functions are used to discretize the modal flexibilities of the soft robot. From here, energy-shaping controller are introduced that allow for shape control and grasping of objects. A major emphasize on the chapter is the effects of discretization in soft robotics and choice of control gain, and their subsequent effects on the closed-loop solutions. Chapter 5 focusses on the Geometry-informed Variable Strain description that ties together Chapter 2 and Chapter 4 into an (optimal) reduced-order modeling and control framework. Chapter 6 presents the culmination of all theoretical material presented in the thesis into a concise, user-friendly, toolkit called `Sorotoki`. The chapter presents an overview of the included programming tools for the design, modeling and control of soft robots. Finally, Chapter 7 closes the main body of the thesis by summarizing the research deliverable of prior chapters, and provides a list of recommendations that could sculp future work.

Note for the reader. Chapters 3-6 are all based on published or submitted researcher articles and can therefore be read independently. A reference to the corresponding research paper is provided at every beginning of these chapters. This thesis, however, provide some minor modifications to these works, either in the context of mathematical or material improvement, or connection between other chapters. An overview of these modifications can be found as a supplementary chapter at the end of the thesis in the chapter named *Modifications*.

2

A brief historical overview on the progress of soft robotics

Abstract - This chapter presents a detailed chronology of the evolution of soft robotics, from its inception in the early 1950s to current research trends. The aim is to provide readers with an insightful introduction to the extensive field of soft robotics. The chapter commences by tracing the origins of soft robotics in pneumatic muscles, including the pioneering work of Joseph McKibben and Victor Scheinman. Subsequently, we explore the emergence of novel concepts and technologies that facilitated the development of increasingly sophisticated soft robots through the utilization of exotic material properties. Throughout the chapter, significant milestones in the development of soft robotics are highlighted, such as the creation of the first soft gripper, the integration of robotics with soft actuations, and the introduction of design and modeling principles for these systems, and early control approaches. Additionally, the chapter highlights some of the key challenges that lie ahead for the field, which serve as a basis for standardization of terminology.In short, this chapter offers an insightful and comprehensive perspective on the history of soft robotics, and will aid the reader in solify the thesis's objectives introduced in the chapter prior.

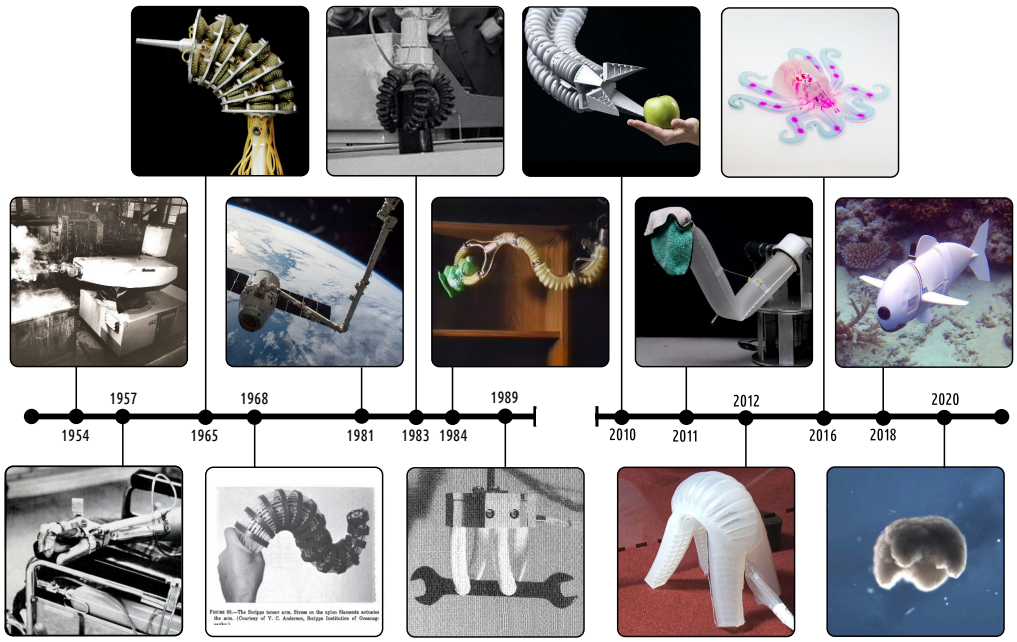


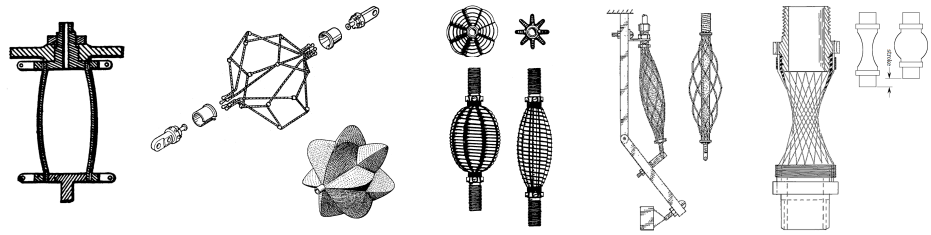
Figure 2.1. A brief timeline of the state-of-the-art of bio-inspired robotics throughout human history. (1954): Unimate, the first industrial robot . (1957): McKibben actuator, an early soft actuator inspired by the human muscle used for rehabilitation purposes [93]. (1965): The Orm, believed to be the first soft robotic system designed by Scheinman and Leifer [75]. (1968): Tensor Scripps arm developed by Anderson [8]. (1981): Canadarm-1, early flexible robotics employed on the International Space Station. (1983): Robot Arm with Pneumatic Gripper by Teleshev [94]. (1984): Bellows robotic arm by Wilson et al. [239]. (1989): The soft robotic gripper developed by Suzumori et al. [203, 204], seen as one of the earliest *academic* soft robot, developed before the word *soft robot* existed. (2010): Festo's Bionic arm inspired by the elephant's trunk [85]. (2011): Soft inflatable robot arm by Sanan and Atkeson [13, 180]. (2012) Multi-gait soft robot capable of terrestial locomotion [47]. (2016): Octobot, the first autonomous 3D-printed soft robot that explores a stabilizing oscillator chemical network that produces preprogrammed repetitive motion [237]. (2018): Autonomous robotic fish made by Katzschmann [109]. (2020): Xenobot, an organic soft robot composed of skin and muscle cells made by Blackiston and Kriegman [122].

2.1 Early soft robots

The ambition of closely mimicking biological creatures is perhaps not commonly associated with the field of robotics, given its important role in the automation industry, yet the inception of robotics can originally be found in bio-mimicry when regarding its rich history. In this section, we will present a short historical overview of soft robotics. Hereby showing that the current trends of bio-mimicry and elasticity in robotics find roots in a periods way before the soft robotic boom in the early 2010's. To guide the reader, in Figure 2.1, we provide a graphical, historic overview of soft robotic systems from 1960 to 2022. We will discuss the inception of soft actuation, early soft robotic designs, and modeling and control strategies for these continuum robotics.

To relate the historical progress of soft robots to rigid robots, let us begin with early rigid robots. In 1954, George Devol filed a patent describing an autonomous robotic machine that could be preprogrammed to execute step-by-step motions [143]. The machine was designed to reduce the workload on the manufacturing work floor, with a major focus on mimicking repetitive (exhausting) human labor. In 1958, those prototypes led to a robotic system under the name *Unimate*. An illustration of this early rigid robot is shown in Figure 2.1. The Unimate was used for manipulating metal die-casts and welding these to the main body of automobiles. In doing so revolutionizing the car industry shortly after. Much later (1969), Victor Scheinman created the Stanford Arm [2, 75], recognized as the first electronic computer-controlled robotic arm because the Unimate's instructions (*i.e.*, predefined setpoints in joint space) were prerecorded on a magnetic drum. He later developed the well-known PUMA robot in 1972 (video available at [1]) – the successor to the Unimate. Keep Scheinman in mind, as he ultimately ties to early soft robots.

Nearly four years after the Unimate was developed, Joseph McKibben developed a pneumatic muscle-inspired actuator capable of linear contraction – called the McKibben actuator. The McKibben muscle is a type of Pneumatic Artificial Muscle (PAM) which is to this date the most frequently used and published artificial muscle in literature. According to [93], he developed the McKibben actuators to bring motion to his little daughter's polio-paralyzed hand. His aim was that eventually such pneumatic actuators may help patients with paralyzed fingers to move, grasp, and even write. Inspired by the human muscle, the McKibben actuator consists of an inflatable inner bladder enveloped with a double-helical weave. When pressurized, the fluidic actuator converts radial expansion into uni-axial contraction [54, 55, 184] since weave inhibits extensive *ballooning* – a term for undesired rapidly-accelerated volumetric expansion. Its material composition is often



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

Figure 2.2. Patent diagrams of pneumatic artificial muscle from 1953 till 1988. (a) Morin Muscle [151]; (b) ROMAN muscle [103]; (c) Yarlott muscle [249]; (d) Kukolj muscle [123]; (e) Paynter Hyperboloid [163].

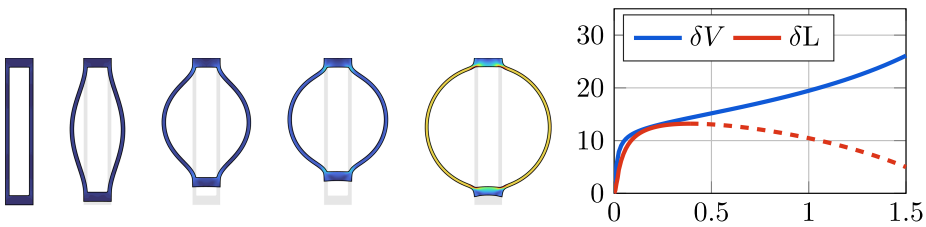


Figure 2.3. Working principle of a basic pneumatic artificial muscle (*i.e.*, Morin muscle [151]) with the internal volume (—) in mL, and the end-effector displacement (—) in mm and (--) is the point at which the undesirable ballooning occurs.

silicone rubber with a nylon-fiber exterior. A schematic representation of a general pneumatic muscle and the effect of ballooning are shown in Figure 2.3. Ballooning is an (often undesired) nonlinear effect, where the hyper-elastic pressure vessel exhibits strain-softening after a critical point is reached. As a result, further increase of the pressure leads to an exponential growth in volume, which ultimately leads to actuator tearing. At stages of ballooning, mechanical performance significantly drops and even produces adverse effects, like actuation reversal. McKibben solved this problem through a combination of soft and inextensible fiber weaves. These inextensible were placed at the exterior wall of the soft muscle, thereby limiting the radial expansion before ballooning could occur. According to Daerden (1999, [54]), there exist many variations of pneumatic muscle besides braided muscles, such as the *netted muscles* (e.g, Yarlott [249], ROMAC [103], and Kukolj [123])

and *embedded muscles* (*e.g.*, Morin [151], Paynter Hyperboloid [162]). Illustration of their patent schematics are shown in Figure 2.2.

Pneumatic muscles are perhaps one of the first fundamental technologies that enabled soft robotics, and to this day, it remains a framework for many soft robotic systems. Nevertheless, besides the many examples of fluidics [109, 134, 135, 152, 203], there exist many other technologies employed in soft robotic motion: such as thermal [244] or chemical expansion/contraction [15, 221, 237], crystal re-alignment [130, 165, 166, 228], di-electric elastomers [113], magnetism [29, 120, 140, 174], and naturally the use of tendons paired with electro-mechanical actuation [19, 50, 116, 170, 230]. Some predate the invention of the McKibben actuator. For example, a popular soft actuation principle still applied in soft robotics today are Dielectric Elastomer Actuators (DEA) developed by Röntgen in 1880 [175]. Therefore, given the abundance of soft robotic actuation, it is difficult to pinpoint the exact date of origin of soft actuation technology. Note, however, that these systems are not categorized as soft robots, they are categorized as soft actuators. Here, we emphasize the difference between soft actuators and soft robots in view of the modeling and control terminology relevant to the thesis:

> ***Soft actuators*** *are controllable flexible actuation units of the constitute soft robot that through external stimuli are responsible for natural motion and/or change in adaptive compliance.*

Remark 2.1 The terminology above attempts to address an ambiguity common to soft robotics, namely the interchangeable use of soft actuator and soft robot. The thesis invokes that soft robots must be comprised of multiple soft actuators that connect to a passive deformable body. Here, the soft body functions as a mechanical conduit between actuators, sensors, and the environment.

2.2 Recognition of soft robotics' potential

Returning to 1965, nearly a decade after the invention of the McKibben actuator and the Unimate robot, Scheinman and Leifer proposed a novel pneumatic robotic arm named the *Orm* – Norwegian for snake (recall that he also developed the popular PUMA robot [1]). The name was also an abbreviation for Object-Relational Mapping tool [53]. To the author's knowledge, this is believed to be the first instance of a soft robotic system. Surprisingly the system predates any rigid snake-like robot, like the Scripps tensor arm by Anderson (1968, [8]). Inspired by the anatomy of snakes, the system featured 28 rubber pneumatic ar-

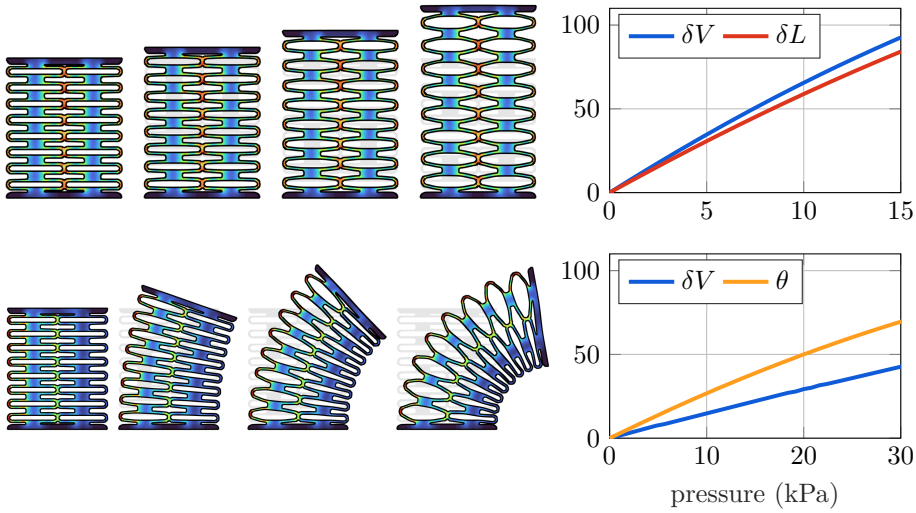


Figure 2.4. Working principle of the Orm robotic manipulator [75] with the internal volume (—) in mL, and the end-effector displacement (—) in mm and bending-angle (—) in deg. By actuation of the pneumatic network, both elongation and bending can be achieved. Observe that the response is significantly more linear than McKibben actuators in Fig. 2.3, emphasizing the importance of geometry.

tificial muscle (*i.e.*, bellows) distributed along a flexible backbone (*i.e.*, skeletal support). The network of artificial muscles were sandwiched between steel plates to prevent misalignment. It is worth mentioning that the technology is analogous to the pneumatic McKibben muscle, where fiber weaves are used to prevent ballooning. Yet, contrary to a single McKibben actuator, the soft robotic system could undergo three-dimensional movement by inflation or deflation of an embedded pneumatic network. This led to a rich set of movements previously unseen in rigid robotics. As an illustrative example, we provided the mechanics of the Orm soft robot in Figure 2.4. The soft robot could achieve bending in any preferred direction by differential pressurization of each channel, and elongation through synchronized actuation. Most notably, comparing the volume-strain response of the Orm with respect to the McKibben actuator, *i.e.*, comparing Figure 2.3 against 2.4, it is noticeably more linear in nature. Although not documented at the time, the comparison highlights the importance of structural geometry in pneumatic muscle networks.

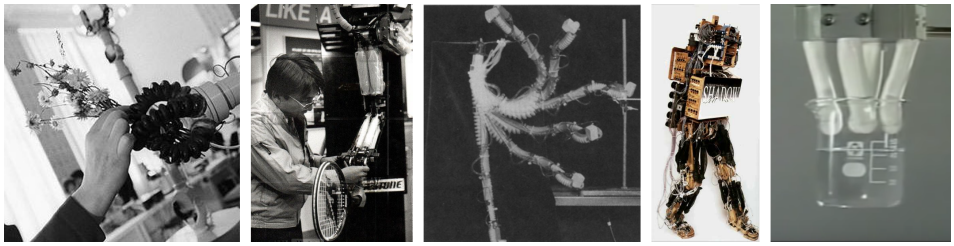
According to an interview with Scheinman led by Asaro et al. [164] in 2010, the positional accuracy of the system was poor, yet the concepts of pneumatically-driven soft arms continued for many years. The positional inaccuracy of pneumatic

soft actuators at the time may have caused its lost of academic interest in the 60's. Three years later, in 1968, an improved hyper-redundant robot manipulator was proposed and patented by Anderson and Horn [8] (see Figure 2.1). Improving upon the Orm, which was deemed slow and had limited positional accuracy, Anderson proposed an array of nylon tendons that were connected to rigid discs distributed along the redundant backbone of the robot. The configurable backbone was comprised of universal spherical joints that allow for pivoting motion with respect to other discs – totalling 16 Degrees-of-Freedom (DOFs). The entire arm was actuated hydraulically, yet the (soft) actuators were placed outside the robot's body rather than placed at each joint, like the Orm. To improve positional accuracy further, Anderson placed sensor tendons parallel the actuator tendons which allowed for operator-based positional feedback. Although Anderson's robot does not categorize as a soft robot since it relies mostly on rigid materials, its flexibility arose from thin nylon tendons that were used for both actuation and sensing. Anderson showed that a network of distributed sensors are necessary to control the complex morphological shapes in hyper-redundant robotic systems, while also mitigating the sensor's effect on mobility. Within this context, let us define soft sensors:

> ***(Proprioceptive) soft sensors*** *are flexible measurements units embedded into the soft robotic body that through external stimuli measure the (local) changes of the system. Softness here implies that the sensor minimally alters the global mechanical behavior of the robot.*

Remark 2.2 As the emphasize lies on "minimally alters the global mechanical behavior", soft sensors may be composed of stiff (perhaps even rigid) components. Our definition infers that these sensors must be placed into or onto the soft body, minimally affecting the operational workspace of the soft actuator network in static or dynamic condition.

Soft robotics in the 80's. Following the fundamental works of McKibben, Scheinman and Anderson, the field of soft Pneumatic Artificial Muscles (PAMs) in robotics evolved rapidly in the early 80's. A few soft robotic systems are shown in Figure 2.5. Teleshev (1981, [94]) developed a soft gripper reminiscent of modern PneuNet actuators [47, 73, 152] – a rectangular bellow-shaped soft actuator. Unlike uniaxial PAMs, which are radially symmetric, these soft grippers explored an asymmetrical design of bellows. The geometry led to a stiffness differential around the circumference, resulting in their icon bending motion. Still popular today, these pneumatic bending actuators find their origin back in early 1974, see Andorf et al. (1974, [9]). A decade later, Takagi et al. ([205], 1983) developed a soft multi-



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

Figure 2.5. Early robotic systems that explored soft PAMs for various tasks. (a) Soft robotic grippers by Teleshev [94]. (b) The soft arm using *Rubbertuator* actuators by Takagi and Sakaguchi [205]. (c) Three-link soft robotic manipulator with gripper reminiscent of the elephant's trunk, developed by Wilson at Duke University [238, 239]. (d) Shadow bipedal walker by Buckley et al. [92] using McKibben muscle in antagonistic pairs to produce locomotion.

joint robot manipulator that resembles the human arm with its movements and antagonistic muscle pairs. Although, their PAMs – called *Rubbertuator* – had a function and design identical to McKibben's PAMs, their system showed the merits of combining soft and rigid. They observed not only a high-degree of positional control of the robot arm, force control was easily regulated by the pressure control. This naturally had safety benefits. The soft robot arm could perform delicate low-force tasks while simultaneously blocking motion when encountering a human. These (soft) properties were lacking in rigid robotic manipulators at the time but reminiscent in its biological counterpart – the human arm. Note that, at that time, force and impedance control for rigid robotics had been topics of academic research for years [7, 90, 91, 114], dating back to the early 1970's. (*e.g.*, see also [136]). Yet achieving similar properties without control were rarely explored at the time.

Shortly after, Wilson (1984, [239]) developed a soft robot manipulator based on the elephant's trunk at Duke University, Durham. His design effectively combined the works of Teleshev [94] and Takagi et al. [205] into a robot with similar dexterity but minimal use of rigid components. According to [238], his idea stemmed from the work of Kier and Smith (1985, [115]) who studied the biomechanics of muscular-hydrostats in animals, like cephalopods (*e.g.*, squids). The work of Kier et al. [115] studied how complex motions are produces in muscular organs, like elongation, shortening, bending and torsion. Inspired by the muscular hydrostat in the elephant's trunk, Wilson developed a soft arm composed of polyurethane

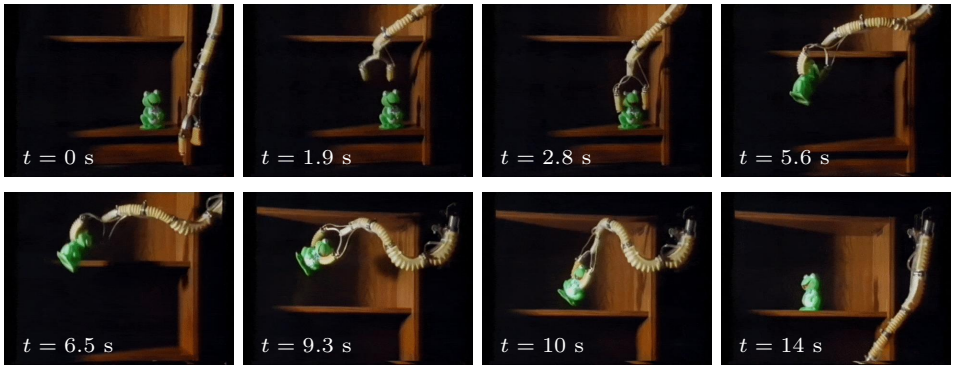


Figure 2.6. Three-link soft robotic manipulator with two-fingered soft gripper by James Wilson from Stanford university (1984, [239]). Unlike classic manipulators, where links and joints are separated, Wilson's robot consisted of three pneumatic bending actuators – being link and joint simultaneously.

tubes that work as half-bellows, which enabled expansion and bending under positive pressurization [238]. To accommodate for three-dimensional movement, each soft pneumatic link was placed at a $\phi = \frac{\pi}{2}$ twist offset w.r.t. to the previous link. To illustrate the motion of the soft arm, a few snapshots are provided in Figure 2.6. Wilson hypothesized that these highly-complaint robots will be more mechanically robust and be sufficiently dexterous for tight workspaces, contrary to its rigid counterparts. Although the dexterity was novel, the positional accuracy was poor. The main problem stemmed from the soft arm being controlled in open-loop (*i.e.*, remote tele-operation) without proprioceptive sensing nor any positional feedback control. An issue akin to the Orm (1965).

A few years later, Buckley et al. (1988, [92]) developed the Shadow walker – a bipedal rigid robot comprised of antagonistic McKibben muscle pairs. Although not fully soft, their system did explore proprioceptive sensing. The hip, knee, and ankle joints were equipped with resistance-variable potentiometers for position feedback, whereas all the muscles had tension sensors for force feedback. Later on, these resistive sensors were replaced by analog optical sensors to improve robustness [92]. Although the system was top-heavy, due to the pneumatic control hardware (*e.g.*, valves and piping), rudimentary locomotion was possible. Interestingly, a similar artificial muscle system is still explored in nowadays humanoid robotics, like the Atlas from Boston Dynamics. The success of pairing soft muscles with proprioceptive sensing eventually led to the development of the McKibben Shadow hand [81, 92], comprised of 40 uniquely addressable soft muscles.

Following, Suzumori and Saiko [203, 204] developed a micro flexible soft actuator driven by an electro-hydraulic system (intrinsic length $L \approx 12$ mm). Each

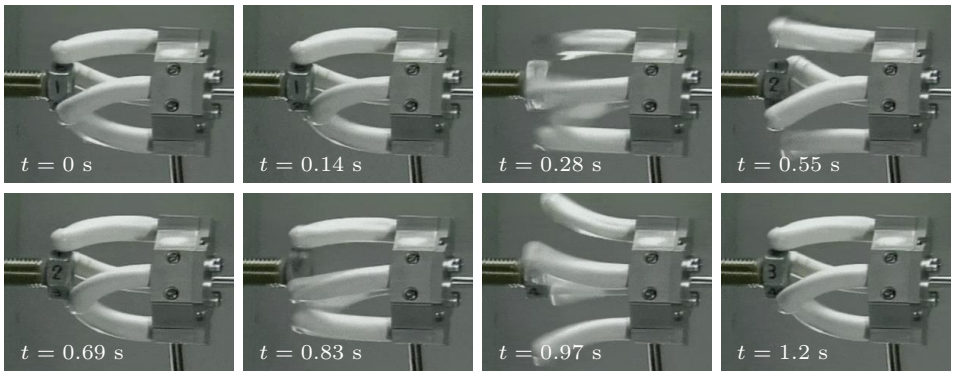


Figure 2.7. Four-fingered soft robotic gripper by Suzumori and Saiko (1989, [203, 204]). Each finger possess three pneumatic chambers that allow for directional bending – analogous the Orm. Through proper coordination of the set of soft fingers various gripping complexity can be achieved, such as the clock-wise turning a mechanical hex bolt (as shown above). Suzumori et al. showed these intricate finger motions can be easily achieved by careful modeling of the fingertip dynamics, and exploring the adaptability of soft materials.

end-effector enables three DOFs including pitch, yaw, and stretch – making it ideal for fingers, arms and legs. Figure 2.7 shows the level of dexterity in their system. By placing the four PAMs parallel on a gripper mount, and assigning a pre-defined trajectory, they showed their soft robotic system has sufficient dexterity to mount an hex-bolt at an incredible speed and precision. To achieve such dexterity and precision, Suzumori et al. [203] employed various modeling and control strategies to account for the dynamical characteristics under high-frequency, fluid compressibility, and the closing mechanics of pressure valves. Futhermore, the kinematics of each finger was derived using generalized homogenous transformation, not unlike traditional robotics. Knowing both the compliance characteristics (pressure-deformation relations) and the forward kinematics, a Jacobian-based positional controller was employed to regulate the Cartesian coordinates of the finger-tips (successfully one might add).

Early controllers for hyper-redundant (soft) robots. Following the increasing interest in highly-flexible robots around the late 80's, academic research into controlling these *hyper-redundant* robots boomed shortly after. At the time, the term hyper-redundancy – being an extension to redundancy in robotics [141] – was defined as the relative degree of kinematic and/or actuator redundancy which is large or even infinite [42, 46]. The term was first introduced by Chirikjian and Burdick (1989, [41]). Others referred to these robots as *highly redundant* [157, 241] or *High Degree-of-Freedom* (HDOF) manipulator [147, 179]. Around that time,

Chirikjian and Burdick provided a plenary of mathematical foundation [42–46] focussed on the kinematics and motion planning of hyper-redundant manipulators. Their work presented a modal discretization approach to describe the shape of the deformable backbone [45], and from this geometric approaches were introduced to solve obstacle avoiding trajectories using generalized *follow-the-leader* strategies [44]. Especially the latter showed the limitation in rigid redundant manipulators. Although Chirikjian laid the foundation for the control of hyper-redundant robot, basic principles of motion planning in pneumatic hyper-redundant robots were already presented by Wilson et al. (1988, [240, 241]) – yet were not called hyper-redundant robots nor soft robots yet. Recall also that the work of Wilson et al. has been sown earlier in Figure 2.1 and Figure 2.6. In Brock et al. (1991, [30]), a similar analysis was used for optimal shape design of thin elastic rods to realize desired robotic compliance. Besides, there exist an abundance of literature prior to [42] on Variable Geometry Truss Manipulators (VGTMs) – a variant of hyper-redundant tensegrity robotics – that dealt with motion planning for such systems [156, 157, 179]. Later, Mochiyama et al. [146, 147], built upon Chirikjian's work by extending it to a dynamic formulation for elastic rods such that classic controller design is possible. They proposed shape-regulation controllers for HDOF manipulator by projecting them onto time-invariant curves, thereby showing that the estimation the desired curve parameters is the crucial key to solving the problem by Lyapunov design [146]. Although there existed a variety of modeling and control strategies, computational power in relation to modeling complexity was the limiting factor for the simulation-to-reality transfer at the time.

II

Design Optimization for Soft Robots

3

Optimal Design of Soft Robots – a Gradient-based Approach

Abstract - In this chapter, we present a novel framework for synthesizing the design of pressure-driven soft robots. Contrary to traditional design methods, a topology optimization scheme is employed to find the optimal soft robotic structure given user-defined requirements. To our knowledge, the combination of pressure-driven topology optimization and soft robotics is, to date of this thesis, unexplored. Two difficulties are related to this problem. First, pressure-based topology optimization is challenging since the adaptive topology changes the pneumatic load at each optimization step. To deal with this issue, we exploit the facial connectivity in mesh tesselations to efficiently simulate the physics involving pneumatic actuation in soft robotics. The second issue is describing the hyper-elastic nature of soft materials. Here, nonlinear finite element is explored such that large deformations can be described accurately. Numerical investigation shows that the framework can produce meaningful and insight-full material layouts with little to no prior knowledge of soft robotic design. This framework does not only accelerates design convergence, but it could also extend to the development of new and unexplored soft robot morphologies.

This chapter is based on: B.J. Caasenbrood, A.Y. Pogromsky, and H. Nijmeijer. *A Computational Design Framework for Pressure-drivenSoft Robots through Nonlinear Topology Optimization.* IEEE International Conference on Soft Robotics (RoboSoft), 2020. doi: 10.1109/RoboSoft48309.2020.9116010

3.1 Introduction

The field of soft robotics has attracted the interest of many researchers from different backgrounds. Soft robots use compliant and hyper-elastic materials, while the use of rigid materials is minimized. The introduction of soft materials into robotics greatly expanded the field of application for robotics. For example, due to their dexterity and environmental robustness, soft robots are often used in medical applications [11, 167, 248], adaptive grasping [73**?**], and locomotion in uncertain environments [61]. Unlike its rigid counterpart, soft robots undergo large continuum-bodied motion that, to some extent, resembles morphologies found in nature. These morphologies arise by virtue of the low compliance in soft materials and, more importantly, the structural layout of the soft robot. As of today, many of the fundamental engineering principles in rigid robotics, like design, actuation, sensing, and control, are often not applicable to soft robotics systems. Since its inception, most of these engineering problems have remained challenging or unresolved.

Although the diversity in soft robotics is significant, ranging from adaptive grippers to soft manipulators, most topologies in soft robotics can be associated with nature or engineered geometries for minimal compliance (e.g., bellow shapes). Soft robots often mimic living creatures and their morphologies, e.g., the tentacle of an octopus [73, 237], or the trunk of an elephant [61]. Hypothetically, the abundance of bio-mimicry in soft robotics might be associated with the design complexity of developing robots from soft materials. The large number of degrees-of-freedom and exotic mechanical nature of soft robots makes design significantly challenging, and consequently, the design process can be iterative and time-consuming [237]. Therefore, it becomes potentially advantageous to use computational tools that assist or develop appropriate soft robotic topologies given a set of user-defined requirements, like desired motion or force.

In the past, researchers have made efforts to finding morphologies through mathematics, in particular through evolutionary algorithms. The concept of automated creature designs was first introduced by Sims [**?**], who showed that, given a set of basic geometries, locomotive organisms could be generated from evolutionary algorithms. These virtual organisms resembled biological morphologies to some extent; however, the complexity of the material layout was limited. More recent work involving the synthesis of virtual soft robots includes Cheney et al. [40], who successfully produced intricate locomotive morphologies using artificial neural networks and multi-material parameter spaces of active and passive soft voxels. Other work involving morphological synthesis includes [19**?** **?**]. Unfortunately, the synthesis of morphologies from previous approaches, though novel,

remains only in ideal simulated environments. An accurate representation of the nonlinear material properties in soft robotics can be challenging, and in favor of computational efficiency, little detail is spent on the nonlinear nature governing soft materials. Besides, these evolutionary frameworks typically involve a network of 'activation' cells or voxels that perform ideal volumetric deformation, biologically resembling muscle functionality while unfortunately lacking resemblance to conventional actuation in soft robotics, e.g., pneumatics, dielectrics, and smart metal alloys (SMA).

Reviewing previous methods, a more efficient approach for solving the optimal morphology might be founded in topology optimization. Topology optimization is the general formulation of a material distribution problem for mechanical solids, where density-based topologies arise throughout an iterative (non-convex) optimization procedure. The synthesis of compliant mechanisms through topology optimization is investigated thoroughly [**?** **?** **?**]; however, its application to soft robotics is relatively unexplored [**?** **?**]. Yet, to obtain meaningful topologies for soft robotics, two problems need to be addressed. Since soft robots undergo large deformations, it becomes necessary to describe the nonlinear geometrical deformations accurately. Inherent to significant deformation of soft materials is the importance of nonlinear material behavior, like hyperelasticity. Another concern is the design-dependency of the external forces, in our case, the pneumatic loads. This class of structural problems is more challenging than traditional problems since the load is continuously interacting with the adaptive interface during the iterative optimization process [**?** **?**]. It should be mentioned that the use of compressed air or pressurized fluid is a popular actuation approach in soft robotics.

In this work, we present a novel framework for generating topologies of soft robotics. Contrary to biometry or convectional designs, finding the (optimal) material layout of the soft robot is accomplished through a gradient-based nonlinear topology optimization, where the distribution of soft materials is optimized given a user-defined objective. Our main contributions include the description of nonlinear geometrical deformation and pneumatic loading. We exploit the connectivity properties in polygonal meshes such that synchronized volumetric contraction or expansion of a group of polygonal elements can artificially mimic the geometrical loads in pneumatic actuation. The advantages of our framework in comparison to other literature are: (*i*) a better representation of pneumatic actuation in soft robotics; (*ii*) improved design convergence in contrast to evolution-based optimization methods. To our knowledge, our approach of pressure-driven nonlinear topology optimization is new for soft robotics, and its application could easily extend to other soft robotic systems.

The remainder of the paper is structured as follows. In section 3.2, we will dis-

cuss the continuum mechanics for hyper-elastic materials, followed by a description of the optimization scheme for soft robotics. In section 3.4, we propose a numerical example for developing a soft robotic structure to illustrate the effectiveness of our approach.

3.2 Nonlinear Finite Elements

3.2.1 Strain theory in continuum mechanics

Here, we state the variational principle of continuum mechanics in a nonlinear geometrical setting. First, we introduce a description of the undeformed material domain described by $\mathcal{B}_0 \subset \mathbb{R}^3$. As external forces induce motion in $\mathcal{B}_0$, a representation of the deformed domain can be described by $\mathcal{B} \subset \mathbb{R}^3$. Suppose there exists an arbitrary material point inside the undeformed configuration represented by a position vector $\boldsymbol{X} \in \mathcal{B}_0$. Due to the motion of $\mathcal{B}_0$, a continuous path can be drawn between the position vectors $\boldsymbol{X}$ and $\boldsymbol{X}'$, that is, a particular material point in undeformed and deformed configuration, respectively. Therefore, let the deformation mapping for every material point in $\mathcal{B}_0$ be described by $\boldsymbol{\varphi} : \mathcal{B}_0 \to \mathcal{B}$ such that $\boldsymbol{X} \mapsto \boldsymbol{\varphi}(\boldsymbol{X}) = \boldsymbol{X}'$. Alternatively, we can write $\boldsymbol{X}' = \boldsymbol{\varphi}(\boldsymbol{X}) = \boldsymbol{X} + \boldsymbol{x}(\boldsymbol{X})$, where $\boldsymbol{x} \in \mathbb{R}^3$ is the displacement vector the material point Since the mapping $\boldsymbol{\varphi}$ is assumed to be sufficiently smooth, the second-order deformation gradient tensor is defined by

$$\boldsymbol{F} := \frac{\partial \boldsymbol{\varphi}}{\partial \boldsymbol{X}} = \boldsymbol{I} + \frac{\partial \boldsymbol{x}}{\partial \boldsymbol{X}}. \tag{3.1}$$

The second-order deformation gradient tensor holds useful information about the deformation locally, i.e., it describes the deformation of an infinitesimal sub-volume of the material domain $\mathcal{B}_0$ around $\boldsymbol{X}$. From the deformation gradient, the Green-Lagrange strain tensor can be derived by $\boldsymbol{\mathcal{E}} := \frac{1}{2}(\boldsymbol{\mathcal{C}} - \boldsymbol{I})$, where $\boldsymbol{\mathcal{C}} = \boldsymbol{F}^\top \boldsymbol{F}$ is the right Cauchy-Green strain tensor. Expressed in terms of displacement gradient, the general expression for the second-order Green-Lagrange strain tensor becomes

$$\boldsymbol{E} = \frac{1}{2}\left(\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{X}} + \frac{\partial \boldsymbol{x}^\top}{\partial \boldsymbol{X}} + \frac{\partial \boldsymbol{x}^\top}{\partial \boldsymbol{X}}\frac{\partial \boldsymbol{x}}{\partial \boldsymbol{X}}\right). \tag{3.2}$$

Since the numerical implementation generally involves matrix-vector notation instead of tensor notation; we briefly introduce the following notation. A Cartesian tensor can be formally represented by an component array in terms of a basis $\boldsymbol{e}_i$. For example, a second-order tensor can be denoted by $\boldsymbol{\mathcal{T}} = \sum_{ij} \boldsymbol{\mathcal{T}}_{ij}\, \boldsymbol{e}_i \otimes \boldsymbol{e}_j$ with $\otimes$ the dyadic product and bases $\boldsymbol{e}_1, \boldsymbol{e}_2, \boldsymbol{e}_3 \in \mathbb{R}^3$. As such, the column vector

representation of a second-order tensor $\boldsymbol{\mathcal{T}}$ can be denoted by $\{\boldsymbol{\mathcal{T}}\} := \text{vec}(\boldsymbol{\mathcal{T}}_{ij})$. Hence, the variational form of Lagrangian strain can be written in vector notation as

$$\{\delta \boldsymbol{E}\} = \left[\frac{\partial \boldsymbol{E}(\boldsymbol{x} + \varepsilon \cdot \delta \boldsymbol{x})}{\partial \boldsymbol{x}}\right]_{\varepsilon=0} \tag{3.3}$$

$$:= \boldsymbol{B}(\boldsymbol{x})\,\{\delta \boldsymbol{x}\}, \tag{3.4}$$

where $\boldsymbol{B}(\boldsymbol{x})$ is a nonlinear strain-displacement matrix that relates displacements to Lagrangian strain [117].

Remark 3.1 Two-dimensional problems In this work, we primarily focus on two-dimensional mechanical problems. We would like to stress that the variational principle for three-dimensional continuum solids discussed earlier are similar to those in two-dimensional situations [117**?**].

3.2.2 Isotropic hyperelasticity

In soft robotics, the use of elastomer materials is widespread due to their relatively low Young's moduli, large reversible strains, and mechanical robustness. These elastic materials are distinguished in material mechanics as hyperelastic materials. The presence of large deformations and rubber-like materials inherently leads to a state-dependent mechanical compliance. In contrast to Hookean materials, whose elasticity is linear, the constitutive behavior of hyperelastic materials is described by a (nonlinear) strain energy function $\Psi : \mathbb{R}^{3\times3} \mapsto \mathbb{R}_{\geq 0}$, i.e., a mapping from the Lagrangian strain tensor to potential energy. Popular constitutive models for hyperelastic behavior include Neo-Hookean, Mooney, Ogden, or Yeoh. In contrast to linear elasticity strain energy, the strain energy for hyperelastic constitutive models are commonly expressed in terms of the strain invariants (J_1, J_2, J_3).

In this work, the Yeoh constitutive model for hyperelasticity is used to describe the mechanics of soft materials. The Yeoh model is a popular constitutive model due to its unique dependency on the first invariant $J_1 = \text{tr}(\boldsymbol{C})$. The strain energy function of the Yeoh model [117] is given by

$$\Psi = \sum_{i=1}^{3} c_i (J_1 - 3)^i, \tag{3.5}$$

where $c_1 > 0$ and c_2, c_3 are material constants (J/m^3). The second Piola-Kirchoff stress of a constitutive material can be calculated by differentiating the strain energy function with respect to the Lagrangian strain tensor [117, 168]

$$\boldsymbol{\mathcal{S}} = \frac{\partial \Psi}{\partial \boldsymbol{E}} = 2\frac{\partial \Psi}{\partial \boldsymbol{C}}, \tag{3.6}$$

which is a symmetric second-order tensor that is the energy conjugate to the Lagrangian strain. Suppose that the external forces acting on the boundary of the material domain $\mathcal{B}_0$ can be represented by the vector $\boldsymbol{f}_{\text{ext}}$. Then, in case of a (quasi)-static equilibrium, the residual between the internal force of the continuum solids and the external forces can be written as

$$\boldsymbol{R}(\boldsymbol{x}) = \underbrace{\int_{\mathcal{B}_0} \boldsymbol{B}(\boldsymbol{x})^\top \{\boldsymbol{\mathcal{S}}(\boldsymbol{x})\}\, dV}_{\boldsymbol{f}_{\text{int}}} - \boldsymbol{f}_{\text{ext}} = 0, \tag{3.7}$$

where the first right-hand term represents a volume integral over the undeformed domain $\mathcal{B}_0$. Given the context of finite elements, this represents a set of nonlinear equalities with unknown nodal displacements $\boldsymbol{x}$. The solutions to these (highly) nonlinear equations can be found through numerical methods, like the Newton-Raphson method.

3.2.3 Polygonal mesh elements

For two-dimensional finite element problems, the three-node triangular and bilinear four-node quadrilateral elements are widely used, while the use of higher-order polygonal elements is still relatively scarce. Polygonal elements have some attractive features such as facial connectivity, regularity, and adaptability that suit complex material domains. This makes polygonal elements ideal for topology optimization since numerical issues such as checkerboard patterns and single-node connections are inherently alleviated [**?** **?**].

An efficient method of generating polygonal meshes is constructing a Voronoi tessellation from a set of seeding points, where each point can be associated with its respective Voronoi cell. For two-dimensional tessellations, consider a set of n number of unique seeding points $\mathcal{P} = \{\boldsymbol{p}_i\}_{i=1}^n$ that are uniformly distributed inside $\mathcal{B}_0 \subset \mathbb{R}^2$. Then, the corresponding (restricted) Voronoi tessellation is defined as the set of Voronoi cells that intersect $\mathcal{B}_0$, that is,

$$\mathcal{T}(\mathcal{P}) = \{\, \mathcal{V}(\boldsymbol{p}, \mathcal{P}) \cap \mathcal{B}_0 \;:\; \forall \boldsymbol{p} \in \mathcal{P} \,\}, \tag{3.8}$$

where $\mathcal{V}$ is the Voronoi cell from the point $\boldsymbol{p}_i$. Mathematically, the definition of a Voronoi cell for a point $\boldsymbol{p}_i \in \mathcal{P}$ is given by

$$\mathcal{V} = \left\{\, \boldsymbol{x} \in \mathbb{R}^2 \;:\; \Delta(\boldsymbol{x}, \boldsymbol{p}_i) \leq \Delta(\boldsymbol{x}, \boldsymbol{y}), \forall \boldsymbol{y} \in \mathcal{P} \backslash \{\boldsymbol{p}_i\} \,\right\}, \tag{3.9}$$

where $\Delta(\boldsymbol{x}, \boldsymbol{y}) = ||\boldsymbol{x} - \boldsymbol{y}||$ denotes the euclidean distance between two points $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^2$. Due to the convexity of euclidean distance function, the Voronoi cells $\mathcal{V}$ are intrinsically convex polygons. Clearly, the choice of the point set $\mathcal{P}$

is of paramount importance in generating the Voronoi tessellation. To obtain a uniformly distributed set, Lloyd's algorithm can be used to iteratively update a set of points that are initially chosen randomly. For more detail on the generation of polygonal meshes, we refer the reader to the work of Talischi et al. [**?**].

3.3 Nonlinear topology optimization

3.3.1 Solid Isotropic Material With Penalization (SIMP)

The Solid Isotropic Material With Penalization (SIMP) method is a popular material interpolation scheme used in topology optimization [**? ? ?**]. In the approach, each finite element $e \in \{1, 2, ..., n\}$ is assigned with a continuous density variable $\rho_e \in [0, 1]$ to indicate if an elemental volume is solid ($\rho_e = 1$) or void ($\rho_e = 0$). This leads to a real-valued density field representing the material distribution within a discretized domain $\mathcal{B}_0$, where the global material distribution is represented by the density vector $\boldsymbol{\rho} = [\, \rho_1,\, \rho_2, ...,\, \rho_n \,]^\top$. The density vector $\boldsymbol{\rho}$ contains the design variables of the optimization scheme. To relate these artificial densities to elasticity, the strain energy of the constitutive material model Ψ is then parameterized using $\boldsymbol{\rho}$. To improve numerical robustness, a modified SIMP approach is used in which the strain-energy function is artificially modified as follow

$$\Psi_e = [\, \varepsilon + (1 - \varepsilon)\, {\rho_e}^p \,]\, \Psi, \tag{3.10}$$

where $p > 1$ is the penalty factor for penalizing intermediate densities during the optimization process, and $0 < \varepsilon \ll 1$ is a lower-bound on the material densities. In this work, we choose $p = 4$ and $\varepsilon = 10^{-3}$. Given the artificial elasticity model above, it shall be clear that residual vector function in (3.7) now depends on the nodal displacements and the densities of the adaptive topology, $\boldsymbol{x}$ and $\boldsymbol{\rho}$, respectively.

3.3.2 Artificial pneumatic loads by exploring dilation

The most general principle of actuation in soft robotics involves pneumatic networks that are embedded in the elastic body. Within the context of topology optimization, however, describing internal pressure-based loads is difficult. Contrary to conventional optimization problems with static loads, in a pressure-based problem, the location, direction, and magnitude of the load changes concerning the material distribution at every optimization step. This inherently yields design-dependency in the global force vector. To resolve this difficulty, we exploit the

connectivity properties of polygonal tessellations to describe the physics involving pneumatics efficiently. More specifically, isolated regions of void polygonal elements will artificially mimic the geometrical loads in pneumatic actuation by volumetric expansion or contraction.

To identify void regions in the topology, we introduce the notion of a so-called 'virus' element whose purpose is to infect neighboring elements with a low elemental density (voids). These virus elements are chosen a-priori and invariant, similar to the input of a pneumatic network. If the elements adjacent to an infected element have a density lower than a specified threshold γ, that is, $\rho_e < \gamma$, they become infected, and its logic will spread to its adjacent neighborhood. Afterward, each infected element will be influenced by the same pneumatic load. Clearly, as the topological layout of the discretized domain $\mathcal{B}_0$ changes during the optimization procedure, the influence of the pneumatic load will vary accordingly. To better reflect the physics involving pneumatics, facial connectivity must be qualified for infection, where two elements must share a common edge to be adjacent.

A flood-fill algorithm is used [**?**] to find the affected region of a virus element efficiently. In case of irregular tessellations, like the Voronoi tessellation, the flood-fill algorithm requires an adjacency matrix $\boldsymbol{\Gamma}$, which contains information about the mesh connectivity. The adjacency matrix can be directly computed from the element-node-incidence matrix. Here, the incidence matrix $\boldsymbol{A} \in \mathbb{N}^{n \times m}$ is a sparse unit-matrix with n columns for each element and m rows for each node, where $\boldsymbol{A}_{ij} = 1$ iff node i is incident upon element j. Given the incidence matrix, the adjacency matrix $\boldsymbol{\Gamma} \in \mathbb{N}^{n \times n}$ can be computed by

$$\boldsymbol{\Gamma} = \boldsymbol{A}\boldsymbol{A}^{\top} - \mathrm{diag}(\boldsymbol{A}\boldsymbol{A}^{\top}), \tag{3.11}$$

where the matrix has non-zero entries $\boldsymbol{\Gamma}_{ij} = 1, 2$ iff elements i and j share a common node or edge, respectively. As mentioned earlier, the adjacency will be modified such that edge connectivity is required for adjacency. Secondly, all the rows and columns corresponding to elements that satisfy $\rho_e \geq \gamma$ will be set to zero to ensure those elements are unaffected by the flood-fill. The pseudo-code for the identification of the pneumatic region is provided in Algorithm 1.

From a mathematical perspective, suppose that Algorithm 1 can be described by a mapping $\phi : \mathbb{R}^n \mapsto \{\mathbb{N}\}$ such that the mapping $\phi(\boldsymbol{\rho})$ returns a set of elemental indices of infected elements $\mathcal{E} \subseteq \{1, 2, ..., n\}$ that undergo volumetric compression or expansion depending on boundary conditions. Then, the global force matrix $\boldsymbol{f}$ in (3.7) can be assembled from elemental force vectors of the elements belonging to the affected region of the virus element $\mathcal{E}$, that is,

$$\boldsymbol{f}(\boldsymbol{x}, \boldsymbol{\rho}) = \sum_{e \in \mathcal{E}} \tilde{p}_e \boldsymbol{W}_e \, \boldsymbol{t}_e, \tag{3.12}$$

Algorithm 1: Find elemental indices subjected to volumetric differential pressure loads

Input : Tessellation $\mathcal{T}$, set of virus elements $\mathcal{V}$, threshold γ, material density vector $\boldsymbol{\rho}$
Output: Set of elemental indices $\mathcal{E}$

construct $\boldsymbol{\Gamma} \leftarrow$ **BuildAdjacencyMatrix**$(\mathcal{T})$;
modify $\boldsymbol{\Gamma} \leftarrow$ **EdgeConnectivity**$(\boldsymbol{\Gamma})$;
find thresholds $\mathcal{X} = \{\, i \in [1, n] \mid \rho_i \geq \gamma \,\}$;
set zeros $\boldsymbol{\Gamma}_i = (\boldsymbol{\Gamma}^\top)_i = 0$ for all $i \in \mathcal{X}$;
initialize empty set $\mathcal{E} \leftarrow \emptyset$;
for $i = 1 : numVirus$ **do**
  $\mathcal{I} \leftarrow$ **doFloodFill**$(\boldsymbol{\Gamma}, \mathcal{V}_i)$;
  $\mathcal{E} \leftarrow \mathcal{E} \cup \mathcal{I}$;
end

where $\tilde{p}_e$ is a dimensionless parameter representing the magnitude and direction of artificial pneumatic loading, $\boldsymbol{W}_e$ a diagonal weighting matrix of densities at nodal level and $\boldsymbol{t}_e = \int_{\mathcal{V}_e} \boldsymbol{B}^\top \boldsymbol{D}_e \boldsymbol{\epsilon}_v \, dV$ the elemental force vector with $\boldsymbol{\epsilon}_v = [1, 1, 0]^\top$ the volumetric strain and $\boldsymbol{D}_e$ the fourth-order constitutive elasticity tensor in matrix notation [168]. We stress that the global force vector in (3.12) represents an artificial pneumatic load, since this method assumes that the pressure loads can be emulated by anisotropic volumetric change of the polygonal elements. The magnitude and direction of this artificial pneumatic load can be controlled by varying the dimensionless parameter $\tilde{p}_e$.

3.3.3 Optimization problem

Given the notion of morphology in soft robotics, which is similar to compliant mechanisms, the general objective is to maximize the output displacement $\boldsymbol{u}_{out}$ on a virtual workpiece modeled by an artificial stiffness $k_{out} > 0$ [**?** **?**]. The objective function or the (desired) output displacement can be expressed by $\Phi = \boldsymbol{L}^\top \boldsymbol{x}(\boldsymbol{\rho})$, where $\boldsymbol{L} \in \mathbb{R}^{2n}$ a sparse unit-vector composed of nonzero entries for the degrees-of-freedom corresponding to the desired morphology of the soft robot. Given this description of directional output displacement, the topology optimization problem

for soft robots can be formulated as

$$\begin{aligned} \max_{\boldsymbol{\rho}} \quad & \Phi = \boldsymbol{L}^\top \boldsymbol{x}(\boldsymbol{\rho}), \\ \text{s.t.} \quad & c := \boldsymbol{R}(\boldsymbol{x}, \boldsymbol{\rho}) \; = \; 0, \\ & g := \boldsymbol{v}^\top \boldsymbol{\rho} \leq V^*, \\ & \boldsymbol{\rho} \in \mathcal{P}, \end{aligned} \tag{3.13}$$

where $\boldsymbol{R} \in \mathbb{R}^{2n}$ the global residual force vector in its equilibrium state, $\boldsymbol{v} \in \mathbb{R}^m$ a constant vector of relative elemental volumes, $V^* \in \mathbb{R}_{\geq 0}$ the maximum material infill, and $\mathcal{P} = \{\boldsymbol{\rho} \in \mathbb{R}^n \mid 0 \leq \rho_e \leq 1 \; \forall e \in [1, n]\}$ a set of feasible material densities that ensure numerical robustness.

3

3.3.4 Solver and sensitivity analysis

In this section, we discuss the use of a gradient-based optimization solver for the synthesis of the soft robot. Herein, we use the Method of Moving Asymptotes (MMA) proposed by Svanberg [**?**]. The MMA is similar to other nonlinear programming approaches, like Optimality Criteria (OC) and Sequential Quadratic Programming (SQP), in that it finds an optimal solution to a nonlinear non-convex optimization problem with inequality constraints. The MMA solves a sequence of sub-problems, i.e., convex approximations of the true problem, which are constructed from gradient-based information of the objective function Φ and their constraints c and g. The sensitivity of the inequality constraint in (3.13) is $\partial g / \partial \boldsymbol{\rho} = \boldsymbol{v}$ since we assume that the elemental volume $\boldsymbol{v}$ is constant. The sensitivity of the objective function is less trivial due to its dependency on the nodal displacements $\boldsymbol{x}(\boldsymbol{\rho})$. Since it is computationally expensive to obtain the displacements $\boldsymbol{x}$ through the Newton-Raphson method, it becomes beneficial to avoid the computation of their sensitivities. Thus, the sensitivities are computed through the adjoint method in which an augmented objective function is considered

$$\Phi = \boldsymbol{L}^\top \boldsymbol{x} - \boldsymbol{\lambda}^\top \boldsymbol{R}, \tag{3.14}$$

where $\boldsymbol{\lambda} \in \mathbb{R}^{2n}$ is a constant vector referred to as the adjoint vector. Note that, in case of equilibrium (i.e., $\boldsymbol{R} = 0$), the global residual forces are equivalent to zero; therefore, the adjoint vector can be chosen freely. For the sake of brevity, we denote differentiations by $\partial(\cdot)/\partial x := (\cdot)_{,x}$. Differentiations of the objective function with respect to the elemental densities ρ_e for each finite element $e \in \{1, 2, ..., n\}$ yields

$$\begin{aligned} \frac{\Phi}{\partial \rho_e} &= \boldsymbol{L}^\top \frac{\boldsymbol{x}}{\partial \rho_e} - \boldsymbol{\lambda}^\top \left(\frac{\partial \boldsymbol{R}}{\partial \boldsymbol{x}} \frac{\partial \boldsymbol{x}}{\partial \rho_e} + \frac{\partial \boldsymbol{R}}{\partial \rho_e} \right), \\ &= \left(\boldsymbol{L}^\top - \boldsymbol{\lambda}^\top \boldsymbol{K}_T \right) \frac{\boldsymbol{x}}{\partial \rho_e} - \boldsymbol{\lambda}^\top \frac{\partial \boldsymbol{R}}{\partial \rho_e}, \end{aligned} \tag{3.15}$$

Algorithm 2: Computational design algorithm for pneumatic soft robots.

Input : Domain $\mathcal{B}_0$, material Ψ, initial $\boldsymbol{\rho}^{(0)}$, infill V^*, virus $\mathcal{V}$, output $\boldsymbol{L}$, pressure $\tilde{p}_e$

Output: Optimal soft robot topology $\boldsymbol{\rho}^*$

construct tessellation $\mathcal{T} \leftarrow$ **VoronoiMesher**$(\mathcal{B}_0)$;
while *convergence* $\neq 1$ **do**
  update artificial material Ψ_e using (3.10) ;
  find infected set $\mathcal{E}$ using **Algorithm 1** ;
  build residual forces $\boldsymbol{R}$ using (3.7) and (3.12);
  solve displacements $\boldsymbol{x} \leftarrow$ **NewtonRaphson**$(\boldsymbol{R})$;
  evaluate $g_{,\boldsymbol{\rho}} \leftarrow \partial g / \partial \rho_e$;
  evaluate $\Phi_{,\boldsymbol{\rho}} \leftarrow \partial \Phi / \partial \rho_e$ using (3.16) ;
  update $\boldsymbol{\rho} \leftarrow$ **MMA**$(\Phi_{,\boldsymbol{\rho}}, g_{,\boldsymbol{\rho}}, \boldsymbol{\rho})$;
end

where the Jacobian of the residual force vector is substituted by the tangent stiffness matrix $\boldsymbol{K}_T = \partial \boldsymbol{R} / \partial \boldsymbol{x}$ (see [117]). By choosing the adjoint vector $\boldsymbol{\lambda}^\top = \boldsymbol{L}^\top (\boldsymbol{K}_T)^{-1}$, the terms involving $\boldsymbol{x}_{,\rho_e}$ can be eliminated and thus computation of the gradient becomes feasible. Therefore, the gradient of the objective function can be compactly written as

$$\frac{\partial \Phi}{\partial \rho_e} \overset{(3.7)}{=} -\boldsymbol{L}^\top (\boldsymbol{K}_T)^{-1} \left(\int_{\mathcal{B}_0} \boldsymbol{B}^\top \frac{\partial \boldsymbol{\mathcal{S}}}{\partial \rho_e} \, dV - \frac{\partial \boldsymbol{f}}{\partial \rho_e} \right). \tag{3.16}$$

However, it shall be clear that the derivation of the loading sensitivity $\boldsymbol{f}_{,\rho_e}$ is not straightforward since the global force vector is constructed from nested functions of logic operations (e.g., flood-fill). Therefore, the loading sensitivity is derived numerically using the forward difference method. To reduce computation time, we propose to only compute the sensitivities of the elements at the boundary of the pressure region, since the gradient of the pneumatic region is largest near the boundary of the infected set $\mathcal{E}$. Conclusively, a brief overview of the computational design algorithm for synthesizing pressure-driven soft robots is provided in pseudo-code in Algorithm 2.

3.4 Numerical Examples

As an illustrative example of the proposed computational algorithm, we synthesize a topology of a soft robot undergoing bending motion. This morphology is often associated with a popular class of soft robots named 'PneuNet' actuators

[73**?** **?**]. PneuNet actuators consist of a set of rectangular pneumatic chambers inside an elastomer medium. When pressurized, these chambers inflate, and the elastomer structure undergoes bending. The goal of this numerical study case is to synthesize a soft robot topology that undergoes bending motion similar to the PneuNet actuator.

In our example, the following settings are chosen. We consider a rectangular design domain $\mathcal{B}_0$ with dimensions 20×40 mm (as shown in Fig. 3.1). The maximum material infill is $V^* = 0.3$. The bottom left corner is fixed, and the right bottom corner is equipped with an artificial spring $k_{out} = 1.0$ N/mm. The objective of the optimization is to maximize the vertical displacement of the left bottom corner in the direction $\boldsymbol{u}_{out}$. It shall be clear that the entries of the vector $\boldsymbol{L}$ are non-zero for the corresponding nodal degrees-of-freedom. The Yeoh parameters are $c_1 = 36$ kPa, $c_2 = 0.25$ kPa, and $c_3 = 0.023$ kPa modelled after the silicone elastomer Dragonskin 10A. Concerning the pneumatic actuation, the adaptive topology is subjected to an artificial pressure parameter $\tilde{p}_e = 0.01$, where the virus element is located at the center of the material domain $\mathcal{B}_0$.

Using the method described in Algorithm 2, an optimized topology is obtained (as shown in Fig. 3.1). As can be seen, the computational algorithm provides a new and interesting variation on the well-familiar PneuNet actuator. Contrary to its rectangular predecessor, the optimal structure has a significant resemblance to a bellow-shaped actuator to accommodate bending mobility further. To validate our synthesized topology, a three-dimensional finite element analysis is conducted. First, we use Gaussian radial basis functions (RBFs) to reconstruct a smooth manifold surface from the discretized optimization mesh. Due to spatial symmetry, the spatial reconstruction can be horizontally repeated to construct a full 'PneuNet' actuator (see Fig. **??**). The two-dimensional geometry is then imported into CAD and converted to three-dimensional geometry with enclosed pneumatic chambers. The pneumatic chambers are subjected to a positive differential pressure of $\Delta p = 15$ kPa. As can be illustrated by Fig. **??**, the finite element analysis verifies that the synthesized soft robot topology accomplishes the desired bending morphology when pressurized.

Figure 3.1. Topology optimization results for a soft bending actuator within the design domain of 20 × 40 mm. The Yeoh material parameters are $c_1 = 36$ kPa, $c_2 = 0.25$ kPa, and $c_3 = 0.023$ kPa; and $V^* = 0.3$.

3.5 Conclusion

In this chapter, we present a novel framework for synthesizing soft robot topologies from hyperelastic soft materials. The design synthesis of classic soft robots is challenging due to material nonlinearities and the numerical implementation of pressure-driven loads in a topology optimization framework. We proposed a nonlinear finite element method for polygonal elements to solve these problems. Compared to previous research, our numerical approach presents a better representation of the physical nature of soft robotics, in particular, the hyperelasticity and the pneumatic actuation. Numerical investigation shows that our framework can produce meaningful and insight-full material layouts when developing a pressure-driven soft robot from soft materials. Theoretically, the proposed framework can be adapted to also express other actuation principles in soft robotics, such as dielectric polymers or thermal expansion; however, minor changes are required.

III

Model-based Control for Soft Robots

4

Dynamic Modeling – The Constant Strain Approach

Abstract - In this chapter, we derive continuum dynamic models for pneumatic soft robot manipulators through the differential geometry of spatial curves. These models are then related to Finite Element (FE) model to capture the intrinsic geometric and material nonlinearities. To accelerate numerical simulation, a reduced-order integration scheme is introduced to compute the dynamic Lagrangian matrices efficiently. This, in turn, allows for high-speed and (multi-link) dynamic models for soft manipulators with a minimal sacrifice in numerical precision. By exploring passivity ideas and a linear parametrization of hyper-elastic material coefficients, we propose a passivity-based adaptive controller that enhances robustness towards material uncertainty and unmodelled dynamics – slowly improving their estimates online. As a study case, a fully 3D-printed soft robot manipulator is developed, which shows good correspondence with the dynamic model under various conditions, *e.g.*, natural oscillations, forced inputs, and subjected to external disturbances like tip-loads. The solidity of the approach is demonstrated through extensive simulations, numerical benchmarks, and experimental validations.

This chapter is based on: B.J. Caasenbrood, A.Y. Pogromsky, and H. Nijmeijer. *Control-oriented Models for Hyper-elastic Soft Robots through Differential Geometry of Curves.* Soft Robotics, 2022. doi: 10.1089/soro.2021.0035.

4.1 Introduction

Traditional robots are made from rigid and dense materials that ensure accurate and repeatable motions. While rigid robotics excel at fast and precise motion, their structural rigidity lacks the compliance and mechanical robustness needed for safe and passive interaction in an unknown environment. Soft robotics, on the other hand, aim to improve the motion complexity and environmental robustness that is generally lacking its rigid counterpart. To further promote these topics in robotics, researchers aim to mimic living creatures by developing bio-inspired robots with similar morphologies and mechanical properties [65, 79, 80, 122, 134, 203]. The hyper-flexible and continuum-bodied structure in soft robots provides them with a rich family of motion primitives. Besides bio-mimicry, soft robotics has proven to be a prominent alternative for rigid robotics with a variety of applications, *e.g.*, manipulation and adaptive grasping [73], untethered locomotion and exploration through uncertain environments [47, 134, 165], rehabilitation [167], and even minimal-invasive surgery [49, 129]. Although the popularity of the field has increased exponentially in recent years, one of the first soft robots dates back already to the early 1990s, *e.g.*, the work of Suzumori et al. [203]. Yet, despite years of soft robotics research, their intrinsic hyper-flexible nature still possesses numerous challenges on modeling and control.

One major challenge in modeling is that the soft robot's elastic body undergoes large, continuous deformation. Since its inception, numerous works have addressed the kinematics for soft continuum robots [107, 147, 148]; yet, its original framework stems from hyper-redundant robotics nearly a decade earlier [46]. Similar to soft robots, hyper-redundant robots exploit their high joint redundancy to achieve a broader range of tasks (*e.g.*, shape control and collision avoidance) besides end-effector manipulation. To some extent, soft robotics can be seen as the successor to hyper-redundant robotics in which rigid mechanical joints or links are substituted with hyper-flexible soft elements. As such, the resulting dynamics involves one continuous deformable inertial body rather than a set of interconnected rigid bodies. As such, conventional modeling approaches cannot be applied directly to these continuously deformable robots, stressing the importance of novel modeling strategies. In this respect, the dynamics of a continuously deformable soft robot are of infinite-dimensional nature. This paradigm shift has further emphasized the challenges in control-oriented modeling of soft robots, as their physical description are often more suited for a Partial-Differential Equations (PDEs) rather than Ordinary Differential Equations (ODEs).

Recently, some significant steps have been made toward formulating reduced-order ODE models for elastic continuum soft robots, paving a path toward model-

based controllers. Perhaps one of the most popular techniques of spatial reduction is the so-called *"Piece-wise Constant Curvature"* model – PCC for short. The PCC model assumes that a soft robot's reachability can be described using a number of spatially-constant curves, which are parameterized using a minimal set of generalized coordinates. Although PCC models can be seen as a significant over-simplification of true continuum mechanics at hand, and is only applicable within some conservative conditions on soft robots (*e.g.*, elasticity dominates gravity), these models have proven to be remarkably viable for various control applications. Besides its use in inverse kinematic control [107, 134, 135], PCC models have also shown to be suitable for feedforward controllers as demonstrated by Falkenhahn et al. [65]; and more recently, closed-loop feedback controllers by Della Santina et al. [60, 110]. Although the aforementioned works utilize the lumped-mass description, others have employed PCC models with uniform mass distribution [79, 80, 170, 207, 208] and current models even extend beyond the constant curvature [46, 60, 148]. However, in the face of significant external loading or (distributed) contact with the environment, the PCC assumption is relatively conservative and leads to undesired kinematic constraints on the continuum deformation. Besides, these models often need additional identification to model compliance as they do not originate from a continuum mechanical framework.

On the other hand, Finite-Element Method (FEM) models do originate from continuum mechanics and, due to their PDE description, provide a more accurate representation of deformations; and are particularly suited to deal with geometric and material nonlinearities. Duriez et al.[62] and related works [51, 82, 124] showed that reduced-order FEM models could play an important role in closed-loop control – allowing accurate volumetric deformation and hyper-elastic behavior. Although such real-time simulations for FEM-based models are possible, a significant state-reduction is required to ensure sufficient computational speed. In the process, FEM-based models often lose desirable control properties, *e.g.*, passivity preservation, which might play an important role in control. An alternative modeling strategy is the recently emerging geometrically-exact Cosserat-beam model. Similar to the PCC models, the Cosserat models benefit from their Lagrangian model structure – the basis for robotics control theory. Rooted in a geometric method for describing the continuum mechanics using Lie theory proposed by Simo et al. [188], Boyer et al. [27, 28] proposed a geometrically-exact modeling framework for Cosserat beams using nonlinear parametrization of the strain field. Other examples include the work of Renda et al. [169, 170], providing various options for Piecewise-Constant Strain (PCS) and Variable Strain modeling approaches. Although recent variants of the Cosserat models offer good computational performance [84, 220], its use in model-based control is slowly upcoming.

In this respect, the topic of reduced-order modeling of soft robots is an active area of research. Yet, a challenge that is frequently overlooked in control-oriented research is the anisotropic material behavior, mechanical saturation, and more importantly, the nonlinear and possibly time-varying nature of the highly hyper-elastic soft materials [65, 148, 208, 220]. This is further amplified by the fact that soft robots are known for their diversity in elastic materials and corresponding morphologies. Mustaza et al. [155] proposed a modified nonlinear Kelvin-Voigt material model to embody the complex material behavior of silicone-composite manipulators (so-called STIFF-FLOP actuators). A similar silicone composite actuator was experimentally validated by Sadati et al.[178] who proposed a novel modeling approach with an appendage-dependent Hookean model and viscous power-law to describe nonlinear and time-dependent material effects, respectively. Both nonlinear material models show good correspondence with physical soft robots under various dynamic conditions, yet they lack general transferability to the soft robots with different geometries – intrinsically captured by FEM-driven models. As of today, there are few control-oriented models that both offer geometry and material versatilely similar to FEM models and the control convenience similar to spatial curve models.

Ultimately, the strong nonlinearities paired with its continuous nature encourage the use of model-based controllers. Nevertheless, regarding the aforementioned model-based control approaches [60, 65, 110], the stability and performance of the closed-loop system could be undermined by uncertainties in physical parameters or unmodelled dynamics. Particularly for state-feedback linearization (e.g., inverse dynamic), as the inversion of inaccurately estimated systems could lead to poor performance and even instability. Adaptive control [150, 192] or energy-based controllers [159] might offer the needed robustness towards material uncertainties and unmodelled dynamics. Unfortunately, up till now, the applicability of adaptive and energy-based control techniques on soft robotics is scarcely explored. Franco et al. [70] used an adaptive energy-based controller that compensates for external disturbances on the end-effector, yet this controller can be extended to include various slowly-varying material uncertainties, *e.g.*, hyper-elasticity and viscosity.

The contributions here are two-fold. First, to derive a finite-dimensional approximate of a soft continuum manipulator, where we briefly recapitulate existing modeling techniques for soft robot manipulators. To address the issue of infinite-dimensionality, we explore the PCC condition that allows for a low-dimensional description of the continuum dynamics. Although such modeling approaches have been thoroughly developed, we will address two issues that will aid the development of model-based controllers. We aim to bridge the gap between the PCC model and the underlying continuum mechanics by matching the quasi-static be-

havior to a Finite-Element-driven model (FEM), and we propose a reduced-order integration scheme using Matrix-Differential Equations (MDEs) to compute the spatio-temporal dynamics in real-time. Preliminary results were shown in Caasenbrood et al. [34, 35].

Second, in regards to the FEM-based hyper-elastic modeling and the possible presence of unmodelled dynamics (e.g., material uncertainties or external loads on the end-effector), a passivity-based adaptive controller is proposed that enhances robustness towards material uncertainties and unmodelled dynamics in closed-loop, slowly improving their estimates online.

4.2 Pneumatic soft continuum manipulator

By using additive manufacturing, we developed a soft and flexible robot manipulator that is suitable for pick-and-place applications. The 3-DOF soft manipulator can be seen in Figure 4.1. The soft manipulators's design is reminiscent of the Orm robot developed by Scheinman [75], the composition of the network of soft actuators is loosely inspired by the elephant's trunk consisting mainly of parallel muscles without skeletal support. The anatomy of the elephant's trunk provides an excellent study case, as it naturally exhibit continuum-body bending and moderate elongation [65, 107, 208]. The design is also similar to other soft robotic systems [61, 65, 203] that all undergo three-dimensional movement by inflation or deflation of an embedded pneumatic bellow network. The pneumatic network has three independent inputs, which are labeled $\boldsymbol{u} = (u_1, u_2, u_3)^\top$. By varying these input, the soft manipulator can achieve bending in any preferred direction by differential pressurization of each channel (<0.1 MPa), *e.g.*, $u_1 > u_2 = u_3 > 0$. Whereas, simultaneous pressurization accomplishes moderate elongation, *i.e.*, $u_1 = u_2 = u_3$. As a demonstration, we provided the following pressure inputs to the system:

$$u_i(t) = \begin{cases} \text{erf}(t) \cdot [P_0 - P_a \sin(\pi t + \delta)] & \text{for } i = 1, 2 \\ \text{erf}(t) \cdot [P_0 - P_a \sin(\pi t)] & \text{for } i = 3 \end{cases} \tag{4.1}$$

where $P_0 = 5$ kPa, $P_a = 25$ kPa, $\delta = \frac{\pi}{2}$, and $\text{erf}(t) := \frac{2}{\pi} \int_0^\tau \exp(-\tau^2)\, d\tau$ the error function to ensure a smooth transient. The demonstration is shown in Figure 4.1.

Remark 4.1 (Additive manufacturing) The soft manipulator is exclusively composed of a printable, flexible thermoplastic elastomer (Young's modulus $\leq$ 80 MPa), which intrinsically promotes softness and dexterity. The elastomer material is developed explicitly for Selective Laser Sintering (SLS), a 3-Dimensional (3D) printing method that uses a laser to solidify powdered material. The main advantage of SLS printing over other techniques is that the printed parts are fully self-

supported, which allows for highly complex and high-detail structures. It should be mentioned, though, that the layer-by-layer material deposition will introduce undesired anisotropic mechanical effects. To mitigate anisotropy, the bellows are printed orthogonal to the printing plane, thereby ensuring mechanical symmetry. For the majority of this work, the 3D-printed soft robot in Figure 4.1 will form the basis of the dynamical model. The 3D model is made publicly available at the *Sorotoki* software repository at `github.com/BJCaasenbrood/SorotokiCode` [34].

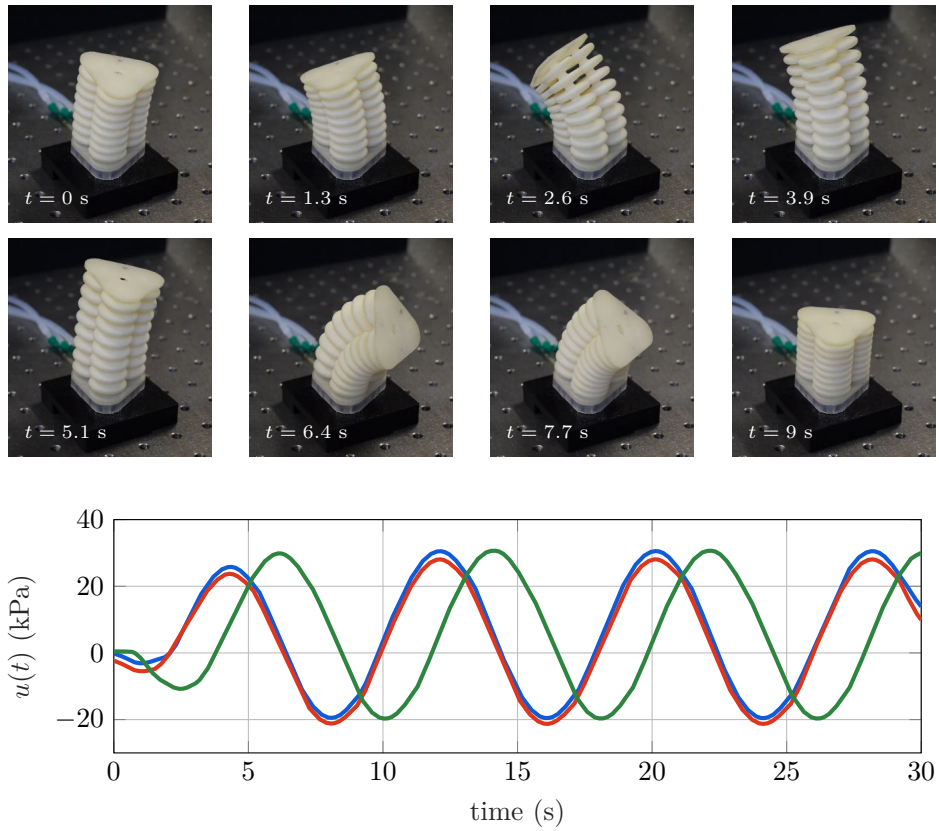


Figure 4.1. (top) Soft robot manipulator with three parallel embedded pneumatic bellows. This manipulator changes its posture by inflation and deflation of an embedded pneumatic network (<0.1 MPa). (bottom) Differential pressure signals applied on the internal bellows structures given by the input vector $\boldsymbol{u} = (u_1\, u_2,\, u_3)^\top$, shown by the trajectories (—, —, —), respectively.

4.3 Generalized models for soft manipulators

As mentioned previously, soft robots are composed of elastic bodies that can be modeled as a dynamically deformable continuum with of infinite-dimensional nature. In this section, we aim to derive a compact and computationally efficient model that envelops the continuous dynamics of the soft manipulator in Figure 4.1 (and soft robotic systems of similar topology, *e.g.*, [65, 75, 109]) through a small set of generalized coordinates. We denote these coordinates by $\boldsymbol{q} \in \mathcal{Q}$. Their respective velocities are denoted by $\dot{\boldsymbol{q}} \in T_{\boldsymbol{q}}\mathcal{Q}$ which belong to the tangent space of the configuration manifold $\mathcal{Q} \subseteq \mathbb{R}^n$. Let it be clear that the choice on $\boldsymbol{q}$ is free. However, finding a choice that minimizes $n = \dim(\boldsymbol{q})$ and that accurately reflects the continuum nature can be challenging. A state representation that satisfies both will help to keep computational cost low which is the current bottleneck for model-based control of soft robots.

We base our modeling framework on the work of Mochiyama et al. (2003, [148]), who outlined a theoretical foundation for continuum manipulators. Their work is extended upon by including extensibility, serial-chaining of multiple soft links, pneumatic actuation, and the introduction of nonlinear and time-dependent material behavior. Earlier modeling strategies addressing similar issues can be found in from Godage et al. (2016, [79, 80]), Della Santina et al. (2020, [58–60]), Renda et al. (2018, [170]), and Boyer et al. (2021, [27]). Leveraging from the aforementioned works, a finite-dimensional approximation of the true soft robot dynamics with can be written in the familiar Lagrangian form:

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{h}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{J}^{\top}(\boldsymbol{q})\boldsymbol{\lambda} + \boldsymbol{\tau}(\boldsymbol{q}, \boldsymbol{u}), \tag{4.2}$$

$$\boldsymbol{\tau} = \boldsymbol{G}^{\top}(\boldsymbol{q})\boldsymbol{u}, \tag{4.3}$$

where $\boldsymbol{M}(\boldsymbol{q}) \in \mathbb{R}^{n\times n}$ denotes the generalized inertia matrix, $\boldsymbol{h}(\boldsymbol{q}, \dot{\boldsymbol{q}}) \in \mathbb{R}^n$ a vector of nonlinear state-dependent force contributions. The nonlinear state-dependent contributions possess a structures as follows: $\boldsymbol{h}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{f}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ given by the Coriolis forces and visco-elastic terms, respectively.

Assumption 4.1 (Finiteness generalized inertia) The generalized inertia matrix is a positive definite symmetric matrix that is bounded from both sides $\lambda^{-} \preceq \boldsymbol{M}(\boldsymbol{q}) \preceq \lambda^{+}$ for all configurations $\boldsymbol{q}$, where λ^{-}, λ^{+} are positive scalars.

Assumption 4.2 (Passivity) For any velocity $\dot{\boldsymbol{q}}$, it holds that $\dot{\boldsymbol{q}}^{\top}\left(\dot{\boldsymbol{M}} - 2\boldsymbol{C}\right)\dot{\boldsymbol{q}} = 0$ – the so-called passivity condition for Lagrangian systems. If the condition holds, it can easily be shown that map $\boldsymbol{u} \mapsto \dot{\boldsymbol{q}}$ is passive, which implies that there exist a constant $\beta \geq 0$, such that the energy produced by the system E^{u} is bounded from

below [159]:

$$E^{\mathrm{u}} := \int_0^T \dot{\boldsymbol{q}}^\top(\tau)\boldsymbol{u}(\tau)\, d\tau > -\beta \quad \forall T > 0. \tag{4.4}$$

Assumption 4.3 (Under-actuation) In many cases, a soft robot that falls under the category hyper-redundant is also intrinsically under-actuated. Mathematically, under-actuation is defined as follows [210]. A second-order system $\ddot{\boldsymbol{q}} = \boldsymbol{f}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \boldsymbol{u}, t)$ is fully-actuated if, for any time t and state $(\boldsymbol{q}, \dot{\boldsymbol{q}})$, the flow map $\boldsymbol{f}$ is surjective. In laymen's terms, for any acceleration $\ddot{\boldsymbol{q}}$ there is exists a unique input $\boldsymbol{u}$ that produces such response. Otherwise, the system is under-actuated. Given the control affine structure in (4.3), the system is under-actuated if configurations $\boldsymbol{q} \in \mathcal{Q}$ exist such that $\mathrm{rank}\,(\boldsymbol{G}(\boldsymbol{q})) < \dim(\boldsymbol{q})$. Let it be clear that fully-actuated systems are dramatically easier to control than under-actuated systems. However, for the sake of simplicity at this stage, we assume the actuation matrix to be full rank and time-invariant, *i.e.*, $\boldsymbol{G}(\boldsymbol{q}) \simeq \boldsymbol{G}$. Under-actuation will be treated further in Chapter 4 and is not considered here in Chapter 3 .

In this chapter, a similar modeling framework is adopted to [148]; however, we propose an extension to incorporate FEM-driven data to more accurately reflect the underlying continuum mechanics – in particular, hyper-elasticity and visco-elastic creep. We also propose a numerical scheme that significantly accelerates the computation of the continuous dynamics.

4.3.1 Piecewise curve kinematics

To represent the hyper-flexible configuration of the soft robot, consider a smooth spatial curve that passes through the geometric center of the continuously deformable body, as shown in Figure 4.2. In literature, this curve is called the *backbone curve* as it simplifies the three-dimensional deformation imposed by distributed forces acting on the elastic body. The arc-length of the backbone corresponds to the extensible length of the soft robot denoted by the variable $l(t)$ which we assume bounded $l_- \leq l \leq l_+$, and let L be a constant denoting the total unstressed length of the soft robot. Next, let us introduce a spatial variable $\sigma \in \mathbb{X}$ that belongs to the one-dimensional material domain of the backbone curve, *i.e.*, $\mathbb{X} = [0,\, L]$. Let it be clear that the spatial variable σ represents the arc-length of a material coordinate along with the unstressed material domain of the soft robot manipulator.

Given each material coordinate, we wish to find a suitable low-dimensional coordinate representation $\boldsymbol{q}$ such that the position vector ${}^0\boldsymbol{\gamma}(\sigma, \boldsymbol{q})$ anywhere on the continuous backbone can be written as a mapping from generalized coordinates

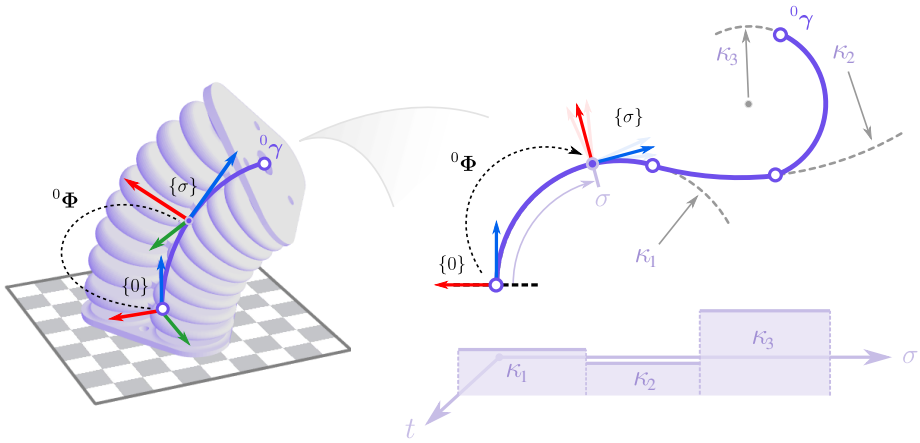


Figure 4.2. Schematic representation of the Piece-wise Constant Curvature (PCC) description for general soft manipulator systems, given by a parameterized curve ${}^0\boldsymbol{\gamma} : \mathbb{X} \times \mathcal{Q} \to \mathbb{R}^3$ and orientation matrix ${}^0\boldsymbol{\Phi} : \mathbb{X} \times \mathcal{Q} \to \mathrm{SO}(3)$. The frame $\{\sigma\}$ is an inertial coordinate frame that evolves over the backbone ${}^0\boldsymbol{\gamma}$ such that variations in σ give insight into its differential geometry.

and space into Euclidean space $\mathbb{R}^3$:

$$ {}^0\boldsymbol{\gamma} : \mathbb{X} \times \mathcal{Q} \to \mathbb{R}^3; \tag{4.5} $$

and similarly the rotation matrix ${}^0\boldsymbol{\Phi}(\sigma, \boldsymbol{q})$ by a mapping from the generalized coordinates and space into SO(3):

$$ {}^0\boldsymbol{\Phi} : \mathbb{X} \times \mathcal{Q} \to \mathrm{SO}(3), \tag{4.6} $$

where SO(3) denotes the special orthogonal group for rotations about the origin of $\mathbb{R}^3$. Under this notion, the position vectors ${}^0\boldsymbol{\gamma}(0, \boldsymbol{q})$ and ${}^0\boldsymbol{\gamma}(L, \boldsymbol{q})$ relate to the base and the end-effector of the soft manipulator, respectively. Note tat left-sided superscript are used to indicate the frame of reference. The set of all points on the backbone $\boldsymbol{\mathcal{C}} = \left\{ {}^0\boldsymbol{\gamma}(\sigma, \boldsymbol{q}) \in \mathbb{R}^3 \mid \sigma \in \mathbb{X} \right\}$ draws a possible spatial configuration of the soft robot given a time instance $t \in \mathbb{T}$ on a finite horizon $\mathbb{T} = [0, T]$. For sake of brevity, the remainder of the chapter will drop the superscripts that indicate the frame of reference, *i.e.*, ${}^0\boldsymbol{\Phi} = \boldsymbol{\Phi}$ and ${}^0\boldsymbol{\gamma} = \boldsymbol{\gamma}$.

Assumption 4.4 (Piece-wise Constant Curvature) Despite the inherent flexibility in soft robotics, it is sometimes sufficient to express the kinematics according to the *Piecewise Constant Curvature* (PCC) condition. This properties often originates

from the "*proper*" structural design of the soft robot, where parasitic motion is reduced by structural compliance . Mathematically, it implies that the curvature of the continuous body satisfies $\kappa(\sigma_1, \boldsymbol{q}) = \kappa(\sigma_2, \boldsymbol{q})$ for spatial coordinates on a local region on the soft manipulator $\sigma_1, \sigma_2 \subseteq \mathbb{X}$. As a result, this condition allows us to describe the full forward kinematics with a significantly reduced set of generalized coordinates, mitigating kinematic complexity in the model. Numerous works employ PCC models [59, 65, 79, 110, 135, 208], and depending on the elasticity and structural geometry of the soft robot, the PCC condition has been proven to be consistent for various soft robotic systems.

Following this Constant Curvature (CC) approach, we assign a coordinate frame that twists minimally along the backbone – formally called the "*Bishop frame*" (see [25]) – parametrized by the following generalized coordinate vector:

$$\boldsymbol{q} := \begin{pmatrix} \varepsilon & \kappa_x & \kappa_y \end{pmatrix}^\top \in \mathcal{Q}, \tag{4.7}$$

where $\varepsilon_- \leq \varepsilon \leq \varepsilon_+$ is the elongation strain, and κ_x, $\kappa_y \in \mathbb{R}$ are the curvatures or angular strains in x-z and y-z plane, respectively; and $\mathcal{Q} \subset \mathbb{R}^3$ is an admissible space on which $\boldsymbol{q}$ evolves. We will also introduce the following geometric variables $\kappa = \langle \kappa_x, \kappa_y \rangle$ and the curvature angle $\phi = \text{atan2}(\kappa_y, \kappa_x)$. It is worth mentioning that the joint description above is somewhat related to Renda. et al. [170] who proposed a *Piece-wise Constant Strain* (PCS) parametrization with the exception of including the twist along the tangent.

By exploring the differential geometry of the smooth backbone curve similar to Mochiyama et al. [148], we can write the position vector $\boldsymbol{\gamma}(\sigma, \boldsymbol{q})$ and the orientation matrix $\boldsymbol{\Phi}(\sigma, \boldsymbol{q})$ for each material point σ along the smooth backbone as a differential equation:

$$\frac{\partial \boldsymbol{\Phi}}{\partial \sigma}(\sigma, \boldsymbol{q}) = \boldsymbol{\Phi}(\sigma, \boldsymbol{q})\, \boldsymbol{\Gamma}^\times(\sigma, \boldsymbol{q}), \tag{4.8}$$

$$\frac{\partial \boldsymbol{\gamma}}{\partial \sigma}(\sigma, \boldsymbol{q}) = \boldsymbol{\Phi}(\sigma, \boldsymbol{q})\, \boldsymbol{U}(\sigma, \boldsymbol{q}), \tag{4.9}$$

where $\boldsymbol{\Gamma}^\times \in \text{so}(3)$ is a skew-symmetric matrix composed of the entries of the vector $\boldsymbol{\Gamma} \in \mathbb{R}^3$, and $\boldsymbol{U} \in \mathbb{R}^3$ a vector representing the tangent along the extensible backbone. The skew-symmetric operator $(\cdot)^\times$ denotes the isomorphism between the Lie algebra so(3) and $\mathbb{R}^3$. The vectors $\boldsymbol{\Gamma}$ and $\boldsymbol{U}$ are vectors that define the differential geometry of the backbone [148] which are unique entries that live in the tangent space of the rigid-body transformation group (*i.e.*, $T_{\text{SE(3)}}$). Given the Bishop parametrization shown in (4.7) and assuming the Piecewise Constant-

Strain (PCC) condition, these geometric entities yield

$$\boldsymbol{\Gamma}^{\times}(\sigma,t) \simeq \boldsymbol{\Gamma}^{\times}(\sigma,\boldsymbol{q}(t)) \xrightarrow{\text{PCC}} \boldsymbol{\Gamma}^{\times} = \begin{pmatrix} 0 & 0 & \kappa_y \\ 0 & 0 & \kappa_x \\ -\kappa_y & -\kappa_x & 0 \end{pmatrix}, \tag{4.10}$$

$$\boldsymbol{U}(\sigma,t) \simeq \boldsymbol{U}(\sigma,\boldsymbol{q}(t)) \xrightarrow{\text{PCC}} \boldsymbol{U} = \begin{pmatrix} 0 \\ 0 \\ \varepsilon + 1 \end{pmatrix}. \tag{4.11}$$

Now, given an initial configuration of backbone's base, *i.e.*, $\boldsymbol{\Phi}(0,\boldsymbol{q}) = \boldsymbol{\Phi}_0$ and $\boldsymbol{\gamma}(0,\boldsymbol{q}) = \mathbf{0}_3$, we can now solve for the position and orientation for each material coordinate σ along the backbone:

$$\boldsymbol{\Phi}(\sigma,\boldsymbol{q}) = \boldsymbol{\Phi}_0 \exp_{\mathrm{SO}(3)}(\sigma\boldsymbol{\Gamma}^{\times}(\boldsymbol{q})), \tag{4.12}$$

$$\boldsymbol{\gamma}(\sigma,\boldsymbol{q}) = \int_0^{\sigma} {}^0\boldsymbol{\Phi}(s,\boldsymbol{q})\,\boldsymbol{U}(\boldsymbol{q})\,ds, \tag{4.13}$$

where $\exp_{\mathrm{SO}(3)} : \mathrm{so}(3) \to \mathrm{SO}(3)$ is the exponential map. Luckily, there exists a compact expression for the exponential mapping related to the orthogonal group of rotation matrices SO(3) called the "*Rodriguez formulas*". The rotation angle along the soft body can be computed by $\theta(\sigma,\boldsymbol{q}) := \int_0^{\sigma} \kappa(s,\boldsymbol{q})\,ds = \kappa(\boldsymbol{q})\sigma$. Notice that the rotation angle θ linearly depends on σ, which is a property that follows from Assumption 4.4. Then, given the expression for the angle of rotation, we can compactly rewrite the rotation matrix (4.12) in terms of $\cos(\theta)$ and $\sin(\theta)$ using these formulas as follows [131]:

$$\boldsymbol{\Phi}(\theta) = \boldsymbol{\Phi}_0 \left(\boldsymbol{I}_3 + \left[\frac{\sin(\theta)}{\theta}\right] \boldsymbol{\Gamma}^{\times} + \left[\frac{1-\cos(\theta)}{\theta^2}\right] \boldsymbol{\Gamma}^{\times}\boldsymbol{\Gamma}^{\times} \right). \tag{4.14}$$

Note that the closed-form solutions (4.12) and (4.13) represent the forward configuration kinematics of the backbone curve. To express the forward velocity kinematic, let $\boldsymbol{\eta}(\sigma,\boldsymbol{q},\dot{\boldsymbol{q}}) = \left(\boldsymbol{\omega}^{\top},\boldsymbol{v}^{\top}\right)^{\top} \in \mathbb{R}^6 \cong \mathrm{se}(3)$ be the aggregate of the angular velocity and linear velocity components relative to an inertial frame at σ, where the space se(3) denotes the Lie algebra of SE(3). The velocity twist is computed by the following integration procedure:

$$\begin{aligned} \boldsymbol{\eta}(\sigma,\boldsymbol{q},\dot{\boldsymbol{q}}) &= \mathbf{Ad}^{-1}_{\boldsymbol{g}(\sigma,\cdot)} \int_0^{\sigma} \mathbf{Ad}_{\boldsymbol{g}(s,\cdot)}\,\boldsymbol{J}^{\star}\dot{\boldsymbol{q}}\,ds, \\ &=: \boldsymbol{J}(\boldsymbol{q},\sigma)\dot{\boldsymbol{q}}, \end{aligned} \tag{4.15}$$

where $\mathbf{Ad}_g : \mathrm{SE}(3) \to \mathbb{R}^{6\times 6}$ denotes the adjoint transformation matrix regarding the rigid body transformation $\boldsymbol{g} \in \mathrm{SE}(3)$ that maps local velocities (i.e., twist) to a

frame located at σ, and $\boldsymbol{J}^\star : \mathcal{Q} \to T_{\boldsymbol{q}}\mathcal{Q}$ the joint-axis matrix that relates the DOFs to the generalized coordinate description. Let it be clear that the joint-axis matrix is naturally constant for a soft segment modeled with the Constant-Strain (CS) assumption. We will later relax this assumption in Chapter 4. Nevertheless here, the joint-axis matrix for an extensible and bendable CS segment parametrized by the Bishop parameters is given by

$$\boldsymbol{J}^\star := \begin{pmatrix} \dfrac{\partial \boldsymbol{\Gamma}}{\partial \boldsymbol{q}}^\top & \dfrac{\partial \boldsymbol{U}}{\partial \boldsymbol{q}}^\top \end{pmatrix}^\top = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}^\top . \tag{4.16}$$

Although we based the forward kinematics on Mochiyama et al. (2003, [148]), the derived expression for the velocity twist in (4.15) is analogous to the work of Renda et al. (2018, 2020; [169, 170]), and Boyer et al. (2010, 2021; [27, 28]). Please also note that (4.15) gives rise to the geometric manipulator Jacobian $\boldsymbol{J}(\sigma, \boldsymbol{q})$ that defines the mapping from joint velocities to the velocity twist for a point σ on the elastic body.

Given the explicit expression for the velocity twist in (4.15), we can derive the acceleration twist [27, 148, 170] which is obtained through time differentiation of (4.15):

$$\begin{aligned} \dot{\boldsymbol{\eta}}(\sigma, \boldsymbol{q}, \dot{\boldsymbol{q}}, \ddot{\boldsymbol{q}}) &= \boldsymbol{J}\ddot{\boldsymbol{q}} + \mathbf{Ad}^{-1}_{\boldsymbol{g}(\cdot,\sigma)} \int_0^\sigma \mathbf{Ad}_{\boldsymbol{g}(s,\cdot)} \mathbf{ad}_{\boldsymbol{\eta}(s,\cdot,\cdot)} \boldsymbol{J}^\star \dot{\boldsymbol{q}}\, ds \\ &:= \boldsymbol{J}(\sigma, \boldsymbol{q})\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}(\sigma, \boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}}, \end{aligned} \tag{4.17}$$

where $\mathbf{ad}_{\boldsymbol{\eta}} : \mathbb{R}^6 \to \mathbb{R}^{6\times 6}$ denotes the adjoint transformation regarding the velocity twist $\boldsymbol{\eta}^\wedge \in \mathrm{se}(3)$. The reader is referred to Appendix A.1 for more detailed expressions on the adjoint transformations.

Remark 4.2 (Numerical instability near zero-curvature) For many of the PCC modeling frameworks [65], there are mentions of a singularity point or discontinuity of the kinematic formulations at zero-curvature $\kappa = 0$. It is often reported that trajectories that pass through the origin lead to unbounded linear velocities $\boldsymbol{v} := \lfloor \boldsymbol{\eta} \rfloor_3$, which may result in critical problems in practice. Although it is believed the problem is simply a by-product of the PCC hypothesis, this is however a common misconception, and it stems from a numerical origin. To illustrate this, consider the inextensible planar case: $\varepsilon = \kappa_y = 0$ and $\kappa = \kappa_x$. For simplicity, we assume $L = 1$. Hence, by solving the forward kinematics for the position vector $\boldsymbol{\gamma}(\sigma, \kappa)$, and approaching zero-curvature from the positive domain $\kappa^+ \to 0$, we see that

$$\lim_{\kappa \to 0^+} \boldsymbol{\gamma}(\sigma, \kappa) = \begin{pmatrix} \dfrac{1 - \cos(\sigma\kappa)}{\kappa}, & 0, & \dfrac{\sin(\sigma\kappa)}{\kappa} \end{pmatrix}^\top = \begin{pmatrix} 0 & 0 & \sigma \end{pmatrix}^\top, \tag{4.18}$$

so its limit clearly exists. Since the position vector $\boldsymbol{\gamma}$ is continuously differentiable when approaching the origin from both sides $\kappa \to 0^+$ and $\kappa \to 0^-$, it follows that $\dot{\boldsymbol{\gamma}}$ must be bounded for all $\kappa \in \mathcal{Q}$. We can simply check this by investigating the behavior of the linear-velocity components of the geometric Jacobian near zero-curvature, which yield

$$\lim_{\kappa \to 0^+} \lfloor \boldsymbol{J} \rfloor_3(\sigma, \kappa) = \left(\frac{\sigma\kappa \sin(\sigma\kappa) + 1 - \cos(\sigma\kappa)}{\kappa^2}, \quad 0, \quad \frac{\sigma\kappa \cos(\sigma\kappa) - \sin(\sigma\kappa)}{\kappa^2} \right)^\top$$
$$= \begin{pmatrix} \sigma^2 & 0 & 0 \end{pmatrix}^\top . \tag{4.19}$$

Again, its limit exists. Since both limits exist, we define $\boldsymbol{\gamma}(\sigma, 0) := \lim_{\kappa \to 0} \boldsymbol{\gamma}(\sigma, \kappa)$ and $\boldsymbol{J}(\sigma, 0) := \lim_{\kappa \to 0} \boldsymbol{J}(\sigma, \kappa)$. Consequently, the magnitude of the linear velocity of the end-effector reads simply $\|\dot{\boldsymbol{\gamma}}(L, \dot{\kappa})\| = L^2 \dot{\kappa} = L\omega_1$ with ω_1 the angular velocity at the tip. Moreover, it is bounded for all $\kappa \in \mathcal{Q}$. This naturally poses an ambiguity on the origin of the kinematic singularity so often reported literature. The problem is believed to be of numerical origin when considering the zero-division. To make matters worse, deriving analytical expressions for accelerations will contain similar expressions that are hard to stabilize numerically. To resolve this issue, we opt for a numerical approximation of the forward kinematics – namely, we employ an explicit forward integration scheme (*i.e.*, trapezoidal integration) to solve (4.8) and (4.9).

Example 4.1 (Kinematic behavior of PCC segment)**.** As an illustrative example, we perform a numerical simulation of the forward kinematics for a single PCC segment. We select a differentiable reference trajectory $\boldsymbol{q}(t) \equiv \boldsymbol{q}_d(t)$, $\dot{\boldsymbol{q}}(t) \equiv \dot{\boldsymbol{q}}_d(t)$ and $\ddot{\boldsymbol{q}}(t) \equiv \ddot{\boldsymbol{q}}_d(t)$ that passes the zero-curvature point given by :

$$\boldsymbol{q}_d(t) = \operatorname{erf}(t) \cdot \begin{pmatrix} \varepsilon_0 \sin(\omega t) & \kappa_0 \cos(\omega t) & \kappa_0 \sin(\frac{3}{2}\omega t - \frac{\pi}{4}) \end{pmatrix}^\top ,$$

where $\operatorname{erf}(t) := \frac{2}{\pi} \int_0^\tau \exp(-\tau^2)\, d\tau$ is referred to as the error function. Note that these are smooth functions such that reference velocity $\dot{\boldsymbol{q}}_d$ and reference acceleration $\ddot{\boldsymbol{q}}_d$ exist and are bounded. The reference signals for the geometric strain of the soft robot are shown in Figure 4.3. Please note that the reference $\boldsymbol{q}_d$ has been carefully selected to ensure it passes the line $\kappa_x = \kappa_y = 0$ on the configuration manifold $\mathcal{Q}$, *i.e.*, the numerical instability point for (near) zero-curvature.

Then, by injecting the reference into the kinematic relations given by (4.8), (4.9), (4.15), and (4.17), we obtain a (close) approximation of forward kinematics as shown in Figure 4.5. Furthermore, we provided a 3D-rendering of the soft robot subjected to the reference $\boldsymbol{q}_d$ in Figure 4.4. Now, two key observations can be made. First, although a simple harmonic trajectory is used, the resulting

trajectory of the end-effector as shown in Figure 4.5 is rather complex. This perhaps stresses the importance of inverse kinematic solver that can be used for task-space control. Second, although we pass the point of numerical instability for $\kappa \to 0$, we see that the velocity solutions are smooth and bounded at these instances. This result shows our approach does not suffer from the near-zero curvature instabilities that are notoriously mentioned in [60, 65].

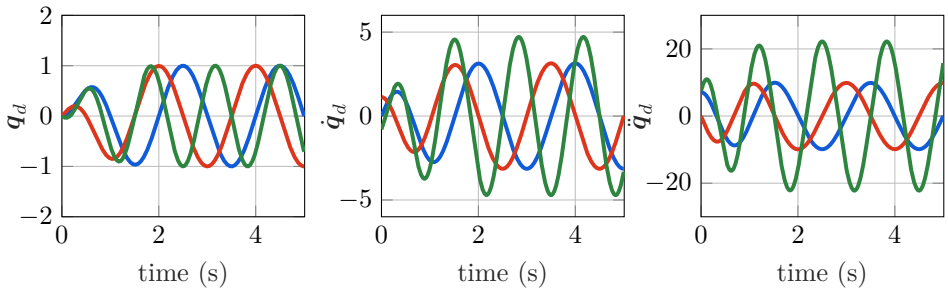


Figure 4.3. The time evolution of the predefined geometric strain parameters of the Piece-wise Constant curvature model $\boldsymbol{q}_d \to (\varepsilon, \kappa_x, \kappa_y)^\top$ and their corresponding time-derivatives $\dot{\boldsymbol{q}}_d$ and $\ddot{\boldsymbol{q}}_d$, given by the (spatially constant) elongation ε (—), and the (spatially constant) curvatures κ_x (—) and κ_y (—).

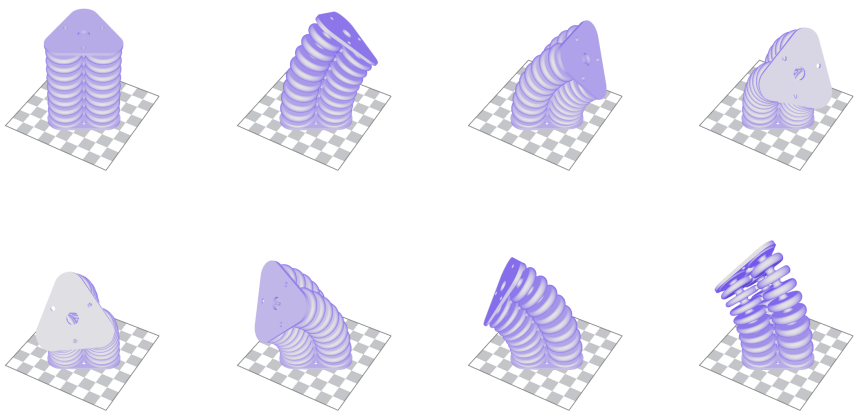


Figure 4.4. Three-dimensional deformation of the three-bellow soft robot manipulator using the PCC model. Based on the prescribed reference $\boldsymbol{q}_d$ (and its time-derivative $\dot{\boldsymbol{q}}_d$), the forward kinematic relations for each point σ along the backbone is computed and the volumetric mesh is deformed accordingly to its closest material-point on $\boldsymbol{\gamma}(\sigma)$.

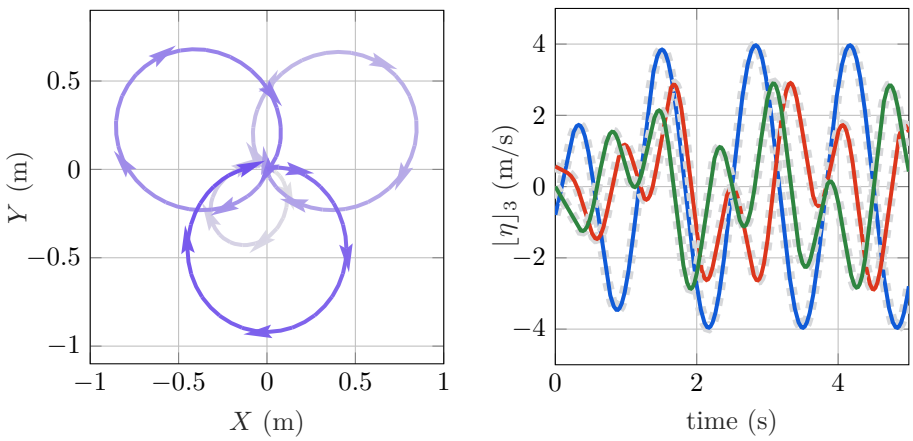


Figure 4.5. (left) The forward kinematics of the end-effector $\boldsymbol{\gamma}$ related to the prescribed reference $\boldsymbol{q}_d(t)$. The time evolution is shown by the colormap $\in [0, 5]$ (s). (right) The linear velocities $\lfloor \boldsymbol{\eta} \rfloor_3 = (v_1, v_2, v_3)^\top$, shown as (—, —, —), respectively. The dashed lines (—) are obtained using numerical time differentiation of $\boldsymbol{\gamma}$ – and the overlap exactly with the forward kinematic model. A key observation here, is that our numerical approach prevents the common numerical instability for near-zero curvatures, *i.e.*, $\kappa \to 0$, since the linear velocities are bounded and continuous even for $\kappa(\boldsymbol{q}) = 0$.

4.3.2 Piecewise curve dynamics using Euler-Lagrange

Given the forward kinematics in (4.12), (4.13), (4.15) and (4.17), we can shift our attention to formulating the finite-dimensional dynamics of the soft robot. Our goal here is to write the spatio-temporal dynamics of the hyper-elastic soft robot as a second-order ODE into the Lagrangian form:

$$\frac{d}{dt}\left(\nabla_{\dot{\boldsymbol{q}}}\,\mathcal{L}\right) - \nabla_{\boldsymbol{q}}\,\mathcal{L} = \boldsymbol{Q}^{\mathrm{nc}}, \tag{4.20}$$

where the mathematical operator $\nabla_{\boldsymbol{x}}\,(\cdot) := \partial(\cdot)^{\top}/\partial \boldsymbol{x}$ denotes the gradient w.r.t. to $\boldsymbol{x}$, $\mathcal{L}(\boldsymbol{q}, \dot{\boldsymbol{q}}) := \mathcal{K}(\boldsymbol{q}, \dot{\boldsymbol{q}}) - \mathcal{U}(\boldsymbol{q})$ the Lagrangian function, $\mathcal{K} \in \mathbb{R}_{\geq 0}$ and $\mathcal{U} \in \mathbb{R}$ the kinetic and potential energy, respectively; and $\boldsymbol{Q}^{\mathrm{nc}} \in \mathbb{R}^n$ a vector of generalized non-conservative forces. To apply the Lagrangian formalism to a continuum dynamical system, consider an infinitesimal slice of the continuum body for each material coordinate σ along the backbone curve. Given this notion, we assign the infinitesimal slice with an inertia tensor $\boldsymbol{\mathcal{M}} = \mathrm{blkdiag}(\rho_0 \boldsymbol{I}, \boldsymbol{\mathcal{J}}_0)$ with $\rho_0 = m_0/L$ the line-density and $\boldsymbol{\mathcal{J}}_0 \in \mathrm{so}^*(3) \times \mathrm{so}(3)$ a symmetric tensor related to the second moment of inertia of infinitesimal slice at σ. Note that operator $\mathrm{blkdiag}(\cdot)$ creates a block diagonal matrix by aligning the input matrices.

The kinetic energy can be obtained through spatial integration of its respective kinetic energy densities [28, 148, 208], *i.e.*, $\mathfrak{T} = \frac{1}{2}\boldsymbol{\eta}^{\top}\boldsymbol{\mathcal{M}}\boldsymbol{\eta}$:

$$\begin{aligned}\mathcal{T}(\boldsymbol{q}, \dot{\boldsymbol{q}}) &= \frac{1}{2}\int_{\mathbb{X}} \boldsymbol{\eta}(\sigma, \boldsymbol{q}, \dot{\boldsymbol{q}})^{\top}\,\boldsymbol{\mathcal{M}}\,\boldsymbol{\eta}(\sigma, \boldsymbol{q}, \dot{\boldsymbol{q}})\; d\sigma, \\ &= \frac{1}{2}\dot{\boldsymbol{q}}^{\top}\left[\int_{\mathbb{X}} \boldsymbol{J}(\sigma, \boldsymbol{q})^{\top}\,\boldsymbol{\mathcal{M}}\,\boldsymbol{J}(\sigma, \boldsymbol{q})\; d\sigma\right]\dot{\boldsymbol{q}} := \frac{1}{2}\dot{\boldsymbol{q}}^{\top}\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{q}},\end{aligned} \tag{4.21}$$

where we used $\boldsymbol{\eta} = \boldsymbol{J}\boldsymbol{q}$ as described in (4.15). Also note that expression for the kinetic energy naturally gives rise to the generalized inertia matrix $\boldsymbol{M}(\boldsymbol{q})$ of the Lagrangian model. By substitution of the kinetic energy into the Euler-Lagrange equation (4.20), we find $\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\boldsymbol{q}$ where $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ denotes the Coriolis matrix. Instead of computing the Coriolis matrix through the conventional Christoffel symbols [154], we a modified computational scheme introduced by Garofalo et al. [76] that is tailored towards long serial-chain manipulators. In their scheme, we replaced the finite summation of N rigid bodies by a spatial integration over the continuum domain $\mathbb{X}$. This leads to the following computation of the Coriolis matrix:

$$\begin{aligned}\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \int_{\mathbb{X}} \boldsymbol{J}(\sigma, \boldsymbol{q})^{\top} \overbrace{\left[\boldsymbol{\mathcal{M}}\mathbf{ad}_{\boldsymbol{\eta}} - \mathbf{ad}_{\boldsymbol{\eta}}^{\top}\boldsymbol{\mathcal{M}}\right]}^{\boldsymbol{\mathcal{C}}_{\boldsymbol{\eta}}} \boldsymbol{J}(\sigma, \boldsymbol{q}) \;+\; \ldots \\ \ldots \;+ \boldsymbol{J}(\sigma, \boldsymbol{q})^{\top}\boldsymbol{\mathcal{M}}\dot{\boldsymbol{J}}(\sigma, \boldsymbol{q}, \dot{\boldsymbol{q}})\; d\sigma,\end{aligned} \tag{4.22}$$

where $\boldsymbol{\mathcal{C}}_{\boldsymbol{\eta}} = -\boldsymbol{\mathcal{C}}_{\boldsymbol{\eta}}^{\top}$ a skew-symmetric matrix. The computation above is slight different from existing literature [27, 169] to ensure that the matrix $\dot{\boldsymbol{M}} - 2\boldsymbol{C}$ is skew-symmetric – the so-called "*passivity condition*"(see Assumption 4.4). The importance of this property will become apparent later in the energy-based controller design. Lastly, the potential energy is given by sum of gravitational potential energy and internal elastic potential, *i.e.*, $\mathcal{U}(\boldsymbol{q}) = \mathcal{U}_{\mathrm{g}}(\boldsymbol{q}) + \mathcal{U}_{\mathrm{e}}(\boldsymbol{q})$. Since gravitational potential energy density is given by $\mathfrak{U}_g = -\rho_0\, \boldsymbol{\gamma}(\sigma, \boldsymbol{q})\boldsymbol{a}_g$ with $\boldsymbol{a}_g \in \mathbb{R}^3$ is a vector of body accelerations, the potential energy related to gravity is obtained by spatial integration of their respective energy densities:

$$\mathcal{U}_{\mathrm{g}}(\boldsymbol{q}) = \int_{\mathbb{X}} \mathfrak{U}_{\mathrm{g}}(\sigma, \boldsymbol{q})\, d\sigma = -\rho_0 \int_{\mathbb{X}} \boldsymbol{\gamma}(\sigma, \boldsymbol{q})^{\top} \boldsymbol{a}_g\, d\sigma. \tag{4.23}$$

To model the hyper-elastic nature, lets introduce two nonlinear stiffness functions for both stretching and bending, denoted by $k_e : \mathbb{R} \mapsto \mathbb{R}_{>0}$ and $k_b : \mathbb{R} \mapsto \mathbb{R}_{>0}$, respectively. These functions allow us to describe a collective elastic behavior imposed by the hyper-elastic materials and the continuum-bodied deformation. It shall be clear that these entities are unique to the soft robot's geometry and soft material choice, and thus finding a suitable candidate model requires further analysis. Later, we will sculpt these nonlinear stiffness functions through Finite Element Methods (FEM). For now, we assume that these analytical nonlinear stiffness functions are known, and thus the (hyper)-elastic potential energy takes the form

$$\mathcal{U}_e(\boldsymbol{q}) = \int_0^{\varepsilon} k_e(s)\, s\, ds + \int_0^{\beta(\boldsymbol{q})} k_b(s)\, s\, ds, \tag{4.24}$$

where ε is the elongation strain, and $\beta(\boldsymbol{q}) := \kappa L(\varepsilon + 1)$ is the bending angle with the total curvature of the segment $\kappa(\boldsymbol{q}) = ||\kappa_x,\, \kappa_y||_2$ (see Figure 4.2).

4.3.3 Overall dynamic model

Finally, by combining (4.20), (4.21), (4.22), (4.23), and (4.24), the continuum dynamics of the soft robot can be casted into the familiar closed form [27, 60, 170] similar to aforementioned model (1):

$$\boldsymbol{M}(\boldsymbol{q})\, \ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\, \dot{\boldsymbol{q}} + \boldsymbol{f}_{\mathrm{e}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{f}_{\mathrm{g}}(\boldsymbol{q}) = \boldsymbol{\tau}(\boldsymbol{u}, \boldsymbol{\delta}), \tag{4.25}$$

where $\boldsymbol{f}_{\mathrm{e}} = \nabla_{\boldsymbol{q}} \mathcal{U}_{\mathrm{e}} + \boldsymbol{R}\dot{\boldsymbol{q}}$ is a vector of generalized forces imposed by the deformation of the soft materials with $\boldsymbol{R} \succ 0$ the Rayleigh damping matrix, $\boldsymbol{f}_{\mathrm{g}} = \nabla_{\boldsymbol{q}} \mathcal{U}_{\mathrm{g}}$ a vector of generalized gravitational forces, and $\boldsymbol{u}(t)$ the control input with the index m the number of pressure inputs. The generalized input vector is chosen of the form:

$\boldsymbol{\tau}(\boldsymbol{u}, \boldsymbol{\delta}) = \boldsymbol{G}\boldsymbol{u} + \boldsymbol{\delta}$ with $\boldsymbol{G}\boldsymbol{u} : \mathbb{R}^m \to \mathbb{R}^n$ a mapping from the input space to the joint actuation space, and $\boldsymbol{\delta}(t)$ an external disturbance (*e.g.*, unmodelled material uncertainties).

Remark 4.3 Given the context of pick-and-place applications in robot manipulators , a possible disturbance $\boldsymbol{\delta}(t)$ could be an external mass applied to the tip of the soft robot. Given the kinematic relations in (4.15) and (4.17), one can describe the disturbance (modeled here as a point-mass located at L) as an external wrench acting on the point $\sigma = L$. The disturbance has two part related to acceleration: *(i)* the gravitational acceleration expressed in the body-frame $\mathbf{Ad}^{-1}_{\boldsymbol{g}(\cdot,L)}\boldsymbol{a}_g$ and *(ii)* the acceleration due to the tip motion $\dot{\boldsymbol{\eta}}(\cdot, L)$. Together they form a state-dependent vector as follows:

$$\boldsymbol{\delta}_m = m_\delta \lfloor \boldsymbol{J}(\cdot, L) \rfloor_3^\top \left(\mathbf{Ad}^{-1}_{\boldsymbol{g}(\cdot,L)}\boldsymbol{a}_g + \lfloor \dot{\boldsymbol{\eta}}(\cdot, L) \rfloor_3 \right), \tag{4.26}$$

where $m_\delta > 0$ the applied mass to the end-effector, $\lfloor \cdot \rfloor_3$ extracts the last three rows of a matrix or vector. Recall that the acceleration twist can be computed through the geometric Jacobian and its time derivative, i.e., $\dot{\boldsymbol{\eta}} = \boldsymbol{J}\ddot{\boldsymbol{q}} + \dot{\boldsymbol{J}}\dot{\boldsymbol{q}}$. Indeed, the PCC condition for a soft body can only accurately describe the true dynamics if external forces produced by mass m_δ do not excessively exceed the intrinsic elastic balancing forces $\boldsymbol{f}_\mathrm{e}(\boldsymbol{q})$. Alternatively, a soft body can be modeled using multiple PCC curves of smaller size, similar to standard Finite Element discretization.

Assumption 4.5 (Input mapping of bellows) Uptil now, we have not specific the input map $\boldsymbol{G}$. In general, the input map is difficult to properly estimate as the system's inputs are distributed over the soft continuum body. Therefore, it involves integrating the actuation wrench along the backbonce curve, while accounting for the spatial dependency of the geometric manipulator Jacobian $\boldsymbol{J}(\boldsymbol{q}, \sigma)$. However, we considering the soft robotic system in Figure 4.1, we can introduce some approximations for the actuation mapping $\boldsymbol{G}$ based on the geometry, placement, and orientation of the (pneumatic) soft actuators. Since the pneumatic chambers are aligned parallel to the backbone curve and are equally spaced along the circumference, we propose the following ansatz:

$$\boldsymbol{G} \simeq \begin{pmatrix} \alpha_\varepsilon & \dots & \alpha_\varepsilon \\ -\alpha_\kappa \cos(\phi_1) & \dots & -\alpha_\kappa \cos(\phi_m) \\ \alpha_\kappa \sin(\phi_1) & \dots & \alpha_\kappa \sin(\phi_m) \end{pmatrix}, \tag{4.27}$$

where $\alpha_\varepsilon, \alpha_\kappa > 0$ are system parameters representing the effective transferal of differential pressure to joint forces, and $\phi_i = (i-1) \cdot \frac{2\pi}{m}$ the angular inter-distance between the m-number of pneumatic bellows. Please note that the parameters α_ε and α_κ are dependent on the bellow area and radius from the bellow to the backbone curve.

4.4 Extension to multi-link systems

We previously expressed the position and velocity kinematics as explicit functions of the generalized coordinates (i.e., Bishop parameters) and their time-derivatives. This explicit dependency stems from the PCC conditions inferring the curvature is non-varying along the spatial domain $[0, L]$, *i.e.*, $\kappa(\sigma, \boldsymbol{q}) = \kappa(\boldsymbol{q})$. Although sufficient for some cases, the condition is generally restrictive, and to some extent inconvenient, since the inclusion of multiple links demands piece-wise integration of the kinematics (4.13), (4.14), (4.15), and (4.17). Rather than separation of integration, we can extend this PCC description by using piece-wise continuous spatial function to distinguishes multiple soft-bodied links along the continuous body of the soft robot. The idea of parametrization through shapes functions has been explored earlier by Chirikjian et al. [42, 46], and later by Boyer et al. [27], Della Santina et al. [60]. A similar discontinuous shape function series was used by Berthet-Rayne et al. [22] to pursue multi-body dynamics for growing continuum robots; and proposed by Chirikjian [42] for hyper-redundant robots earlier.

Following the aforementioned works, let us parameterize the geometric strains $\boldsymbol{\Gamma}$ and $\boldsymbol{U}$ for a multi-link soft robot with N number of links through the product of a basis of orthonormal functions $\{\theta_i\}_{i=1}^{N}$ and the Bishop parametrization. Contrary to (4.10) and (4.11)

$$\boldsymbol{\Gamma}(\sigma, \boldsymbol{q}) \simeq \sum_{i=1}^{N} \theta_i(\sigma) \lceil \boldsymbol{J}^{\star} \rceil_3 \, \boldsymbol{z}_i, \tag{4.28}$$

$$\boldsymbol{U}(\sigma, \boldsymbol{q}) \simeq \sum_{i=1}^{N} \theta_i(\sigma) \lfloor \boldsymbol{J}^{\star} \rfloor_3 \, \boldsymbol{z}_i + \boldsymbol{U}^{\circ}, \tag{4.29}$$

where $\boldsymbol{J}^{\star}$ is the joint-axis matrix as in (4.16), the mathematical operators $\lceil \cdot \rceil_3$ and $\lfloor \cdot \rfloor_3$ extract the first or last three rows of a matrix, respectively; $\tilde{q}_i$ the joint variables of the i-th link, and $\theta_i : [0, L] \mapsto \{0, 1\}$ is a piece-wise constant shape function, whose purpose is to be non-zero for a given interval on $\mathbb{X}$. The new generalized coordinate vector becomes the aggregate of all joint variables of the multi-body soft robotic system $\boldsymbol{q} = \left(\boldsymbol{z}_1^{\top}, \boldsymbol{z}_2^{\top}, ..., \boldsymbol{z}_N^{\top}\right)^{\top}$ with the vector $\boldsymbol{z}_i = (\varepsilon_i, \kappa_{x,i}, \kappa_{y,i})^{\top}$ relating to the Bishop parametrization of the i-th link.

Given (4.28) and (4.29), we may now rewrite the velocity-twist as

$$\boldsymbol{\eta}(\sigma, \boldsymbol{q}, \dot{\boldsymbol{q}}) = \left[\mathbf{Ad}_{\boldsymbol{g}}^{-1} \int_0^{\sigma} \mathbf{Ad}_{\boldsymbol{g}} \boldsymbol{J}^{\star} \boldsymbol{\Theta}(s) \, ds \right] \dot{\boldsymbol{q}} := \boldsymbol{J}(\sigma, \boldsymbol{q}) \dot{\boldsymbol{q}} \tag{4.30}$$

where $\boldsymbol{\Theta}(\sigma) = (\theta_1, \theta_2, ..., \theta_n) \otimes \boldsymbol{I}_n$ is an unitary selection matrix derived from the basis of piece-wise continuous shape functions $\{\theta_i\}_{i=1}^{n}$. Substitution of the

discontinuous variation of the geometric Jacobian in (4.30) into (4.21) leads to the dynamic model of a N-link soft robot manipulator in the Lagrangian form similar to (4.25).

Example 4.2 (Piece-wise selection for two-link system)**.** To reduce ambiguity on the selection matrix $\boldsymbol{\Theta}(\sigma)$, lets consider a spatial coordinate $\sigma_2 \in [L_1, L_1 + L_2)$ that lies on the spatial interval of the second link. Consequently, the operation $\boldsymbol{\Theta}(\sigma_2)\boldsymbol{q} = \boldsymbol{z}_2$ returns the corresponding joint variable of the second link. This selection of generalized coordinates follows analogously for other links along the serial-chain of the soft manipulator.

4.5 Accelerated computation of PDE-like systems using Matrix Differential Equations (MDEs)

Due to the partial differential nature of soft robots, obtaining a closed-form expression for the projected Lagrangian model in (4.25) can become notoriously long and complex (especially for multi-link systems). The origin of this problem stems from the integrands of inertia matrix $\boldsymbol{M}(\boldsymbol{q})$ in (4.21) and Coriolis forces $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ in (4.22); which become highly nonlinear and therefore difficult to calculate a-priori. As a result, solving the forward dynamics using traditional solvers often deteriorates the real-time performance, and in turn its usability for closed-loop control. Inspired by Boyer et al. [27] and Godage et al. [79], instead of finding an exact solution to the dynamic entries $\boldsymbol{M}(\boldsymbol{q})$, $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ and $\boldsymbol{f}_{\mathrm{g}}(\boldsymbol{q})$, let us introduce a similar reduced-order integration scheme that produces an approximate of the dynamic model (4.25). Yet, instead of using an inverse Newton-Euler algorithm (i.e., Featherstone algorithm [201]) in which the Lagrangian entries are built column-wise, we propose an explicit integration scheme that efficiently computes all Lagrangian entities in parallel through a so-called Matrix-Differential Equation (MDE).

The idea here is to replace all necessary spatial integrations required for the Lagrangian entities with an equivalent Matrix-Differential Equation of the form:

$$\frac{\partial \boldsymbol{Z}}{\partial \sigma} = \boldsymbol{F}(\boldsymbol{Z}, \sigma), \tag{4.31}$$

where $\boldsymbol{Z}(\cdot, \sigma)$ is a matrix-valued function composed of the necessary elements for the forward kinematics and forward dynamics, and $\boldsymbol{F}(\boldsymbol{Z}, \sigma)$ a matrix-valued flow function that describes the spatial evolution of $\boldsymbol{Z}$. Then, by choosing the appropriate initial condition for $\boldsymbol{Z}(\cdot, \sigma = 0) =: \boldsymbol{Z}_0$ and numerically solving (4.31) over a finite horizon $\mathbb{X}$, we can retrieve an approximate of the Lagrangian model in (4.25) by extracting the necessary elements from the solution $\boldsymbol{Z}(\cdot, L)$.

Before describing the MDE, let us first introduce two intermediate matrices related to the computation of the manipulator Jacobian and its time-derivative, namely:

$$\frac{\partial \boldsymbol{B}_1}{\partial \sigma} = \mathbf{Ad}_{\boldsymbol{g}(\cdot,\sigma)}\, \boldsymbol{J}^{\star}\boldsymbol{\Theta}(\sigma) \tag{4.32}$$

$$\frac{\partial \boldsymbol{B}_2}{\partial \sigma} = \mathbf{Ad}_{\boldsymbol{g}(\cdot,\sigma)}\mathbf{ad}_{\boldsymbol{\eta}(\cdot,\sigma)}\, \boldsymbol{J}^{\star}\boldsymbol{\Theta}(\sigma) \tag{4.33}$$

such that they satisfy $\boldsymbol{J}\dot{\boldsymbol{q}} = \mathbf{Ad}_{\boldsymbol{g}}{}^{-1}\boldsymbol{B}_1\dot{\boldsymbol{q}}$ and $\dot{\boldsymbol{J}}\dot{\boldsymbol{q}} = \mathbf{Ad}_{\boldsymbol{g}}{}^{-1}\boldsymbol{B}_2\dot{\boldsymbol{q}}$. Given the expressions above, we can now include a partial computation Jacobians into the MDE. By collecting all the differential relation for the forward kinematics (4.8), (4.9), (4.15) and forward dynamics (4.21), (4.22) and (4.23), we can assign a flow function $\boldsymbol{F} := \text{blkdiag}\,(\boldsymbol{F}_{\text{kin}}, \boldsymbol{F}_{\text{dyn}})$ composed of two matrices:

$$\boldsymbol{F}_{\text{kin}} = \left(\begin{array}{cc|c|c} \boldsymbol{\Phi}\boldsymbol{\Gamma}^{\times} & \boldsymbol{\Phi}\boldsymbol{U} & \mathbf{Ad}_{\boldsymbol{g}}\boldsymbol{J}^{\star}\boldsymbol{S} & \mathbf{Ad}_{\boldsymbol{g}}\mathbf{ad}_{\boldsymbol{\eta}}\boldsymbol{J}^{\star}\boldsymbol{S} \\ \mathbf{0}_{3\times 3} & \mathbf{0}_3 & & \end{array}\right), \tag{4.34}$$

$$\boldsymbol{F}_{\text{dyn}} = \left(\frac{\partial \boldsymbol{M}}{\partial \sigma} \quad \frac{\partial \boldsymbol{C}}{\partial \sigma} \quad \frac{\partial \boldsymbol{f}_{\text{g}}}{\partial \sigma}\right), \tag{4.35}$$

in which the differential form of the dynamic entities $\boldsymbol{M}(\boldsymbol{q})$, $\boldsymbol{C}(\boldsymbol{q},\dot{\boldsymbol{q}})$, and $\boldsymbol{f}_{\text{g}}(\boldsymbol{q})$ of the Lagrangian model are given by

$$\frac{\partial \boldsymbol{M}}{\partial \sigma} = (\mathbf{Ad}_{\boldsymbol{g}}^{-1}\boldsymbol{B}_1)^{\top}\boldsymbol{\mathcal{M}}(\mathbf{Ad}_{\boldsymbol{g}}^{-1}\boldsymbol{B}_1), \tag{4.36}$$

$$\frac{\partial \boldsymbol{C}}{\partial \sigma} = (\mathbf{Ad}_{\boldsymbol{g}}^{-1}\boldsymbol{B}_1)^{\top}\left[\boldsymbol{\mathcal{C}}_{\boldsymbol{\eta}}(\mathbf{Ad}_{\boldsymbol{g}}^{-1}\boldsymbol{B}_1) + \boldsymbol{\mathcal{M}}(\mathbf{Ad}_{\boldsymbol{g}}^{-1}\boldsymbol{B}_2)\right], \tag{4.37}$$

$$\frac{\partial \boldsymbol{f}_{\text{g}}}{\partial \sigma} = (\lfloor \boldsymbol{B}_1 \rfloor_3)^{\top}\rho_0\, \boldsymbol{a}_g, \tag{4.38}$$

We wish to stress that $\boldsymbol{F}_1$ collects all elements related to the forward kinematics, whereas $\boldsymbol{F}_2$ contains the dynamic entities related to the Lagrangian model. Following the spatial Matrix-Differential equation in (4.31), its solution will be a matrix $\boldsymbol{Z} := \text{blkdiag}\,(\boldsymbol{Z}_{\text{kin}}, \boldsymbol{Z}_{\text{dyn}})$ composed of two state matrices $\boldsymbol{Z}_{\text{kin}}$ and $\boldsymbol{Z}_{\text{dyn}}$:

$$\boldsymbol{Z}_{\text{kin}}(\sigma,\boldsymbol{q},\dot{\boldsymbol{q}}) := \left(\begin{array}{cc|c|c} \boldsymbol{\Phi} & \boldsymbol{\gamma} & \boldsymbol{B}_1 & \boldsymbol{B}_2 \\ 0_{3\times 3} & 0_3 & & \end{array}\right), \tag{4.39}$$

$$\boldsymbol{Z}_{\text{dyn}}(\sigma,\boldsymbol{q},\dot{\boldsymbol{q}}) := \begin{pmatrix} \boldsymbol{M} & \boldsymbol{C} & \boldsymbol{f}_{\text{g}} \end{pmatrix}, \tag{4.40}$$

Such a set of Matrix-Differentials as in (4.31) are not supported natively by standard ODE solvers. Therefore, an explicit second-order Runge-Kutta solver for

MDEs is developed such that efficiently computes the evolution of the state matrix $\boldsymbol{Z}$ along $\mathbb{X} = [0, L]$. The numerical solver is written in *MATLAB* `2021a` and it can be found in the public repository of *Sorotoki* (see implementation at [33]).

As for state trajectories along the temporal regime $\mathbb{T} = [0, T]$, an implicit trapezoidal integration scheme is proposed to solve the approximated continuum dynamics, which are generally less conservative on discretization to preserve numerical stability. Here implicit schemes are favored over the explicit scheme since a coarser time integration can significantly increase real-time performance. In addition, to further boost the performance of the temporal integration, a cost-effective approximation of the Hessian is introduced. For more detail on the temporal integration scheme of the solver can be found in Appendix A.3

4.6 Parameter identification

In this section, the nonlinear stiffness function for elongation and bending, $k_e : \mathbb{R} \to \mathbb{R}_{>0}$ and $k_b : \mathbb{R} \to \mathbb{R}_{>0}$, respectively, are solidified such that the elastic deformation aligns with the physical system seen in Figure 4.1. Numerous studies consider these stiffnesses to be linear, however, the presence of exotic materials and complicated structures would justify the modeling of nonlinear elastic behavior. Here, we extend these conservative material models and explore the nonlinear and time-varying regime.

4.6.1 Finite element method and hyper-elasticity

Generally, soft robots are operated by (differential) pressure to air channels embedded in the elastic body. If the applied pressure is sufficiently larger than the ambient pressure, the elastic body deforms to counteract the external forces – the critical point at which the external force overcome the internal elastic forces is proportional to the Young's modulus of the material. To enable efficient mobility, soft robots often explore of materials (or material composites) with a low Young's moduli, e.g., silicone elastomers. Unfortunately, large deformations of these rubber-like materials inherently lead to state-dependency in the mechanical compliance, and thus Hookean elasticity is no longer accurate – rendering them hyper-elastic. These hyper-elastic materials branch a whole new subfield in continuum mechanics. Although analytic descriptions exist, hyper-elasticity is generally treated numerically through Finite Element techniques [51, 62, 124] paired with a (nonlinear) continuum mechanics framework.

Many variations of constitutive models for hyper-elastic materials are available, including Saint Venant-Kirchhoff, Neo-Hookean, Mooney-Rivlin, Ogden, and Yeoh

which are detailed in various work [142**?**] . In Mustaza et al. [155], a Yeoh constitutive model is explored to describe hyper-elastic material characteristics of a silicone-composite actuator. Duriez et al. [62] and related works [51, 124] employ Neo-Hookean material models to enrich the nonlinear deformations in FEM-driven models. There are many different constitutive models available, each better suited to describe specific nonlinear elastic behavior. Constitutive material models are mathematical functions used to express the (nonlinear) relationship between stress and strain in terms of deformation. Let the spatial domain of robot's soft continuum body be given by $\mathbb{V}$ which is a compact subset of $\mathbb{R}^3$. This continuum body can be regarded as a collection of continuum particles called "*material points*" represented by a spatial coordinate $\boldsymbol{x} \subseteq \mathbb{V}_0$. Now, assume there exists a mapping $\boldsymbol{\phi} : \mathbb{V}_0 \to \mathbb{V}$ that maps these material points $\boldsymbol{x}$ into their deformed configuration $\mathbb{V}$. Then, given the deformation map $\boldsymbol{\phi}$, the local geometrical deformations of the continuum solid relative to an undeformed configuration are described by the deformation gradient tensor given as follows [**?**]:

$$\boldsymbol{F}(\boldsymbol{x}) := \boldsymbol{I}_3 + \nabla_{\boldsymbol{x}}\, \boldsymbol{\phi}(\boldsymbol{x}), \tag{4.41}$$

For hyper-elastic materials, it is postulated that there exists a potential energy function $\boldsymbol{F} \mapsto \Psi(\boldsymbol{F})$ that is a function of the strain tensors. This potential function $\Psi(\boldsymbol{F})$ is also referred to as strain-energy density function, which depends exclusively on the material deformation.

In this work, we regard the Saint Venant-Kirchhoff constitutive model for hyper-elasticity [95, 117]. The Saint Venant-Kirchhoff model is in many ways similar to linear elastic materials (*i.e.*, Hooke's law), however, it is an extension from linear deformations into the nonlinear regime. The strain-energy density function for the Saint Venant-Kirchhoff model is defined as

$$\Psi^{\mathrm{SV}} := \frac{\lambda}{2} \operatorname{trace}(\boldsymbol{\mathcal{E}})^2 + \mu \operatorname{trace}(\boldsymbol{\mathcal{E}}^2), \tag{4.42}$$

where $\boldsymbol{\mathcal{E}} = \frac{1}{2}(\boldsymbol{F}^\top \boldsymbol{F} - \boldsymbol{I})$ the Green-Lagrange strain tensor, trace$(\cdot)$ denotes the trace of a tensor, and $\lambda > 0$ and $\mu > 0$ are the Lamé parameters which arise from the strain-stress relationships of the elastic material. The relations for the Lamé parameters are given by

$$\lambda = \frac{\nu E_0}{(1+\nu_0)(1-2\nu_0)} \ \text{MPa}, \qquad \mu = \frac{E_0}{2(1+\nu_0)} \ \text{MPa}; \tag{4.43}$$

where E_0 is the Young's modulus or elasticity modulus and ν_0 is a dimensionless constant denoting the Poisson ratio. It is worth mentioning that the Yeoh or Ogden model is more suitable for silicone elastomer materials that are conventional material models for soft robotics.

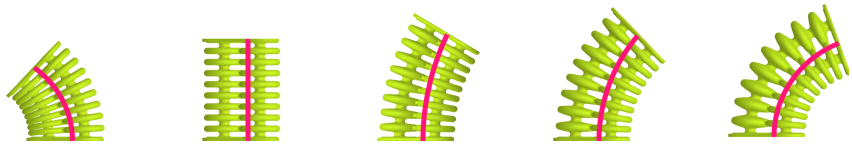


Figure 4.6. High resolution finite element simulations of the soft robot maipulator (—) subjected to various input pressures $-20\,\text{kPa} \leq \boldsymbol{u}_i(t) \leq 80\,\text{kPa}$. These results are produced using `Abaqus/CEA` numerica solver. To validate the PCC condition, an optimal backbone curve $\boldsymbol{\gamma}(\boldsymbol{q}^\star)$ is shown (—) whose joint coordinates are recovered by solving the optimization problem (4.44).

In order to invoke the constitutive material law (4.42) for three-dimensional solids, we explore the finite element method. We generated a finite element mesh of the soft robot manipulator in Figure 4.1. The finite element analysis has been carried out in `Abaqus/CEA` with variable time increments. Given preliminary uniaxial tension tests, the 3D-printed elastomer material is estimated to be linear isotropic described by the following Lamé parameters: $E_0 = 80$ MPa, $\nu_0 = 0.4$ (-). The Lamé parameters can be computed according to (4.43). Furthermore, tangential (frictionless) contact interaction is included in the numerical simulation to prevent self-intersection of the elastic body.

Two finite element simulations are performed to independently characterize the elongation and bending stiffness of the soft robot. Due to simplicity, we start with the elongation stiffness. Each embedded bellow is actuated simultaneously up to a quasi-static differential pressure -20 kPa $\leq u_1(t) = u_2(t) = u_3(t) \leq 30$ kPa. Due to the symmetry of soft actuators, the resulting deformation will be exclusively in axial-direction. The corresponding elongation strain of the soft robot can then be found by recovering the maximum vertical displacement of the nodal mesh, *i.e.*, $\varepsilon = L^{-1} \max(U_z)$. Secondly, the analysis of the bending stiffness is conducted by actuating a single bellow up to a quasi-static differential pressure -30 kPa $\leq u_1(t) \leq 80$ kPa, while $u_2(t) = u_3(t) = 0$ kPa. To recover the bending angle of the elastic body, certain spatial coordinates of nodes close to the end effector are tracked. Given their global coordinates, a constraint nonlinear optimization `fmincon.m` is used to recover optimal Bishop parameters κ and ε subjected to the kinematic relation in (4.14) and (4.13).

$$\begin{aligned} \boldsymbol{q}^\star = \; & \underset{\boldsymbol{q}}{\operatorname{argmin}} \; \left\| \log_{\mathrm{SE}(3)} \left[\boldsymbol{g}^{-1}(L, \boldsymbol{q})\, \boldsymbol{g}_L^{\mathrm{FEM}}(\boldsymbol{u}) \right] \right\|_2 , \\ & \text{s.t.} \quad \boldsymbol{q} \in \mathcal{Q} \end{aligned} \tag{4.44}$$

where $\boldsymbol{g}_L^{\mathrm{FEM}}$ is simply the corresponding end-effector configuration derived from

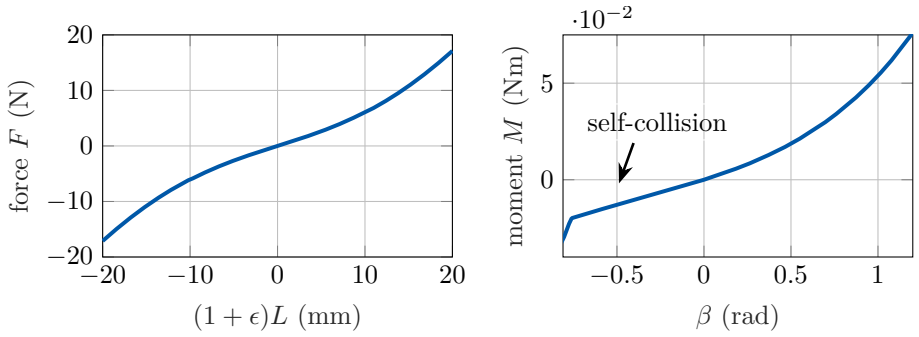


Figure 4.7. Elongation force F (N) and bending moment M (N m) recovered from the finite element data analysis. Note that the curve parametrization using the Bishop variables $\varepsilon = q_1$ and $\kappa = \sqrt{\kappa_x^2 + \kappa_y^2} = \sqrt{{q_1}^2 + {q_2}^2}$ are found using the optimization routine in (4.44). Notice also that self-collision occurs between the bellows for negative bending angles.

4

the high-resolution finite element simulation given the quasi-static input $\boldsymbol{u}$. To some extent, the problem (4.44) can be viewed as an inverse kinematics optimization problem subject to the desired end-effector configuration.

Remark 4.4 It is worth noting that the optimization routine (4.44) for state variable $\boldsymbol{q}^\star$ has a global minimizer. Namely, when both position and orientation are considered in the inverse kinematic optimization, there exist only one solution to $\boldsymbol{q}^\star$ on the configuration manifold $\mathcal{Q}$ (i.e., the length is bounded). This makes the problem well-defined. Uniqueness is desirable as this makes it more easy to relate the reduced beam model to the FEM data. A problem arises, however, when the approach extends to non-constant curvature or multi-link cases. In this cases, the system is hyper-redundant and thus many, perhaps infinite, solutions may exist. To solve this, a regularization term must be added $\mathcal{U}^\star = \frac{1}{2}\boldsymbol{q}^\top \boldsymbol{K}\boldsymbol{q}$ with some positive definite matrix $\boldsymbol{K} \succ 0$. An example for $\boldsymbol{K}$ is the generalized stiffness matrix linearized around $\boldsymbol{q} = \boldsymbol{0}_n$. This ensures the inverse kinematic solutions on $\mathcal{Q}$ will also minimize the elastic potential energy – which is again a global minimum. If gravity dominates elasticity, the optimization routine can be enriched by using the full potential energy and energy applied by the external (pressure) input. However, in this case many solution may exist thus making it difficult to relate the estimates to the FEM data. Furthermore, its solutions also depend on the initialization of th solver.

Following the PCC condition, the bending angle β can be calculated straightforwardly. Given these geometric curve parameters and the effective areas of the bellows, the applied elongation force and bending torque can be computed accordingly. The numerical results are provided in Figure 4.7. In practice, these nonlinear strain relations can also be determined experimentally; however, the numerical methods have the beneficial convenience of several post-processing procedures and gravitation-free deformations. To support our previous claim concerning the consistency of the PCC condition, Figure 4.6 provides a few FEM snapshots results together with the optimal backbone curve from (4.13). It can be seen that the piece-wise constant curvature (PCC) condition, although a clear oversimplification of the true mechanics at hand, is remarkably consistent with the FEM simulations.

4.6.2 Hyperelasicity in joint space

From the finite element results in Figure 4.6, the mathematical description for the nonlinear stiffness can be detailed further. However, a suitable candidate function must be chosen first to properly represent the hyperelastic stress-strain relation. The stiffness function $k_e(\varepsilon)$ and $k_b(\beta)$ have to satisfy the following properties.

Assumption 4.6 (Elastic boundedness) There exists positive constants k^- and k^+ such that $k^- \leq k_e(\xi), k_b(\xi) \leq k^+$ for all possible strains $\xi \in \mathbb{R}$. Given these bounds, it follows that conservative force produced by any deformation, given by $\mathcal{F} = \int_0^\xi k(s)s\, ds$, must be a monotonically increasing unction.

Assumption 4.7 (Deformations reversibility) The stiffness functions k_e and k_b have a global optimum (i.e., either a maximum or minimum) at the origin.

Assumption 4.6 and 4.7 are necessary since they inhibit any buckling-type behavior, *i.e.*, elastic bodies being able to store energy when a large enough forces is applied. Then. consider the following elasticity models for the nonlinear (hyperelastic) elongation and bending stiffness:

$$k_e(\varepsilon) = \alpha_1 + \alpha_2 \left(\tanh[\alpha_3\, \varepsilon]^2 - 1\right), \tag{4.45}$$

$$k_b(\boldsymbol{q}) = \alpha_\phi(\boldsymbol{q}) \cdot \left[\alpha_4 + \alpha_5 \left(\tanh[\alpha_6\, \beta(\boldsymbol{q})]^2 - 1\right)\right], \tag{4.46}$$

where $\boldsymbol{\alpha} = (\alpha_1,\, \alpha_2,\, \alpha_3, \alpha_4,\, \alpha_5,\, \alpha_6)^\top$ is a vector composed of the (possibly time-varying) stiffness parameters, and $\alpha_\phi : \mathcal{Q} \to [1, \infty)$ a nonlinear correction term for asymmetry along the circumference of the radial-axis. Please note that these nonlinear functions possess a decomposable structure: a linear term and a nonlinear term that mimics strain-hardening or strain-softening. As for the asymmetric stiffness due to the layout of the pneumatic bellows, we propose the following ansatz:

Assumption 4.8 (Stiffness variation under radial offset) Given the radial layout of the pneumatic bellows of the soft robot (see Figure 4.1), we assume that the nonlinear correction term for asymmetric radial stiffness along the circumference can be modeled by:

$$\alpha_\phi(\boldsymbol{q}) = \frac{1}{2}\beta\big[\sin(m\,\phi(\boldsymbol{q})) + 1\big] + 1, \tag{4.47}$$

where m is the number of bellows, and $\phi(\boldsymbol{q}) = \text{atan2}(\kappa_y, \kappa_x)$ the direction angle or heading. This stiffness correction term ensures the nonlinear bending stiffness becomes larger between bellows – as it causes simultaneous deformation of multiple bellows. Please note that $\alpha_\phi(\boldsymbol{q}) \geq 1$ for all $\boldsymbol{q} \in \mathcal{Q}$.

To satisfy the aforementioned conditions, it should hold that $\alpha_1 > \alpha_2$, $\alpha_4 > \alpha_5$ and $\alpha_{1,4} > 0$. To find the hyper-elastic parameter vector $\boldsymbol{\alpha}$, we use a weighted least-squares optimization. Recalling the previous profiles for elongation force and bending moment obtained from the FEM data, see Figure ??, we integrate the expressions (4.45) and (4.46) to find an analytic approximation, i.e., $F(\varepsilon;\boldsymbol{\alpha}) := \int_0^\varepsilon k_e(\xi)\xi\, d\xi$ and $M(\beta;\boldsymbol{\alpha}) := \int_0^\beta k_b(\xi)\xi\, d\xi$, respectively. The weighted regression is biased towards positive strains, to better represent the deformation characteristics under positive pressurization. Again, we use `fmincon.m` optimizer in *Matlab*. The hyper-elastic material parameters $\boldsymbol{\alpha}$ that minimize the residual between the FEM data and the analytic model are given in Table 4.1, and the comparison between the FEM force profiles and our solution is given in Figure 4.8 Notice that self-collision is not capture by the stiffness model. Theoretically, the self-contact interactions (as seen in Figure 4.8) can also be parameterized using a different set of polynomials, which are non-zero for the compression regions and zero otherwise. In other words, the regression can be enriched with polynomials that are asymmetric with respect to the origin. We also provided Figure 4.9 to show the change in stiffness when the soft robot deforms, where we can clearly see a discrete radial symmetry arise due to the radial composition of the three pneumatic bellow network of the sot robot.

4.6.3 Visco-elastic creep

In material mechanics, the tendency of mechanical solids to move slowly under stress is called creep. Unlike ideal elastic materials, when polymeric materials are subjected to abrupt change in stress, the constitutive network of polymer chains reconfigure until the stress is evenly distributed. Therefore, we introduce a new state vector $\boldsymbol{\lambda}(t) \in \mathbb{R}^n$, which contains the creep state variables. According to Meyer et al. (2009, [142]), the Kelvin-Voigt model for creep is given by a first-

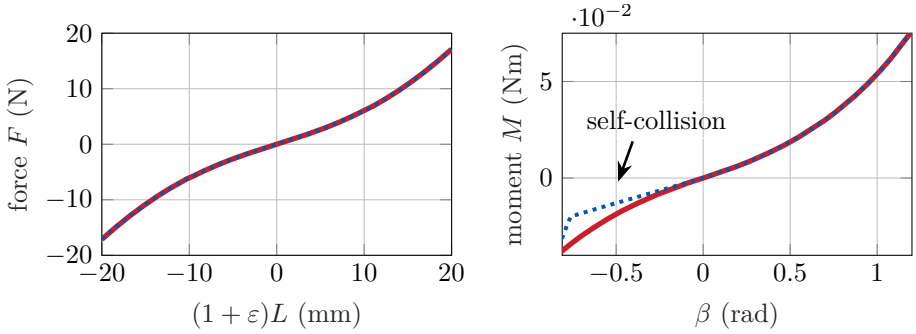


Figure 4.8. Comparison of the (nonlinear) mechanical compliance between the proposed hyper-elastic model (—) and the finite element simulations (—). Notice the proposed stiffness model (4.46) does not capture the self-collision.

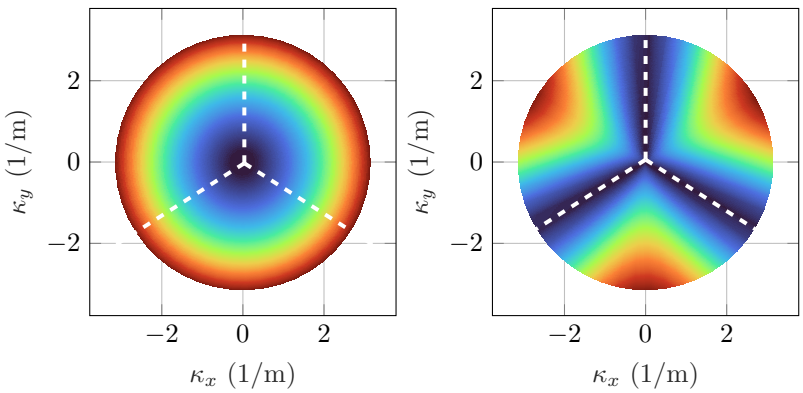


Figure 4.9. Amplitude of the nonlinear stiffness components for elongation and bending. Both stiffness functions are evaluated for a radially distributed sampling of the curvature joint space $(\kappa_x, \kappa_y) \in [-\pi, \pi)$ (left) Elongation stiffness, the stiffness values is shown by the colormap $\in [0.8, 1.2]$ Nm^{-1}. (right) Elongation stiffness, the stiffness values is shown by the colormap $\in [10, 100]$ mN m. Note that the bending stiffness has a discrete symmetry in the circumferential direction with periodicity $\phi = k\frac{\pi}{3}$.

Table 4.1. Estimated hyper-elastic and visco-elastic material parameters for the study case soft robot

	$i=1$	$i=2$	$i=3$	$i=4$	$i=5$	$i=6$
α	$2.23{\cdot}10^{+3}$	$1.74{\cdot}10^{+3}$	$-4.55{\cdot}10^{+2}$	$1.31{\cdot}10^{-3}$	$1.23{\cdot}10^{-2}$	$-2.29{\cdot}10^{-1}$
α_λ	$3.21{\cdot}10^{+2}$	$5.22{\cdot}10^{-1}$	$26.4{\cdot}10^{+1}$	$1.82{\cdot}10^{-4}$	$26.4{\cdot}10^{+1}$	$1.82{\cdot}10^{-4}$

order ordinary differential equation of the form

$$\frac{d}{dt}\begin{pmatrix}\lambda_1\\ \vdots\\ \lambda_n\end{pmatrix} = \begin{pmatrix}-\alpha_{\lambda,1} & \dots & 0\\ \vdots & \ddots & \vdots\\ 0 & \dots & -\alpha_{\lambda,2n-1}\end{pmatrix}\begin{pmatrix}\lambda_1\\ \vdots\\ \lambda_n\end{pmatrix} - \begin{pmatrix}\alpha_{\lambda,2}\\ \vdots\\ \alpha_{\lambda,2n}\end{pmatrix}\dot{\boldsymbol{q}} \tag{4.48}$$

where the vector $\boldsymbol{\alpha}_\lambda = (\alpha_{\lambda,1}, \alpha_{\lambda,2}, ..., \alpha_{\lambda,2n})^\top$ contains positive parameters that describe the visco-elastic material dynamics, and the state variables $\lambda_i(t)$ with $i \in \{1, 2, ..., n\}$, *i.e.*, the creep strains. Note that the state dimension of the creep strains is equivalent to the state dimension $\dim(\boldsymbol{q})$. Now, the dynamics of the visco-elastic creep can be easily included into (4.25) as an external disturbance $\boldsymbol{\delta}(t)$. Hence, consider the visco-elastic creep forces of the Kelvin-Voigt model to be characterized by

$$\boldsymbol{\delta}_c(\boldsymbol{\lambda}) = \boldsymbol{K}_\lambda^\top \boldsymbol{\lambda}, \tag{4.49}$$

where $\boldsymbol{K}_\lambda \succ 0$ denotes the creep compliance matrix, which is a linear mapping from creep strains to creep forces. Since creeping strains are difficult, if not impossible, to distinguish from the elastic strains alone, we must include dynamic experiments to properly identify the creep coefficients. Using a regression approach similar to Section 4.6.2, the creeping parameters $\boldsymbol{\alpha}_\lambda$ and the creep compliance $\boldsymbol{K}_\lambda$ are empirically estimated from experimental data (*e.g.*, unforced oscillations). In this regression, the full dynamics are simulated and the parameter $\boldsymbol{\alpha}_\lambda$ are optimized to minimize the dynamic residual in tip position of the soft robot. The estimated visco-elastic parameters are also provided in Table 4.1. The linear damping parameters from the Rayleigh damping matrix $\boldsymbol{R}$ and the initial conditions for $\boldsymbol{\lambda}(t_0)$ are identified similarly.

Example 4.3 (Kelvin-Voigt creep dynamics)**.** To highlight the dynamics of elastomer materials exhibiting creep, let us consider an rudimentary illustrative example of a 1-DOF mass-spring-damper system with a Kelvin-Voigt creep element. Let $\boldsymbol{x} = (x_1, x_2, x_3)^\top = (\varepsilon, \dot{\varepsilon}, \lambda)^\top$ be the state vector composed of the elongation strain, elongation rate, and the creep strain, respectively. Then, the dynamics can be written in the familiar state space form as follows:

$$\dot{x}_1 = x_1, \tag{4.50}$$

$$\dot{x}_2 = \frac{1}{m}\left[-kx_1 - cx_2 + k_\lambda x_3 + u\right], \tag{4.51}$$

$$\dot{x}_3 = -\alpha_{\lambda,1}x_3 - \alpha_{\lambda,2}x_2 \tag{4.52}$$

where $m, k, c > 0$ the mass, spring, damper coefficients; respectively, $k_\lambda > 0$ the creep stiffness, $\alpha_\lambda > 0$ the creep parameters, and $u : \mathbb{T} \to \mathbb{R}$ an auxiliary input

(*i.e.*, prescribed force). The following parameters are consider for this illustrative example: $m = 0.1$ kg, $k = 10$ Nm^{-1}, $c = 0.1$ Nsm^{-1}, $k_\lambda = \alpha_{\lambda,1} = 2$, and $\alpha_{\lambda,2} = 5$. To highlight the effects of visco-elastic creep, we choose a reference signal with fast-changing dynamics: $u(t) = 0.15 \cdot \text{sign}\,[\sin(0.2\pi t)]$. Given the input, the simulation results are shown in Figure 4.10. Note that the simulation result presents an unmodified Hookean model (—) in which $k_\lambda = 0$, and the Kelvin-Voigt variant (—). Clearly, we see a difference between the two trajectories. The original

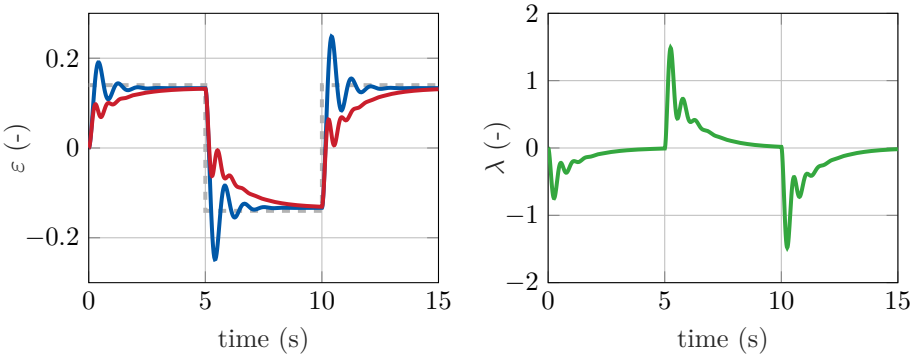


Figure 4.10. Simulation study of introducing visco-elastic Kelvin-Voigt dynamics to a mass-spring-damper system, where we show the evolution of the elongation ε subjected to an smooth block signal $u(t) = 0.15 \cdot \text{sign}\,[\sin(0.2\pi t)]$ in (—) compared between the original Hookean model (—) and visco-elastic Kelvin-Voigt model (—). Also, the evolution of the visco-elastic creeping strain λ is shown in (—).

Hookean model oscillates around the quasi-static equilibrium, whereas the Kelvin-Voigt variant slowly converges to the setpoint. Note that this is clearly different than the traditional overdamped response, as the Kelvin-Voight model does allow for oscillations during the smooth transient.

4.7 Adaptive control

As briefly discussed in the introduction, the dynamics model will be used as a control-oriented framework for model-based controllers applicable to soft robotics. In retrospect to previous model-based controllers, Della Santina et al. [60] proposed a combination of feedforward and model-based feedback; yet, satisfying the passivity condition, more robustness approach could be acquired through energy-based controller (especially in the face of material uncertainties). Franco et al. [70] proposed an adaptive energy-based controller but the underlying model (multi-link pendulum) is not rooted in a continuum description. Here, we wish to provide a

mix of the control methodologies – an energy-based control approach for the continuous PCC model with an adaptive material law.

4.7.1 Passivity-based adaptive control

The continuous dynamics of the soft robotic manipulator are described by (4.25), where the Lagrangian system matrices depend on physical parameters, e.g., mass, moments of inertia, stiffness, and viscosity. Within the context of robust control, these parameters often deviate from their true value. So merely an estimate of the system matrices $\widetilde{\boldsymbol{M}}(\boldsymbol{q})$, $\widetilde{\boldsymbol{C}}(\boldsymbol{q},\boldsymbol{q})$, $\widetilde{\boldsymbol{f}}_{\mathrm{g}}(\boldsymbol{q})$ and $\widetilde{\boldsymbol{f}}_{\mathrm{e}}(\boldsymbol{q},\dot{\boldsymbol{q}})$ can be acquired, where we denote $\Delta(\cdot) = \widetilde{(\cdot)} - (\cdot)$ as the difference between the true value and its estimate. The difference (or uncertainty) between true and estimated values is of particular relevance in soft robotics, where material properties play a significant role on both the statics and dynamics. Poor estimates of the material parameters could lead to instability in some model-based controllers if not considered carefully. Exploiting the passivity in Lagrangian models, we can derive a passivity-based adaptive controller similar to the works of Slotine et al. [192] and Ortega et al. [159]. The benefit of passivity-based control techniques is its robustness regarding parameter uncertainties and unmodelled dynamics. Passivity-based control is rooted in energy-shaping and damping injection techniques, leading to simple implementation yet effective means of stabilization.

Let $\boldsymbol{q}_d \in \mathcal{Q}$ be the desired trajectory of the soft robot together with its time-derivative $\dot{\boldsymbol{q}}_d, \ddot{\boldsymbol{q}}_d \in \mathbb{R}^n$. Next, let $\boldsymbol{\pi} \in \mathbb{R}^p$ be a vector containing all unknown values from a set of physical parameters, and the parametrization error $\boldsymbol{e}_p := \tilde{\boldsymbol{\pi}} - \boldsymbol{\pi}$ in which the the vector $\tilde{\boldsymbol{\pi}} \in \mathbb{R}^p$ denotes the parameter estimates. An important note is that the model must be linear in $\boldsymbol{\pi}$, which holds true for the line-density ρ_0, the linear elasticity parameters in (4.46) and (4.45), damping, creep coefficients in (4.48) and (4.49), and also for an unknown mass applied at the tip of the soft robot (4.26). Unfortunately, we cannot included the material parameters α_3 and α_6 due to their nonlinear dependence. The control objective is given by finding an appropriate control input and update law such that $\lim_{t\to\infty} \boldsymbol{q}(t) = \boldsymbol{q}_d(t)$ is achieved with good transient behavior. Assuming linearity in the parameters the linear parametrizability matrix of the soft robot's dynamics is given as follows

$$\boldsymbol{Y}(\cdot,\boldsymbol{\pi})\,\boldsymbol{e}_p = \Delta\boldsymbol{M}\,\ddot{\boldsymbol{q}}_r + \Delta\boldsymbol{C}\,\dot{\boldsymbol{q}}_r + \Delta\boldsymbol{f}_{\mathrm{g}} + \Delta\boldsymbol{f}_{\mathrm{e}} + \Delta\boldsymbol{\delta}, \tag{4.53}$$

where $\dot{\boldsymbol{q}}_r = \dot{\boldsymbol{q}}_d - \boldsymbol{\Lambda}\boldsymbol{e}$ is called the reference velocity vector, $\boldsymbol{\Lambda} \in \mathbb{R}^{n\times n}$ a positive diagonal matrix, and $\boldsymbol{Y}(\boldsymbol{q},\dot{\boldsymbol{q}},\dot{\boldsymbol{q}}_r,\dot{\boldsymbol{q}}_r,\boldsymbol{\pi}) \in \mathbb{R}^{m\times n}$ is called the regressor matrix. Following the work of Slotine and Li [192], the control law and adaptation law are

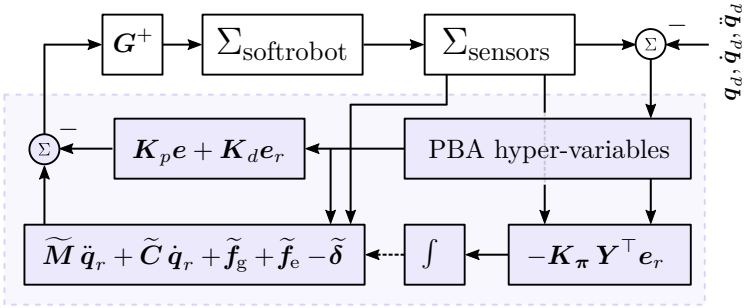


Figure 4.11. Schematic diagram of the passivity-based adaptive controller (PBAC), where $\Sigma_{\text{softrobot}}$ denotes the dynamical system (4.25) and Σ_{sensor} a system of sensors suitable of measuring $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$.

4

given by

$$\boldsymbol{\tau} = \widetilde{\boldsymbol{M}}\,\ddot{\boldsymbol{q}}_r + \widetilde{\boldsymbol{C}}\,\dot{\boldsymbol{q}}_r + \widetilde{\boldsymbol{f}}_{\text{g}} + \widetilde{\boldsymbol{f}}_{\text{e}} - \widetilde{\boldsymbol{\delta}} - \boldsymbol{K}_p\,\boldsymbol{e} - \boldsymbol{K}_d\,\boldsymbol{e}_r, \tag{4.54}$$

$$\dot{\widetilde{\boldsymbol{\pi}}} = -\,\boldsymbol{K}_{\boldsymbol{\pi}}\,\boldsymbol{Y}^\top \boldsymbol{e}_r, \tag{4.55}$$

where $\boldsymbol{e}_r := \dot{\boldsymbol{q}} - \dot{\boldsymbol{q}}_r = \dot{\boldsymbol{e}} + \boldsymbol{\Lambda}\,\boldsymbol{e}$, $\boldsymbol{K}_p, \boldsymbol{K}_d \in \mathbb{R}^{n\times n}$ are controller gains, and $\boldsymbol{K}_{\boldsymbol{\pi}} \in \mathbb{R}^{p\times p}$ is a positive definite matrix called the adaptation rate. Since $\boldsymbol{\tau}$ define the desired generalized forces acting on the system (4.25), the desired pressures are computed as $\boldsymbol{u} = \boldsymbol{G}^+\boldsymbol{\tau}$ with $\boldsymbol{G}^+$ the generalized inverse of $\boldsymbol{G}$. A schematic diagram of the passivity-based controller is provided in Figure 4.11. It should be mentioned that the magnitude of adaptation rate does not affect the global stability of the system (if unmodelled dynamics are not excited); however, it sets the rate of adaptation, and accordingly the performance of the system.

Remark 4.5 (Persistence of excitation) It is important to note that the convergence of the tracking error $\boldsymbol{e} \to 0$ does not imply convergence of the estimated parameter to their true values. According to [191–193], asymptotic convergence can be shown if the matrix $\boldsymbol{Y}(\boldsymbol{q}, \dot{\boldsymbol{q}}, \dot{\boldsymbol{q}}_r, \dot{\boldsymbol{q}}_r, \boldsymbol{\pi})$ is persistently excited and it is uniformly continuous. To elaborate, under the condition of persistent excitation, that is, for any instances t_1, t_2 with $t_1 \leq t_2$ there exists a positive constant α such that $\int_{t_1}^{t_2} \boldsymbol{Y}^\top \boldsymbol{Y}\, dt \preceq \alpha\, \boldsymbol{I}$, it can be proven that the parameter estimates converge asymptotically to their true values. The authors in [192] state that the proof for convergence here applied to nonlinear robot dynamics is similar to those of linear dynamics [150] although the proof is fairly involved.

4.8 Experimental platform

Before invest, we detail the experimental setup and control platform of the soft robotic system. Since sensing is a challenging issue in soft robotics, due to large distributed deformations, a combination of sensors were used to recover an estimate of the states $\boldsymbol{q} \in \mathcal{Q}$. A full overview of the setup is given in Figure 4.12. First of all, we employed a 6-DOF inertial measurement unit (MPU-6050, `InveSense`) that measures the angular displacement of the soft robot's end-effector (*i.e.*, $\sigma = L$). Through on-board sensor fusion, the bending angle of the soft robot can be recovered, *i.e.*, $\beta = \kappa l$. Since the bending angle alone is not sufficient to decouple the curvature and elongation, additional sensing is required. Consequently, we use a stereo-vision depth camera (RealSense D435, `Intel`) with an infrared dot projector and RGB camera module. A spherical optical marker is attached to the end-effector of the soft robot, whose relative position can be recovered using a combination of depth-sensing and image post-processing with a Hough-space circle transformation. To retrieve a global reference frame of the vision system, four 30× 30 mm Aruco marker are uniformly distributed whose location and orientation can be found using the `opencv-python.py` library. We show the implementation of the optical vision system and its post-processing in Figure 4.13. Through trigonometry and the measured bending angle β, an filter measurement of the position vector $y = \widetilde{\boldsymbol{\gamma}}_L$ can be recovered. Given the analytic expressions for the orientation and position in (4.12) and (4.13), an inverse Jacobian kinematic solver is employed to recover an estimate of the state vector, *i.e.*, $\widetilde{\boldsymbol{q}}_{\text{dyn}} = \text{argmin}_{\boldsymbol{q}} \|\widetilde{\boldsymbol{\gamma}}_L - \boldsymbol{\gamma}(L, \boldsymbol{q})\|_2$. During each experimental trail, it was made sure the soft robotic body does not occlude the optical marker.

As for the pneumatic actuation, an array of proportional-pressure regulators (VEAB-B-D16, `Festo`) was used with an active pressure range of -0.1 MPa $< \boldsymbol{u}(t) \leq 0.1$ MPa, which simultaneously allow for pressure measurements. These measurements are fed into the (quasi-static) model to also recover a quasi-static estimate of the states $\widetilde{\boldsymbol{q}}_{\text{qs}}$. Then, the dynamic estimates $\widetilde{\boldsymbol{q}}_{\text{dyn}}$ and the quasi-static pressure-based estimates $\widetilde{\boldsymbol{q}}_{\text{qs}}$ are fused using an ordinary complementary filter. The control and data acquisition are done using a Raspberry Pi 4 (2GB).

4.9 Numerical and experimental implementation

In this section, we will discuss the simulation results of the dynamic model (4.25), the passivity-based controller (4.54), and the adaptive law (4.55). To illustrate effectiveness and performance of the approach, we segregate our analysis into several study-cases of various complexity. First, focusing on the physical one-link

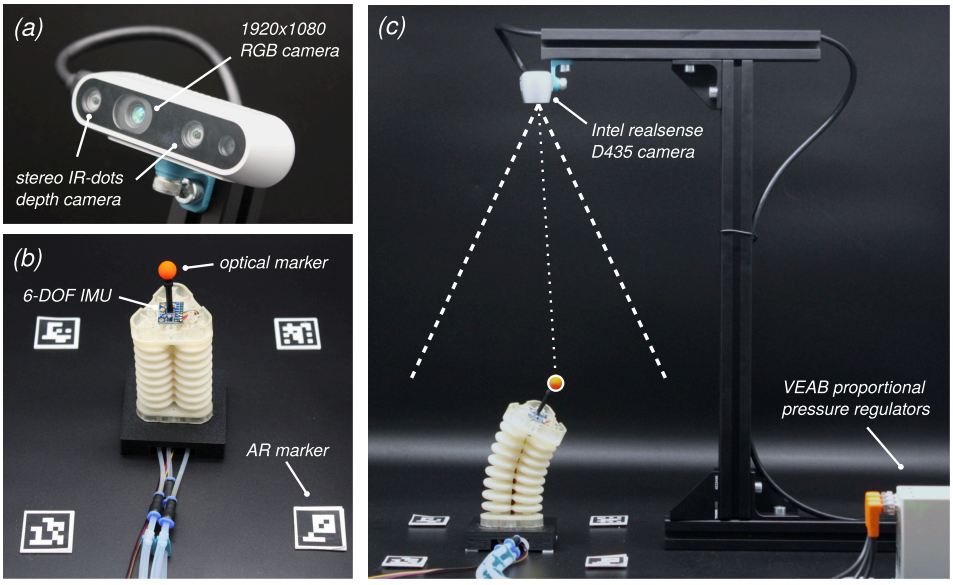


Figure 4.12. General overview of the experimental platform for the testing and development of the 3-DOF soft manipulator. (a) Close-up of the RealSense D435 stereo-vision depth camera. (b) Soft continuum manipulator with pressure inputs $\boldsymbol{u} = (u_1, u_2, u_3)^\top$, a MPU-6050 Inertial Measurement Unit (IMU) to measure the end-effector angle β, and a color-coded optical marker to recover $\boldsymbol{\gamma}_L$, and four Aruco markers to recover Φ_0 and $\boldsymbol{\gamma}_0$. (c) Overview of full setup with the array of three VEAB-B-D16 pressure regulators.

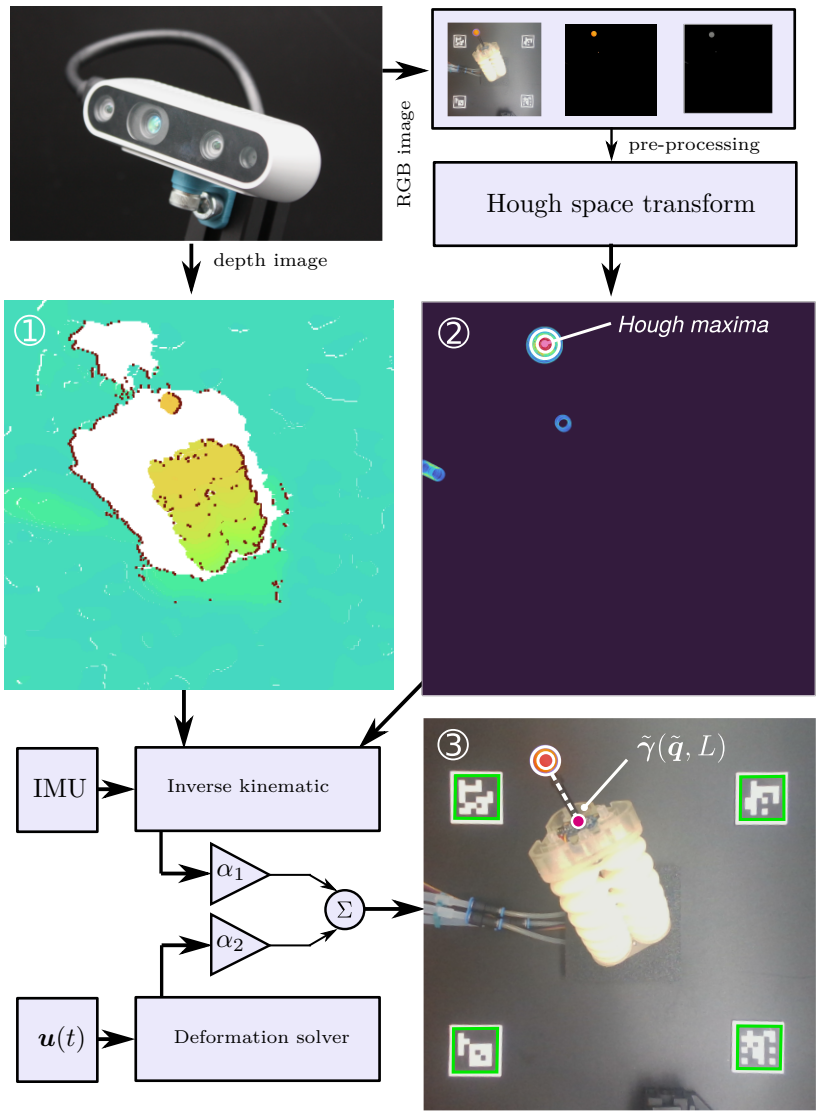


Figure 4.13. General overview of the optical vision system used to estimate the state variables q. ① First, an RBG image is processed using a Circular Hough transformation filtering for circles with a radius 32 `pix`, it has a global maxima at the optical marker. ② Then, a sample of the depth camera is generated. ③ Finally, all sensor data is combined using a sensor fusion algorithm of depth, RGB camera, input pressures and IMU data, resulting in an accurate estimate of soft manipulator's end-effector $\tilde{\gamma}(L,\tilde{q})$.

soft robot in Figure 4.1 ($N = 1$), we investigate the unforced system's equilibria and their corresponding stability. In continuation, we compare the simulated trajectories of the dynamical model with experimental data for natural oscillations, forced pneumatic inputs, and external loading conditions; where we also highlight contribution of the hyper-elastic FEM-driven material model. Second, to illustrate the flexibility and computational efficiency of the numerical framework, we extend the one-link model to a multi-link model with $N = 6$ soft-bodied links.

The numerical solutions to the ordinary differential equations in (4.25) together with (4.54) and (4.55) are computed using the aforementioned MDE integration scheme which is developed in *MATLAB* , and the underlying code can be found at Caasenbrood et al. [36]. The software architecture is compactly written as Object-Oriented class labeled under `./src/Model.m` that enables a minimal programming interface to set-up various soft robotic simulation models easily. The simulation results provided in this section can be reproduced using the open-source *Sorotoki* package found at [33]

Natural dynamics – One-link soft robot: The following physical parameters are chosen for the soft robot: the mass $m_0 = 17.3$ g, the relaxation length $L = 64.4$ mm. The material parameters for hyper-elasticity and visco-elasticity models are chosen identical to Table 4.1. For the additional viscous material behavior, the Rayleigh damping matrix and the creep compliance matrix are chosen a follow:

$$\boldsymbol{R} = \begin{pmatrix} 0.01 & 0 & 0 \\ 0 & 1.05 \cdot 10^{-5} & 0 \\ 0 & 0 & 1.05 \cdot 10^{-5} \end{pmatrix};$$

$$\boldsymbol{K}_\lambda = \begin{pmatrix} 502.3 & 0 & 0 \\ 0 & 1.53 \cdot 10^{-2} & 0 \\ 0 & 0 & 1.53 \cdot 10^{-2} \end{pmatrix}.$$

We stress that the values for the Rayleigh damping and creep compliance shown above are identified empirically through open-loop measurements, similar to the creep coefficient provided in Table 4.1. We will explain how these coefficients are derived later in this chapter.

First, we investigate the existence and the stability of the equilibria of the unforced system. If the system is at rest (i.e., $\dot{\boldsymbol{q}} = 0$, $\ddot{\boldsymbol{q}} = 0$), then by definition there are no conservative forces acting on the system. Thus, for any equilibrium point $\boldsymbol{q}_0$ it holds that $\nabla \mathcal{U}(\boldsymbol{q}^\star) \equiv \boldsymbol{0}$. If $\mathcal{U}(\boldsymbol{q}^\star) \equiv E_0$ is a local minimum, then the equilibrium is deemed stable. Any small disturbance will result in a new energy-state E_1 and will consequently bring the system in motion. However, regarding E_0 is a local minimum, the system will remain in a neighborhood of $\boldsymbol{q}^\star$ and eventually converge

towards its nearest low-state energy E_0. If $\mathcal{U}(\boldsymbol{q}^\star) \equiv E_0$ is a local maximum, the equilibrium is deemed to be unstable, since there exist a configuration close to $\boldsymbol{q}^\star$ with a lower energy-state, *i.e.*, $\mathcal{U}(\boldsymbol{q}^\star + \delta\boldsymbol{q}) < E_1$. By analysis of the gradient of the potential energy function $\nabla\mathcal{U}(\boldsymbol{q})$, two unique equilibria can be found numerically. The potential function has a local maximum for $\boldsymbol{q}^\star_{\text{unstab}} = \left(-\frac{m_0 g}{L(\alpha_1-\alpha_2)},\, 0,\, 0\right)^\top$ which is unstable. To some extent, it is analogous to the unstable equilibrium position of the inverted pendulum system.

For the stable equilibria, the bisection method was used to find the zero-crossing of $\nabla\mathcal{U}(\boldsymbol{q})$, where it was found that all stable solutions of the unforced system will tend to the following set:

$$\Omega_{\text{stab}} = \left\{ \boldsymbol{q} \in \mathcal{Q} \;:\; \varepsilon = -\varepsilon_\star,\, \kappa(\boldsymbol{q}) = \frac{\kappa_\star}{\alpha_\phi(\boldsymbol{q})} \right\},$$

where $\varepsilon_\star$ and $\kappa_\star$ are nonzero constants. Notice that the set Ω_{stab} is topologically equivalent to a ring. This set corresponds to the hanging position of the soft robot. Given the physical parameters of the robot in Figure 4.1, the following constants are found: $\varepsilon_\star = 0.0021$ and $\kappa_\star = 0.0174$. It is important to note that the stable set of equilibria stems from the force balance between the internal elastic potential forces and the external gravitational potential forces, and thus any stiffness will lead to a stable set with a similar topology. By changing the base orientation of the soft manipulator (*i.e.*, by modifying $\boldsymbol{\Phi}_0$), both equilibria vanish and all state trajectories will tend to a global stable equilibrium. For fully reversing the orientation, this trivially leads to the stable equilibrium $\boldsymbol{q}^\star_{\text{stab}} = (+\frac{m_0 g}{L(\alpha_1-\alpha_2)},\, 0,\, 0)$. This phenomenon is referred to as local bifurcation, in which the change of parameter values alters the existence and stability of equilibria. This property might be interesting for soft robot manipulators with multiple soft-bodied links, as they are likely to be subjected to different gravitational loads.

To illustrate the unforced dynamics and the existence of stable equilibria, time-domain simulations of the dynamical model with nonzero initial conditions:

$$\boldsymbol{q}_0 = (0,\, -15,\, 15\,)^\top,$$
$$\dot{\boldsymbol{q}}_0 = (0,\, 2500,\, 0)^\top.$$

Figure 4.14 shows the state trajectories of the soft robot; whereas Figure 4.15 is provided to better illustrate the underlying dynamics and the trajectory of the end-effector.

Besides the existence of stable solutions, the numerical simulations perfectly illustrate the coupled dynamics between the elongation and bending of the soft robot. Due to the difference in mechanical stiffness for elongation and bending, we

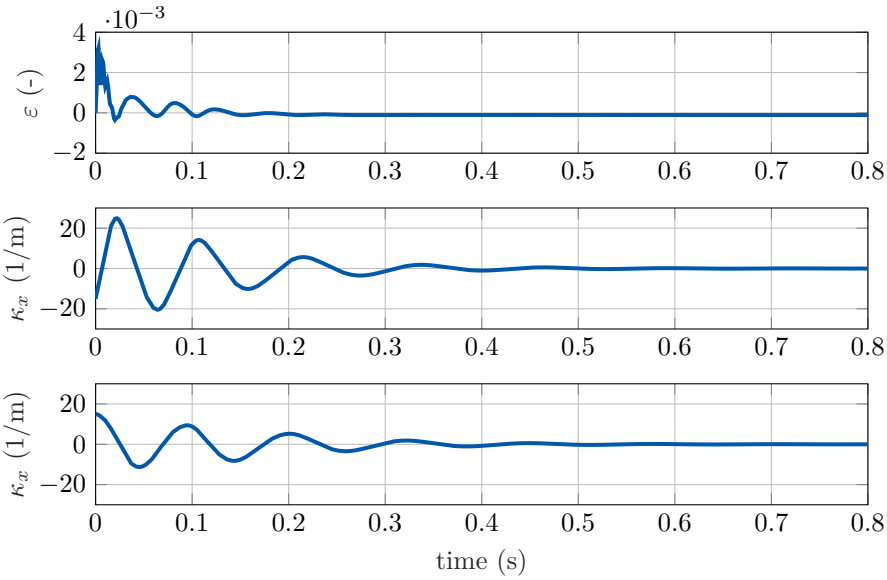


Figure 4.14. State trajectories of one-link soft robot model with initial conditions $\boldsymbol{q}_0 = (0, -15, 15)^\top$ and $\dot{\boldsymbol{q}}_0 = (0, 2500, 0)^\top$. The figure shows the elongation strain ε and the curvatures κ_x, κ_y in the xz-plane and yz-plane, respectively. Clearly the one-link soft robot oscillates about the set Ω_{stab}.

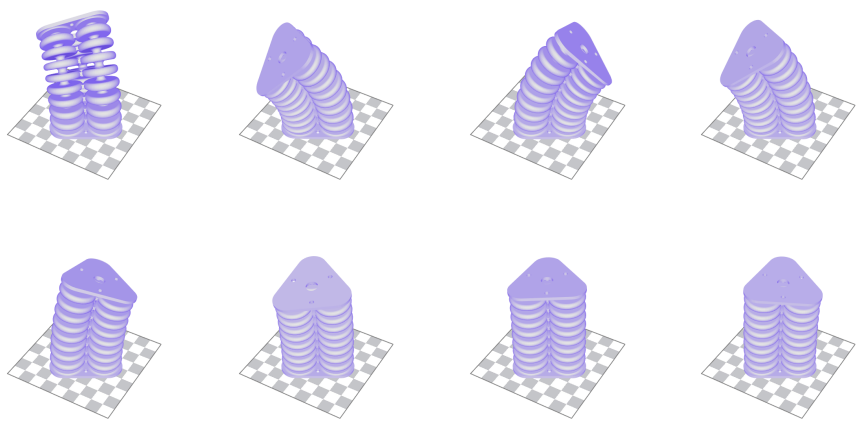


Figure 4.15. Three-dimensional volumetric evolution of the one-link soft robot model with initial conditions $\boldsymbol{q}_0 = (0, -15, 15)^\top$ and $\dot{\boldsymbol{q}}_0 = (0, 2500, 0)^\top$. Notice that the states of the one-link soft robot quickly converge to the the set of stable equilibria Ω_{stab}.

observe high-frequency and low-frequency oscillation for the elongation strain $\varepsilon(t)$, and we observe low-frequent oscillations for the curvatures $\kappa_x(t)$ and $\kappa_y(t)$. Interestingly, the low-frequency oscillations are passed from the curvature dynamics to elongation dynamics; conversely, the dynamics of the elongation barely affect the curvatures. After sufficient time passes, the trajectories indeed tend to the set of stable equilibria Ω_{stab}.

Experimental comparison– unforced, forced, and external loads: To validate the dynamic model, the solutions of the model are compared with measurements of the physical system in unforced, forced, and tip-load conditions. As such, the model validation is separated into three parts: *i)* unforced, *ii)* forced conditions, and *iii)* external tip-loads applied on the end-effector.

We start with the unforced scenario, *i.e.*, no input is considered $u_i(t) \equiv 0$. For the unforced analysis, two experimental trails are performed for the unforced validation. First, the soft robot is deformed slightly and then released from rest, which corresponds to the initial conditions $\boldsymbol{q}_0 = (0.015, 4.75, 0)^\top$ and $\dot{\boldsymbol{q}}_0 = \mathbf{0}_3$. Since the mechanical deformations are relatively small here, the presence of hyper-elastic and visco-elastic material behavior are less dominant. Secondly, the soft robot is moderately deformed such that the initial configuration (or shortly after) lies within the hyper-elastic and visco-elastic regime. In this scenario, the non-linear and time-dependent material effects may not be neglected. These initial conditions correspond to $\boldsymbol{q}_0 = (0.046, 11.25, 0)^\top$ and $\dot{\boldsymbol{q}}_0 = \mathbf{0}_3$. It is worth mentioning that the creep strains $\boldsymbol{\lambda}$ are difficult to distinguish from the true strain, and thus the initial conditions for $\boldsymbol{\lambda}(t_0)$ are determined empirically using a set of dynamic measurements with different initial condition. The comparison between our model and the unforced dynamic measurements are shown in Figure 4.16.

As can be seen, the state trajectories of the end-effector closely match the ground truth trajectories, even for significant nonlinear deformation. For the first test run (inside linear elastic regime), the RMS error and the maximum error are ± 0.19 and ± 0.50 degrees, respectively. For the second case (outside the linear elastic regime), the RMS error and the maximum error is ± 0.78 and ± 2.33 degrees, respectively.

Second, we consider a forced scenario in which a regulated pressure input is applied to the pneumatic bellows, *i.e.*, $\boldsymbol{u} \neq \mathbf{0}_3$. Since the pneumatic mapping $\boldsymbol{G}$ in (4.27) plays an important role here, the actuator coefficients are recomputed to match the experimental data better. To be more specific, by considering a pre-defined set of excitation signals $u(t)$ of various amplitudes and frequencies, a least-squares optimization routine is employed that minimizes the difference between the measured states $\hat{q}$ with the simulated states q by tuning the coefficients α_ε and α_κ. This leads to the following values: $\alpha_\varepsilon = 2.34 \cdot 10^{-7}$ and $\alpha_\kappa = 1.61 \cdot 10^{-8}$. As

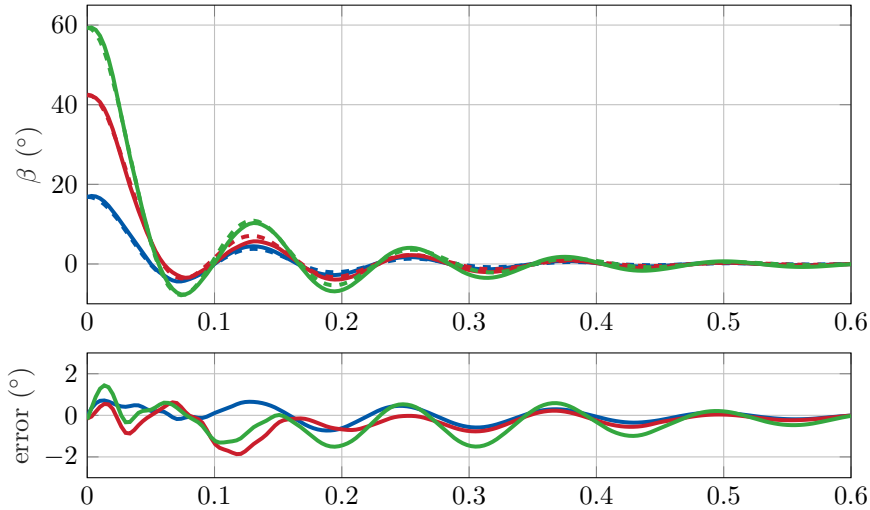


Figure 4.16. Experimental comparison results of the dynamic model in unforced conditions. (top) Bending angles of the soft robot in unforced conditions, where the dashed lines represent the experimental measurements and the solid lines are the simulated trajectories. The dataset (—) are within the linear elastic regime whereas datasets (—, —) are in the nonlinear regime. (bottom) The error between experiments and the optimized numerical model. As can be seen, all trajectories remain within an error bound of ±2° degree.

for the excitation signal, we have chosen the following input:

$$u_i(t) = P_0 + P_A \left[\frac{1}{2} + \frac{1}{2}\sin(t + \phi_i)\right] \cdot \max(0.05t, 1), \tag{4.56}$$

with a static offset $P_0 = 10$ kPa, an amplitude $P_A = 25$ kPa, and a phase offset $\phi_i = (i-1)\frac{2\pi}{m}$ rad. To highlight the significance of the proposed hyper-elastic modeling approach, we also compare the results using an optimized Hookean material material model with $k_e = 50.6$ N/m and $k_b = 5.8 \cdot 10^{-4}$ Nm/rad. The initial conditions are set to zero. The validations results for both the FEM-driven hyper-elasticity model and linear model in the forced setting are shown in Figure 4.17. The figure also shows the measured outputs from the pneumatic VEAB regulators $u_i(t)$, which are directly fed into both linear and hyper-elastic models.

Given these results, two key observations can be made. First, both the linear Hookean and hyper-elastic models provide reasonable accuracy for small deformations $0 \leq \kappa(t) \leq 3$ with a RMS error of ±0.13 and ±0.16 in curvature, respectively. However, as deformations exceed the linear regime, the hyper-elastic model signifi-

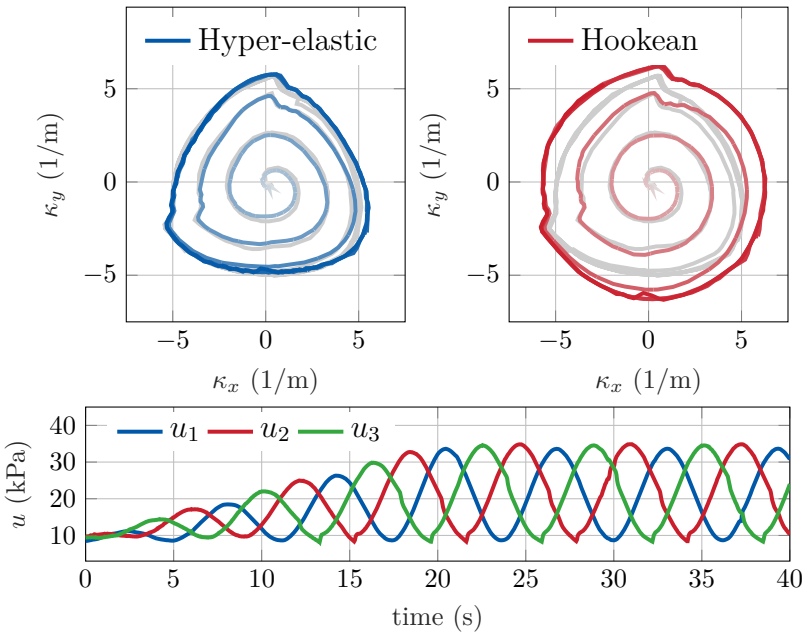


Figure 4.17. Spatial representation of the nonlinear stiffness, *i.e.*, elongation stiffness (left) and bending stiffness (right) for a radially distributed sampling of the curvature joint space $(\kappa_x, \kappa_y) \in [-\pi, \pi)$. The experimental data is shown in (—). Again, note that the bending stiffness has a discrete symmetry in the circumferential direction with periodicity of $\frac{2\pi}{3}$.

cantly outperforms the Hookean model. Focusing on the hyper-elastic model, both the asymmetric stiffness in radial direction and the strain-hardening are captured well, where the the linear models is not sufficiently rich to capture the material effects. The overall RMS errors for the linear and hyper-elastic model are ± 1.79 and ± 0.21 in curvature, respectively. Regarding the end-effector accuracy, the overall RMS errors for the linear and hyper-elastic model are ± 2.58 and ± 0.65 mm, respectively; which translates to an arc-length normalized error of $\pm 4.09\%$ and $\pm 1.03\%$. These numerical comparison results show that introducing nonlinear elastic effects driven by FEM-data can further improve the accuracy for a larger region of the soft robot's workspace.

For the last validation case, we subject the soft robot to an external payload of mass δ_m located at the end-effector. To model the disturbance, we use the expression for the external payload disturbance model δ_m in (4.26). The goal here is *a)* to demonstrate the accuracy of the proposed payload model and quasi-static behavior of the dynamic model, and *b)* to highlight the limitations of the PCC

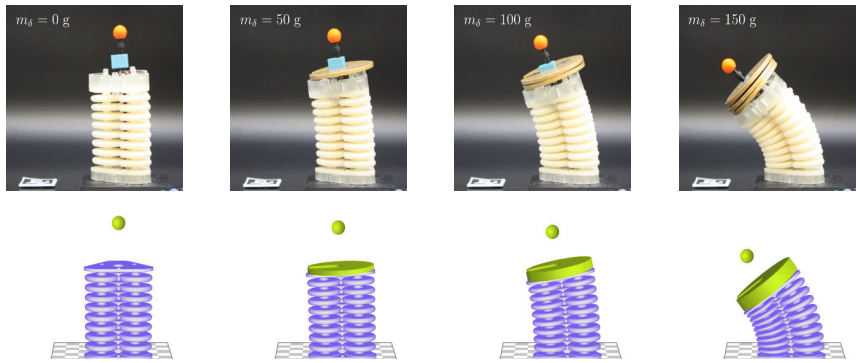


Figure 4.18. Experimental validation of the one-link soft robot subjected to various end-effector payloads of different mass $m_\delta = \{0, 50, 100, 150\}$ g. The deformed 3D-model in (—) corresponds to the estimated joint configuration based on the measurements from the IMU and depth camera, where the optimal marker location is shown in (—).

assumptions under certain conditions. In this analysis, we consider three different payloads $m_\delta = \{0, 50, 100, 150\}$ g. The experimental results of the payloads deformations and the resulting quasi-static deformations of the dynamic model are shown in Figure 4.18.

As can be seen, the quasi-static behavior of the dynamic model matches the experimental results relatively well for smaller payloads. For the mass $m_\delta = 0.05$ kg, the Euclidean error between the model and the measurement are ± 1.31 mm. Increasing the payload to $m_\delta = 0.1$ kg leads to an error of ± 2.15 mm. We can also clearly observe that the estimate of the backbone curve subject to the PCC condition is beginning to deviate from the ground truth yet the overall shape still matches the experimental data. Lastly, by further increase the payload $m_\delta = 0.15$ kg, we clearly observe the limitations of the PCC condition under external loads – with an end-effector error of 3.98 mm. Also, there is a clear discrepancy in the backbone curve of the model and the ground truth, which might imply the PCC condition is no longer valid here as the payload could induce non-constant curvatures along the backbone. A possible solution might be to introduce a different shape parametrization, similar to the works [27, 46, 60, 169].

Model benchmark – Multi-link soft manipulator case: In this section, we benchmark the proposed numerical integration scheme for a dynamic model of a six-link soft robot manipulator ($N = 6$). Here, we want to highlight that sufficient numerical speed can be obtained while preserving sufficient numerical

precision. This is an important criteria for model-based control, as slow numerical models lack transferability from theory to application. As mentioned earlier, the numerical integration of the Lagrangian entities is the computational bottleneck. Although the MDE solver does aid with numerical performance; ultimately, using a balanced spatial and temporal discretization impacts real-time performance the most. Trivially, using larger stepsize – both in space and time – lead to a decrease in numerical precision; and in some cases numerical instability. In this benchmark, we investigate these effects by varying two solver parameters: the spatial stepsize of the explicit MDE solver denoted by $\Delta\sigma$ and the temporal stepsize of the implicit trapezoidal solver denoted by Δt. For convenience, we represent these stepsizes as standardized parameters: the number of finite elements $N_s = L/\Delta\sigma$ and the implicit solver frequency $f_s = \Delta t^{-1}$. For the benchmark, we choose a total length of $L = 0.15$ m and simulation time of $T = 10$ s.

The extension to the multi-link soft robot ($N = 6$) can be described by a following generalized coordinates with the following structure:

$$\boldsymbol{q} = \left(\varepsilon_1,\ \kappa_{x,1}, \kappa_{y,1}, ..., \varepsilon_6,\ \kappa_{x,6}, \kappa_{y,6} \right)^\top \in \mathcal{Q} \tag{4.57}$$

To ensure the soft manipulator is self-supporting, we introduce slight variations to the hyper-elastic stiffness, link lengths, and the inertial properties of the dynamical system. Considering homogeneity, all links are chosen identical in length and mass: intrinsic link length $L_i = 0.025$ m and mass $m_i = 0.05$ kg. Next, the bending stiffness is slightly altered where we choose $\alpha_3 = 0.425$ Nm/rad and $\alpha_4 = 0.4$ Nm/rad. Please note that the material domain is now given by $\mathbb{X} = [0, \sum_{i=1}^{N} L_i]$. To introduce some interesting dynamics for the benchmark, we purely excite the first link of the serial-chain soft robot manipulator with a harmonic input:

$$u_i(t) = \begin{cases} P_a \cos(\pi t) & \text{for } i = 1, \\ 0 & \text{otherwise}, \end{cases} \tag{4.58}$$

where $P_a = 125$ kPa is the pressure amplitude. As for the pneumatic mapping that converts pressure to joint torques, we choose $\boldsymbol{\tau} = (\boldsymbol{G} \otimes \boldsymbol{I}_6)\, \boldsymbol{u}$ as the corresponding pneumatic map for the six-link soft manipulator. In total 36 benchmark simulations with different solver settings were performed and tested for their precision relative to a high-precision model ($f_s = 500$ Hz and $N_s = 500$). The state trajectories of the high-precision model are shown in Figure 4.19, whereas Figure 4.20 is provided to highlight the underlying dynamics and the trajectory of the end-effector. The results for all benchmark simulations are shown in Table 4.2.

Let us first discuss the dynamics of the six-link soft manipulator subjected to a harmonic input. Given this relatively straightforward harmonic excitation, some interesting (stable) nonlinear dynamics appear. Although we excite the

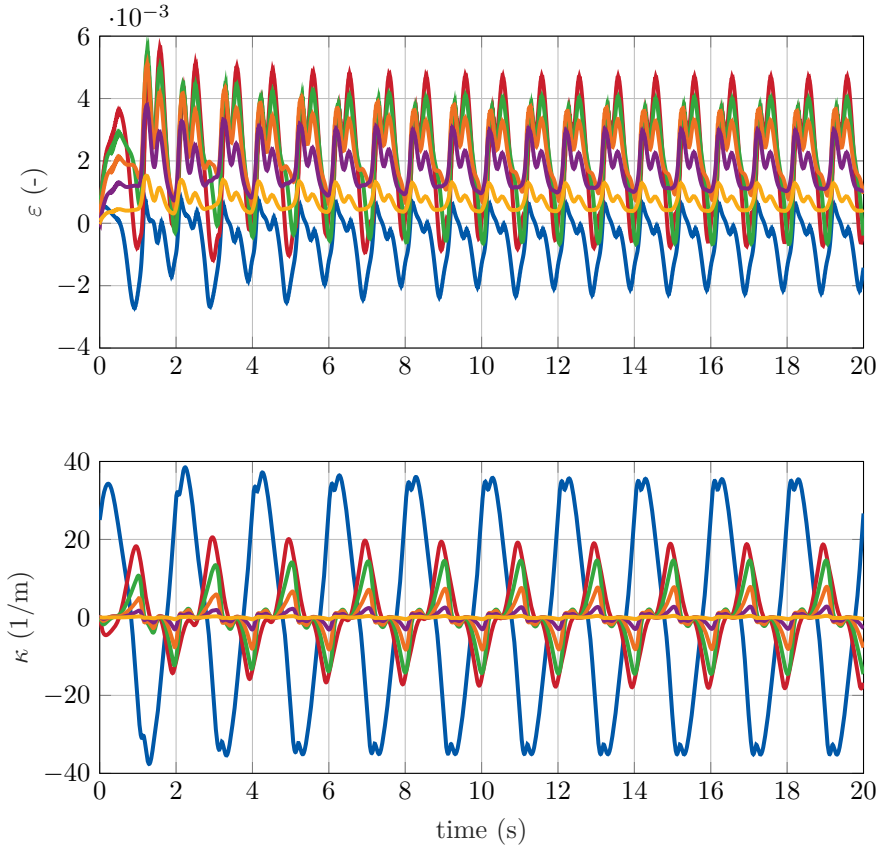


Figure 4.19. State trajectories of six-link soft robot model under dynamic excitation. The figure shows the extensible elongation strains and the total curvature κ (*i.e.*, planar dynamics). The link indexing follows (—, —, —, —, —, —).

system using one harmonic, the dynamics of the soft robot show a rich collection of harmonic oscillations – highlighting its nonlinear nature.

Remark 4.6 After a short transient time (i.e., $t < 5$), the solutions of the multi-link soft robotic system tend to a so-called *periodic solution*. Here, a solution is called periodic if there exists a period time $T_c > 0$ such that $\boldsymbol{q}(t) = \boldsymbol{q}(t + T_c)$ for all time t. Similar observations of the existence of periodic solutions (and control of such oscillations) were reported by Della Santina et al. [58] for articulated soft robots. Given the harmonic excitation in (4.58), the period time here is $T_c = 1$.

Now, we can exploit these periodic solutions for benchmarking the numerical solver in which we compare the solution of a high-resolution model (i.e., ground

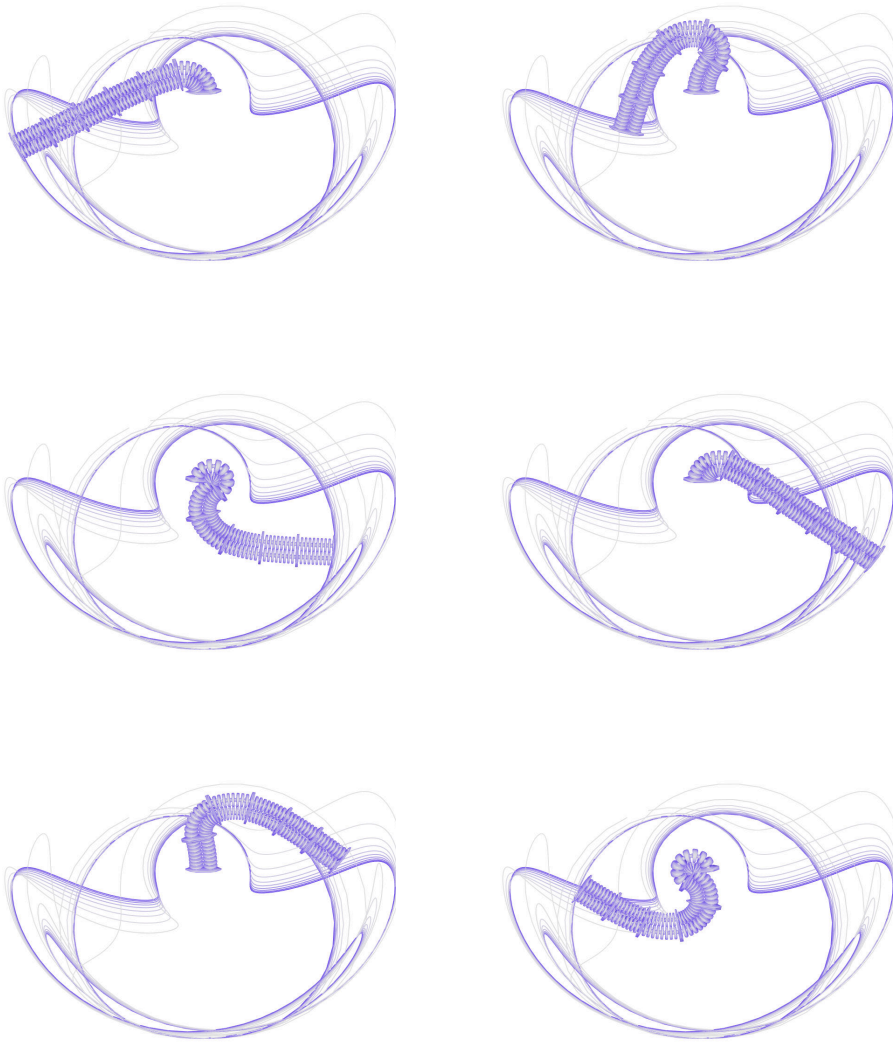


Figure 4.20. The deformed 3D-model of the six-link soft robot according to the input excitation $\boldsymbol{u}$. Notice that the dynamics of the six-link robot converge to a periodic solution which is shown in curve (—). Interestingly, the periodic orbit is slightly asymmetric with respect the sagittal plane at the origin, which is caused by the *hanging-down* initialization of the soft arm at the left side.

truth) to the benchmark solution. Let us define the benchmark error as the Euclidean distance between the two periodic solutions:

$$\begin{aligned} e &= \int_0^{T_c} \left\| \widetilde{\boldsymbol{\gamma}}(L, \widetilde{\boldsymbol{q}}(t)) - \boldsymbol{\gamma}(L, \boldsymbol{q}(t)) \right\|_2 dt, \\ &\approx \frac{1}{W} \sum_{i=1}^{W} \left\| \widetilde{\boldsymbol{\gamma}}(L, \widetilde{\boldsymbol{q}}(t_i)) - \boldsymbol{\gamma}(L, \boldsymbol{q}(t_i)) \right\|_2, \end{aligned} \tag{4.59}$$

where W is the number of time samples of the benchmark model inside the time interval of the periodic solution, and $\widetilde{\boldsymbol{\gamma}}(\cdot, L)$ and $\boldsymbol{\gamma}(\cdot, L)$ the tip position of the benchmark and the ground truth, respectively. The index W naturally depends on the sampling frequency of the implicit solver. Table 2 shows the normalized errors (i.e., the tracking error e normalized with the manipulator length L) together with the effective computation times of the numerical solver.

The benchmark table above gives some useful insight into which settings benefit numerical precision and speed the most. First, regarding the coarser models $N = 24$ (*i.e.*, 4 elements per link), we observe large numerical errors of $> 10\%$ independent of solver frequency. Although one could argue these settings grantee real-time performance (2s of simulation time for 1s computation at $N_s = 24$, $f_s = 75$ Hz) and might suffice for model-based control in slow settings (*i.e.*, set-point stabilization), they are most likely unsuited for dynamic tracking objectives. Moving towards $N_s = 60$ (*i.e.*, 10 elements per link), we observe a significant improvement in numerical precision $\pm 4\%$ and while still retaining real-time performance (1.6s of simulation time for 1s computation time at $f_s = 50$ Hz). As such, these settings show that both numerical precision and real-time computation can be achieved for more complicated multi-links soft robots. However, regarding the high-accuracy models $N_s > 100$, we see only a slight increase in numerical precision $\pm 3\%$ yet lose real-time capabilities. Consequently, these models might be suited for offline simulations, but lack transferability to online model-based control. However, a possible solution here might be to convert the `MATLAB` model into a `C` or `C++` equivalent model to further increase numerical speed [84, 178].

Closed-loop – Passivity-based controller under material certainty and mass disturbance: In the last numerical analysis, we demonstrate the results of the proposed passivity-based adaptive scheme. Due to its compactness and computational speed, we again consider the model of the one-link soft robot as seen in previous simulations. To illustrate the robustness of the controller, we purposely introduce some uncertainties. First, we introduce uncertainties in the hyper-elastic material parameters that deviate moderately from their true values in Table 1. Second, we introduce a payload to the end-effector of mass $m_\delta = 0.1$ kg. We again model this external disturbance using the relation in (4.26). Fol-

Table 4.2. Benchmark results of the six-link soft robot manipulator for various temporal and spatial discretizations ($T = 10$ s). The tables shows the mean tracking error of the end-effector relative to a ground truth ($f_s = 500$ Hz and $N_s = 500$ elements). The RMS errors are normalized with the total length $L = 0.15$ (i.e., the errors are presented in %). The CPU times are also given; where the highlighted entries achieve real-time computation (consistently).

$T = 10$ s	$N_s = 24$	$N_s = 36$	$N_s = 60$	$N_s = 90$	$N_s = 120$	$N_s = 180$
$f_s = 25$ Hz	19.0 / 1.77s	9.37 / 2.54s	5.36 / 4.06s	4.63 / 5.98s	4.62 / 7.93s	4.43 / 11.75s
$f_s = 50$ Hz	17.5 / 3.50s	7.04 / 5.04s	4.33 / 6.73s	3.69 / 11.60s	3.60 / 15.3s	3.69 / 22.71s
$f_s = 75$ Hz	13.77 / 5.31s	5.80 / 7.56s	4.51 / 12.41s	4.22 / 17.49s	3.68 / 22.35s	3.72 / 33.33s
$f_s = 100$ Hz	14.73 / 6.48s	6.12 / 9.10s	4.42 / 14.54s	3.71 / 21.46s	3.53 / 28.35s	2.89 / 42.03s
$f_s = 150$ Hz	15.22 / 7.26s	5.73 / 9.78s	4.32 / 15.58s	3.46 / 23.07s	3.16 / 30.38s	2.67 / 45.06s
$f_s = 250$ Hz	14.61 / 11.98s	5.32 / 16.71s	4.14 / 27.11s	3.35 / 40.19s	2.86 / 53.06s	2.21 / 78.61s

lowing the linear parametrizability of the uncertainties, the parameter estimation vector yields:

$$\widetilde{\boldsymbol{\pi}}(t) = (\tilde{\alpha}_1,\, \tilde{\alpha}_2,\, \tilde{\alpha}_4,\, \tilde{\alpha}_5,\, \tilde{m}_\delta)^\top, \tag{4.60}$$

with initial conditions $\widetilde{\boldsymbol{\pi}}(t_0) = \widetilde{\boldsymbol{\pi}}_0 = (0.75\alpha_1,\, 0.75\alpha_2,\, 0.45\alpha_4,\, 0.45\alpha_5,\, 0.65m_\delta)^\top$. Furthermore, the feedback gains and the adaptation rate are chosen as diagonal matrices as follows:

$$\boldsymbol{K}_p = \begin{pmatrix} 5 & 0 & 0 \\ 0 & 5\cdot 10^{-5} & 0 \\ 0 & 0 & 5\cdot 10^{-5} \end{pmatrix}, \qquad \boldsymbol{K}_d = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1\cdot 10^{-5} & 0 \\ 0 & 0 & 1\cdot 10^{-5} \end{pmatrix},$$

$$\boldsymbol{K}_\pi = \begin{pmatrix} 5\cdot 10^{3} & 0 & 0 \\ 0 & 2\cdot 10^{-6} & 0 \\ 0 & 0 & 2\cdot 10^{-6} \end{pmatrix};$$

and $\boldsymbol{\Lambda} = \boldsymbol{I}$. These values were found to yield the best performance while avoiding noticeable oscillations in the closed-loop dynamics. In this controller gain tuning process, we first find suitable values for $\boldsymbol{K}_p$ and $\boldsymbol{K}_d$ such that controller stabilizes the closed-loop and the error transients are within acceptable time windows (e.g., converge within < 5 s). During this process, the adaption gain is chosen zero. Once $\boldsymbol{K}_p$ and $\boldsymbol{K}_d$ are found, we slowly increase the adaptation gain. Again, we observe the transients of the parameter estimates. During the tuning of the adaptive controller, it was observed that a high-gain $\boldsymbol{K}_{\boldsymbol{\pi}}$ diminishes performance. Better convergence was found for low adaption gains – resulting in slowly-varying parameter estimates. Lastly, the following reference trajectory is considered:

$$\boldsymbol{q}_d(t) = (0.01 + 0.01\,\sin(t),\, 30\,\sin(t),\, 30\,\cos(t))^\top, \tag{4.61}$$

Since the reference trajectory above satisfies persistence of excitation (see Remark 4.5), it should grantee the all convergence of the hyper-elastic material estimation and the unknown mass contribution. Figure 4.21 shows the state trajectories of the soft robot, and Figure 4.22 shows the evolution of the nonlinear stiffnesses estimates, $\tilde{k}_e(\boldsymbol{q},\widetilde{\boldsymbol{\pi}})$ and $\tilde{k}_b(\boldsymbol{q},\widetilde{\boldsymbol{\pi}})$, respectively; and the payload estimate $\tilde{m}_\delta$.

As can be seen in Figure 4.21 and Fig. 4.22, the passivity-based controller offers good performance in the face of material uncertainties and external disturbances. The RMS tracking error in steady-state (*i.e.*, $t \geq 30$s) between the desired end-effector trajectory and true trajectory is ± 0.77 mm; which translates to an arc-length normalized error of $\pm 1.22\%$. Despite the presence of uncertainties, the passivity-based controller also ensures the states converges to the desired trajectory with a smooth transient. Regarding the results of Figure 9, the (nonlinear) stiffness

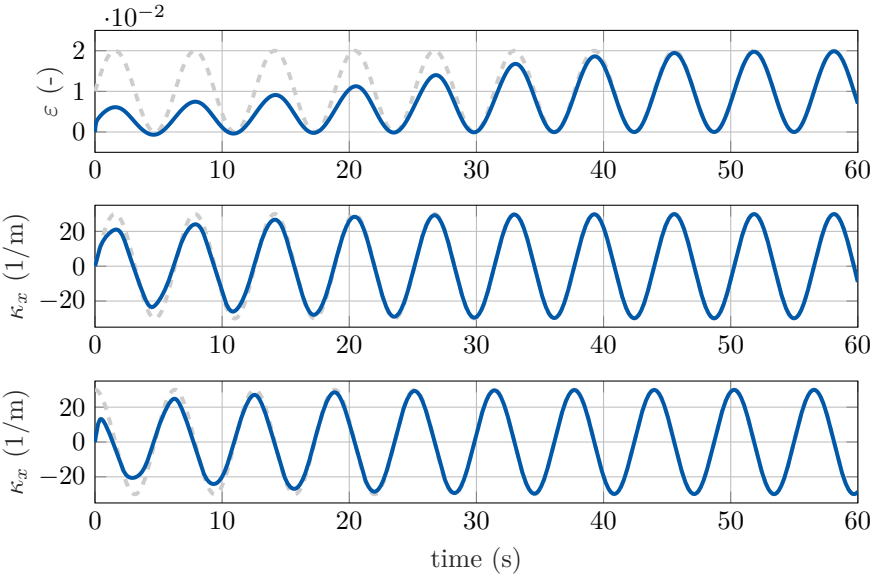


Figure 4.21. State trajectories of soft robot with the passivity-based adaptive controller in (4.54). The figure shows the elongation strain $\varepsilon(t)$ and the curvatures $\kappa_x(t)$, $\kappa_y(t)$ in the xz-plane and yz-plane, respectively; where (—) shows the reference trajectory of the joint configuration $\boldsymbol{q}_d$.

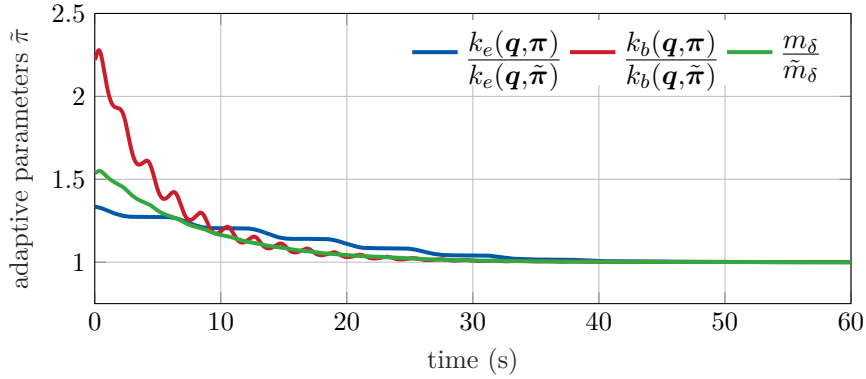


Figure 4.22. The evolution of the elongation stiffness estimate and bending stiffness estimate, $k_e(\boldsymbol{q},\tilde{\boldsymbol{\pi}})$ and $k_b(\boldsymbol{q},\tilde{\boldsymbol{\pi}})$ respectively; and the estimate of the payload $\tilde{m}_\delta$. For illustration, the estimates are normalized with their true value, where we can see they slowly approach one after sufficient time passes. This implies the adaptive algorithms converges to the true hyper-elastic stiffness values independent of parameter uncertainties.

estimates $k_e(\boldsymbol{q},\widetilde{\boldsymbol{\pi}})$ and $k_b(\boldsymbol{q},\widetilde{\boldsymbol{\pi}})$, and the unknown payload mass $\hat{m}_\delta$ slowly converge to the their true values. It should be mentioned that increasing the adaption rate leads to undesired (but bounded) oscillations of the estimates rather than faster convergences, therefore negatively affects the controller's performance.

4.10 Concluding remarks

In this chapter, we aimed to reduce the gap between modeling and control-oriented research in soft robotics. First, the dynamic models that describe the continuum-bodied motions need to be sufficiently accurate, and second the model must retain real-time performance to be applicable in control. By building upon the existing PCC models, we express the continuum deformation using a minimal set of coordinates related to the differential geometry of spatial curves; and explored FEM-based data to model the hyper-elasticity. To retain numerical efficiency, a reduced-order integration scheme is developed that efficiently computes the entries of the Lagrangian model through a Matrix-Differential equation; resulting in a continuum dynamical model for soft manipulators with real-time capabilities at minimal lost in numerical precision.

The dynamic model has been extensively analyzed through simulations and experimental results. Not only does the dynamic models allow for real-time simulations with systems with various degrees of motion, it show good correspondence with the true physical soft robot. Furthermore, a passivity-based adaptive controller is proposed that provides good tracking performance even in the face of parameter uncertainties. The adaptive controller enables online estimation of the hyper-elastic stiffness and external loads, which further enhances the robustness toward modeling uncertainty undoubtedly present in soft robotics. In future work, we wish to further explore FEM-driven data for the parametrization of the spatial shape functions – extending beyond constant-curvature, and employ the proposed controller methods to multi-link soft robots.

5

Dynamic Modeling – Beyond the Constant Strain Approach

Abstract - In this chapter, we address some of the limitation of the previous *Piecewise-Constant-Curvature* (PCC) model presented in Chapter 4, in particular the infinite-dimensionality of the soft robot's deformable body. The continuous dynamics of the soft robot are modeled through the differential geometry of Cosserat beams. Contrary to the PCC model, a wide variety of spatial discretization can be used that better respect the spatial continuity and continuum mechanics of the system. Using a finite-dimensional truncation, the infinite-dimensional system can be written as a reduced-order port-Hamiltonian (pH) model that preserves desirable passivity conditions. Then,a model-based controller is introduced that produces a local minimizer of closed-looppotential energy for the desired end-effector configuration. The stabilizing control utilizes an energy-based approach and exploits the passivity of soft robotic system. The effectiveness of the controller is demonstrated through extensive simulations of various soft manipulators that share a resemblance with biology

This chapter is based on: B.J. Caasenbrood, A.Y. Pogromsky, and H. Nijmijer. *Energy-shaping Controllers for Soft Robot Manipulators through Port-Hamiltonian Cosserat Models.* SN Computer Science, Springer, 2022. doi: 10.1089/soro.2021.0035.

5.1 Introduction

The field of soft robotics is slowly growing as a well-recognized research discipline aimed at developing highly compliant bio-inspired robot. Contrary to rigid robots, soft robots explore '*soft materials*' that significantly enhance the robot's dexterity, inherent safety, enable a rich family of motion primitives, and provide environmental robustness. By fully exploiting soft materials, soft robotics places the first steps towards achieving performance similar to biology [47, 65, 134]. In this work, we primarily focus on a subclass of soft robots called '*soft manipulators*'.

Although significant steps have been taken towards bridging biology and soft robotics, the innate infinite-dimensionality of soft robot models poses substantial challenges on the applicability of model-based control. To be more specific, when considering the soft body's deformation as a state of the system, soft robots theoretical have infinitely many degrees-of-freedom along their continuously deformable body. For this reason, their models are often better suited for a PDE description [62, 124, 242] – for example the mass and momentum balance equations – rather than the conventional ODE that are so common to traditional robotics [154, 201]. Additionally, their actuation often employs distributed loads (*e.g.*, pneumatics [65, 134] and tendons [220, 242]). Consequently, classical descriptions of rigid links and joints paired with local actuation are no longer viable nor physically representative. This paradigm shift calls for novel control-oriented modeling approaches tailored for hyper-flexible and under-actuated robots.

In the last decade, the field of modeling for soft robotic systems has matured sufficiently and currently their applicability in model-based control is slowly feasible [58]. To highlight a few: reduced-order finite element models [62, 242, 252], constant and non-constant curvature approaches [60, 110], Cosserat-beam models [27, 169], and learning-based approaches [31]. Recalling Chapter 4, a popular method of state reduction is the *Piece-wise Constant Strain* (PCS) model that assumes piecewise constant strains along the soft robot's body. Such assumption has proven to be remarkably reliable in solving the dimensionality issue. *Piece-wise Constant Curvature* (PCC) is a subclass of the PCS assumption. Models based on the assumption have shown succesful implementation in feedforward controllers [65], and more recently model-based feedback controllers [60, 110]. Nevertheless, the PCC assumption has (severe) limitations. It does not properly reflect the continuum mechanics and thus are only applicable in restrictive settings. Although computationally performance might surpass continuous models, due to intrinsic kinematic restrictions, it is unable to capture important continuum phenomena, like buckling, environmental interaction, or wave propagation.

On the contrary, Cosserat beam-models have shown to capture a wide range of

continuum deformations. Cosserat models originate from continuum mechanical PDE description and thus allow a more accurate description of the hyper-flexible nature under large deformations. The computational dynamics of Cosserat beams have been extensively developed by [188] through Geometrically-Exact finite elements on the Lie group SE(3); and recently, these models are slowly gaining popularity in the soft robotics community [27, 77, 169, 170, 220]. Ultimately, the strong nonlinearities paired with the diligence to achieve biological performance encourages the use of Cosserat models for control. Yet, compared to the abundance of PCC soft robotic models, literature on model-based control is scarce.

In this chapter, we aim to highlight the capabilities of Cosserat models for model-based control, in particular energy-based strategies. To this end, a finite-dimensional modeling approach is proposed such that the continuous dynamics can be cast into a port-Hamiltonian (pH) structure. The Lagrangian modeling framework is adopted from [27] and [169], but modified to suit a pH-structure. The main advantage of pH systems is the common formalism with energy-based control. Through the pH structure, we propose an energy-shaping control law that ensures stabilization of the end-effector of the soft robot. Similar energy-based control strategies can be found in [70, 159, 160, 182] for rigid-body systems. As a study case, we consider a soft robot manipulator inspired by an octopus tentacle. With the ability to deform continuously and its distributed muscular system, it is ideal for illustrating the complex morphological motions present in soft robotics. Again, all code is made publicly available under the *Sorotoki* toolkit on [33], which builds upon previous work Caasenbrood et al. [36].

The chapter is organized as follows. Section 5.2 will detail a modeling approach for a general class of soft robot manipulators, starting with the Cosserat-beam theory. In Section 5.3, we propose an energy-shaping control strategy. Lastly, we show the effectiveness of energy-based controller through numerical simulation in Section 5.4, followed by a brief conclusion in Section **??**.

5.2 Generalized models for soft manipulators

In this chapter, we will explore Lie group theory applied to deformable robots. We have introduced some of notations earlier in Chapter 4, but for convenience, we briefly recapulate.The Lie group of rigid-body transformation on $\mathbb{R}^3$ is denoted by SE(3). The group of homogeneous rotation is denoted by SO(3). The tangent space at the identity of the group is called the *Lie algebra*, and it can be used to describe the evolution of the Lie group. The Lie algebra of SE(3) and SO(3) are denoted by se(3) and so(3), respectively. Lastly, the cross operator (*i.e.*, "$\times$") and hat operator (*i.e.*, "$\wedge$") are used to transform a column vector of $\mathbb{R}^3$ or $\mathbb{R}^6$

into an element of the Lie algebra so(3) or se(3), respectively. A comprehensive introduction is given in Appendix B.1 based on the work of Murray et al. [154].

5.2.1 Preliminary on geometric Cosserat theory

In Cosserat theory, slender deformable solids are modeled as elastic strings subjected to geometric finite-strain theory. Drawing the analogy to soft robotics, we model the soft robot as a one-dimensional spatial curve passing through the geometric center of the soft robot (see Figure 1). Given its spatial-temporal nature, we introduce a temporal variable $t \in [0,\, T]$ with finite horizon time T, and a spatial variable $\sigma \in [0,\, L]$ with L the undeformed length of the soft robot. For each point on the backbone, we introduce a (mobile) coordinate frame. The homogeneous rotation related to these coordinate frames is given by the rotation matrix $\boldsymbol{\Phi} : [0, L] \times [0, T] \to \mathrm{SO}(3)$, and their origin by the position vector $\boldsymbol{\gamma} : [0, L] \times [0, T] \to \mathbb{R}^3$. For convenience and readability, we will denote the temporal and spatial domains as $\mathbb{T} = [0,\, T]$ and $\mathbb{X} = [0,\, L]$, respectively.

Following the geometric approach [27, 28, 169, 170, 188], we may equivalently represent each coordinate frames that are rigidly attached to the continuous backbone of the soft robot by a parameterized space curve in SE(3):

$$\boldsymbol{g}(\sigma, t) = \begin{pmatrix} \boldsymbol{\Phi}(\sigma, t) & \boldsymbol{\gamma}(\sigma, t) \\ \mathbf{0}_3^\top & 1 \end{pmatrix} \in \mathrm{SE}(3). \tag{5.1}$$

An important notion in Lie group theory applied to robotics is the so-called "*exponential coordinate representation*". Conventionally, for rigid multi-body mechanics, any rigid-body transformation from one coordinate frame into another can characterized by a vector living in $\mathbb{R}^6$ – called the *twist* – whose six entries are referred to as the exponential coordinates. Three of these exponential coordinates are related to the translation and the other three to orientation. As such, the same principle applies to rigid-body rotations described by matrices in SO(3) that can be parameterized by three exponential coordinates. For example, we may express a general rotation in terms of its exponential coordinates $\boldsymbol{R}(\boldsymbol{\Omega}, t) = \exp\left(\boldsymbol{\Omega}^\times t\right)$ where $\boldsymbol{\Omega}^\times \in \mathrm{so}(3)$ is a skew-symmetric matrix with its three unique entries $\boldsymbol{\Omega} = (\omega_1, \omega_2, \omega_3)^\top$ being the unit-time angular velocities. Recall that the superscript denotes the skew-symmetric matrix operator $(\cdot)^\times : \mathbb{R}^3 \to \mathrm{so}(3)$.

Since soft robot models are expressed in spatial and temporal coordinates, we aim to seek two exponential coordinate descriptions – a spatial twist $\boldsymbol{\xi}$ called the geometric strain , and a temporal twist $\boldsymbol{\eta}$ called the geometric velocity. The expressions for the strain field $\boldsymbol{\xi}$ and velocity field $\boldsymbol{\eta}$ anywhere on the Cosserat beam can be found by exploring the differential geometry of the curve $\boldsymbol{g}$. To do so, we must introduce some smoothness conditions.

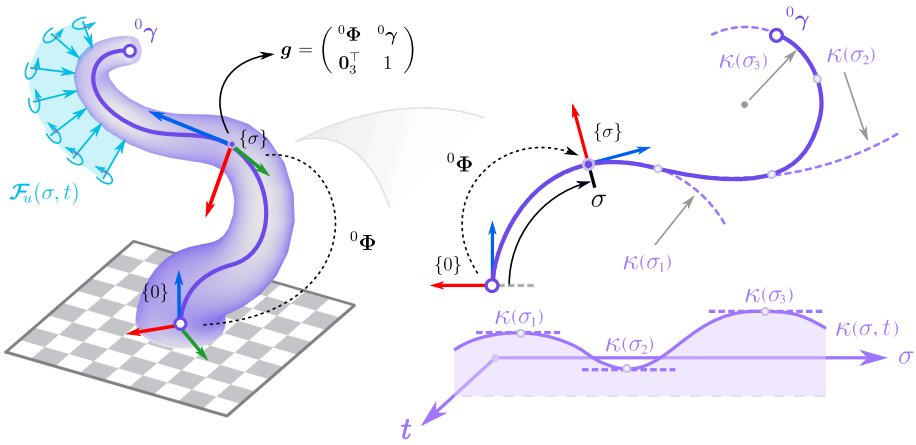


Figure 5.1. Schematic representation of the continuously variable Cosserat beam model for a general class of soft manipulators, given by a backbone curve $\boldsymbol{\gamma}$ and orientation matrix $\boldsymbol{\Phi}$ relative to a base frame at $\{0\}$ Together they form a parameterized curve $\boldsymbol{g}(\sigma, t) = (\boldsymbol{\Phi}(\sigma, t), \boldsymbol{\gamma}(\sigma, t)) \in \mathrm{SE}(3)$. The representation of the soft robot is inspired by the octopus' tentacle whose the muscle forces are modelled as a distributed input $\boldsymbol{\mathcal{F}}_u$.

Assumption 5.1 (Distributed actuation and body loads) Contrary to Chapter 4 where the (pneumatic) inputs are lumped, we propose a more generalizable control input to our system, which is described by a distributed input force $\boldsymbol{n}_u : \mathbb{X} \times \mathbb{T} \to \mathbb{R}^6$ and distributed input moment $\boldsymbol{m}_u : \mathbb{X} \times \mathbb{T} \to \mathbb{R}^6$ acting on the deformable backbone $\boldsymbol{g}$. For convenience, we collect these into a vector $\boldsymbol{\mathcal{F}}_u = \left(\boldsymbol{m}_u^\top, \boldsymbol{n}_u^\top\right)^\top$ called the *control wrench*. We wish to emphasize that modeling a distributed forces and moments encompasses a broad collection of soft actuation models, including: pneumatics, hydraulics, tendons, and ferro-magnetic and thermo-mechanical actuation. Similarly, we also introduce a distributed body-load force (*e.g.*, due to gravity) given by $\boldsymbol{\mathcal{F}}_\mathrm{b} = (\mathbf{0}_3, \boldsymbol{n}_\mathrm{b}^\top)^\top$ where $\boldsymbol{n}_\mathrm{u} = -\rho_0 A_0 \boldsymbol{\Phi}^\top \boldsymbol{a}_\mathrm{b}$ with $\boldsymbol{a}_\mathrm{b}$ a body acceleration, ρ_0 the density, and A_0 the cross-sectional area. Both $\boldsymbol{\mathcal{F}}_\mathrm{u}$ and $\boldsymbol{\mathcal{F}}_\mathrm{g}$ are wrenches expresses in the body frame – their direction depends on the configuration $\boldsymbol{g}$.

Assumption 5.2 (On differentiability) Any external distributed force acting on the continuum time-variant backbone curve (5.1) is considered to be sufficiently smooth for any instance $t \in \mathbb{T}$ and $\sigma \in \mathbb{X}$ such that parametrized backbone $\boldsymbol{g}(\sigma, t) \in \mathrm{SE}(3)$ is differentiable everywhere on $\mathbb{X}$ and $\mathbb{T}$.

5.2.2 Local strain and velocity

Following the Cosserat beam modeling approach from Simo et al. [188], Boyer et al. [27], Renda et al. [169, 170], let $\boldsymbol{\Gamma} = (\kappa_1,\, \kappa_2, \kappa_3)^\top$ and $\boldsymbol{U} = (\nu_1,\, \nu_2,\, \nu_3)^\top$ be the torsion-curvature and elongation-shear strain vector, respectively. Then, an expression for strain field $\boldsymbol{\xi}(\sigma, t)$ is obtained through spatial differentiation of $\boldsymbol{g}$:

$$\hat{\boldsymbol{\xi}} := \boldsymbol{g}^{-1}\frac{\partial \boldsymbol{g}}{\partial \sigma} = \begin{pmatrix} \boldsymbol{\Gamma}^\times & \boldsymbol{U} \\ \mathbf{0}_3^\top & 0 \end{pmatrix} \quad \xRightarrow{\text{isomorphism}} \quad \boldsymbol{\xi} := \begin{pmatrix} \boldsymbol{\Gamma} \\ \boldsymbol{U} \end{pmatrix}. \tag{5.2}$$

Note the difference between $\hat{\boldsymbol{\xi}}$ and $\boldsymbol{\xi}$ in (5.2), being a matrix element of se(3) and column vector of $\mathbb{R}^6$, respectively. The spaces se(3) and $\mathbb{R}^6$ are isomorphic (*i.e.*, structure preserving) and the mapping between these spaces is called the *"isomorphism"* denoted by $\hat{(\cdot)} : \mathbb{R}^6 \to \text{se}(3)$ and its inverse $\check{(\cdot)} : \text{se}(3) \to \mathbb{R}^6$.

Following, the temporal exponential coordinates are given by $\boldsymbol{\Omega} = (\omega_1, \omega_2, \omega_3)^\top$ and $\boldsymbol{V} = (v_1,\, v_2,\, v_3)^\top$ that denote the angular and linear velocity vector, respectively. Then, an expression for velocity field $\boldsymbol{\eta}(\sigma, t)$ is obtained through time differentiation of $\boldsymbol{g}$:

$$\hat{\boldsymbol{\eta}} := \boldsymbol{g}^{-1}\frac{\partial \boldsymbol{g}}{\partial t} = \begin{pmatrix} \boldsymbol{\Omega}^\times & \boldsymbol{V} \\ \mathbf{0}_3^\top & 0 \end{pmatrix} \quad \xRightarrow{\text{isomorphism}} \quad \boldsymbol{\eta} := \begin{pmatrix} \boldsymbol{\Omega} \\ \boldsymbol{V} \end{pmatrix}. \tag{5.3}$$

Let it be clear that $\boldsymbol{\xi}(\sigma, t)$ and $\boldsymbol{\eta}(\sigma, t)$ are yet unknown vector fields, they simply follow from the differential geometry of SE(3). As such, they live in its tangent space se(3) – called its Lie algebra. Now given the geometric structures in (5.2) and (5.3), we can start detailing the forward kinematics of the soft continuum robot by exploring the spatial and temporal continuities of the manifold $\boldsymbol{g}(\sigma, t)$. Their connection is described in detail in [27, 28, 170, 171, 188], but briefly recapitulated here to be self-contained. Recalling Assumption 5.2, which assumes the configuration space $\boldsymbol{g}$ to be everywhere differentiable, we can introduce the equality of mixed partials, *i.e.*, $\frac{\partial}{\partial t}(\frac{\partial \boldsymbol{g}}{\partial \sigma}) = \frac{\partial}{\partial \sigma}(\frac{\partial \boldsymbol{g}}{\partial t})$. Then, by substitution of $\frac{\partial \boldsymbol{g}}{\partial t} = \boldsymbol{g}\hat{\boldsymbol{\eta}}$ and $\frac{\partial \boldsymbol{g}}{\partial \sigma} = \boldsymbol{g}\hat{\boldsymbol{\xi}}$, we obtain the so-called *compatibility equation*:

$$\boldsymbol{g}\hat{\boldsymbol{\eta}}\hat{\boldsymbol{\xi}} + \boldsymbol{g}\frac{\partial \hat{\boldsymbol{\xi}}}{\partial t} = \boldsymbol{g}\hat{\boldsymbol{\xi}}\hat{\boldsymbol{\eta}} + \boldsymbol{g}\frac{\partial \hat{\boldsymbol{\eta}}}{\partial \sigma}, \tag{5.4}$$

The equality above ensures that the partial derivatives with respect to time and space are continuously connected on the manifold $\boldsymbol{g}$. Simply put, deformations that are tied to the spatial derivatives need to have continuous velocity profiles (*i.e.*, the temporal derivative) – and vice versa. Pre-multiplication of the expression above with $\boldsymbol{g}^{-1} \in \text{SE}(3)$ and rearranging the equality, we obtain

$$\frac{\partial \hat{\boldsymbol{\eta}}}{\partial \sigma} = -\left(\hat{\boldsymbol{\xi}}\hat{\boldsymbol{\eta}} - \hat{\boldsymbol{\eta}}\hat{\boldsymbol{\xi}}\right) + \dot{\hat{\boldsymbol{\xi}}}. \tag{5.5}$$

Focusing on the RHS term $(\hat{\boldsymbol{\xi}}\hat{\boldsymbol{\eta}} - \hat{\boldsymbol{\eta}}\hat{\boldsymbol{\xi}})$, we can recognize the Lie bracket or the commuter between the geometric vector fields $\boldsymbol{\xi}$ and $\boldsymbol{\eta}$ (see [154]). Since the Lie bracket $[\hat{\boldsymbol{\xi}}, \hat{\boldsymbol{\eta}}]$ itself also belongs to the Lie algebra se(3), which is isomorphic to $\mathbb{R}^6$ via the operator $\hat{\boldsymbol{\eta}} \to \boldsymbol{\eta}$, we can rewrite the expressions as follows

$$\frac{\partial \boldsymbol{\eta}}{\partial \sigma} = -\mathbf{ad}_{\boldsymbol{\xi}}\boldsymbol{\eta} + \dot{\boldsymbol{\xi}}, \tag{5.6}$$

where $\mathbf{ad}_{(\cdot)} : \mathbb{R}^6 \to \mathbb{R}^{6\times 6}$ defines the adjoint action on vectors belonging to the Lie algebra se(3). Drawing an analogy to rigid robotics, the expression in (5.6) may be seen as the forward velocity kinematics for a serial chain robot manipulator with infinitely many links. To that end, we can collect the PDEs (5.2), (5.6) and the time derivative of (5.6) (*i.e.*, the acceleration) into one global *Partial Matrix Differential Equation* (PMDE) that describes the full continuum-body kinematics of the soft deformable robotic body. The PMDE curve kinematics takes the form

$$\frac{\partial}{\partial \sigma}\left(\begin{bmatrix}\boldsymbol{g}\\ \mathbf{0}\end{bmatrix}, \boldsymbol{\eta}, \dot{\boldsymbol{\eta}}\right) = \left(\begin{bmatrix}\boldsymbol{g}\hat{\boldsymbol{\xi}}\\ \mathbf{0}\end{bmatrix}, -\mathbf{ad}_{\boldsymbol{\xi}}\boldsymbol{\eta} + \dot{\boldsymbol{\xi}}, -\mathbf{ad}_{\boldsymbol{\xi}}\dot{\boldsymbol{\eta}} - \mathbf{ad}_{\dot{\boldsymbol{\xi}}}\boldsymbol{\eta} + \ddot{\boldsymbol{\xi}}\right) \tag{5.7}$$

which is a square differential matrix. For a general case, the boundary conditions of PDE in (5.7) should satisfy $\boldsymbol{g}(0,t) = \boldsymbol{g}_0$, $\boldsymbol{\eta}(0,t) = \boldsymbol{\eta}_0$ and $\dot{\boldsymbol{\eta}}(0,t) = \dot{\boldsymbol{\eta}}_0$. However, in case of a manipulator whose base is spatially fixed, the boundary conditions should satisfy $\boldsymbol{g}(0,t) = \boldsymbol{g}_0$, and $\boldsymbol{\eta}(0,t) = \dot{\boldsymbol{\eta}}(0,t) = \mathbf{0}_6$. An importance remark is that if the strain fields $\boldsymbol{\xi}$, $\dot{\boldsymbol{\xi}}$, and $\ddot{\boldsymbol{\xi}}$ are analytic such that we choose an instance $t \in [0, T]$ and keeping σ variable, the PMDE in (5.7) becomes an ordinary Matrix Differential Equation (MDE) which can be solved numerically using the techniques discussed in Chapter 4.

Example 5.1 (Kinematic behavior of variable strain soft segment)**.** Let $\mathbb{X} = [0, L]$ with $L = 100$ mm and $\mathbb{T} = [0, T]$ with $T = 1$ s. Consider in this example a planar bending problem where the geometric strain vector takes the form $\boldsymbol{\xi} = (0, \kappa(\sigma,t), 0, 1, 0, 0)^\top$ with κ is an a priori known non-constant curvature profile. Inspired by the analytic solutions of the Euler-Bernoulli cantilever model [95], we propose an analytic expression for the planar curvature strain:

$$\kappa(\sigma,t) = \kappa_{\max}\, \theta(\sigma) h(t), \tag{5.8}$$

$$\theta(\sigma) = \sinh(\beta\sigma) + \sin(\beta\sigma) + \frac{\cos(\beta L) + \cosh(\beta L)}{\sin(\beta L) + \sinh(\beta L)}\big(\cos(\beta\sigma) - \mu\cosh(\beta\sigma)\big), \tag{5.9}$$

where $\kappa_{\max} = 60$ mm^{-1} is the curvature amplitude, $\theta(\sigma)$ the first strain eigenmode of the cantilever beam model, $h(t) = \frac{(1-k)t}{(1+k)-2k|t|}$ a transient modeled after a sigmoid function with $k = \frac{1}{2}$, and a parameter β which is the smallest positive real

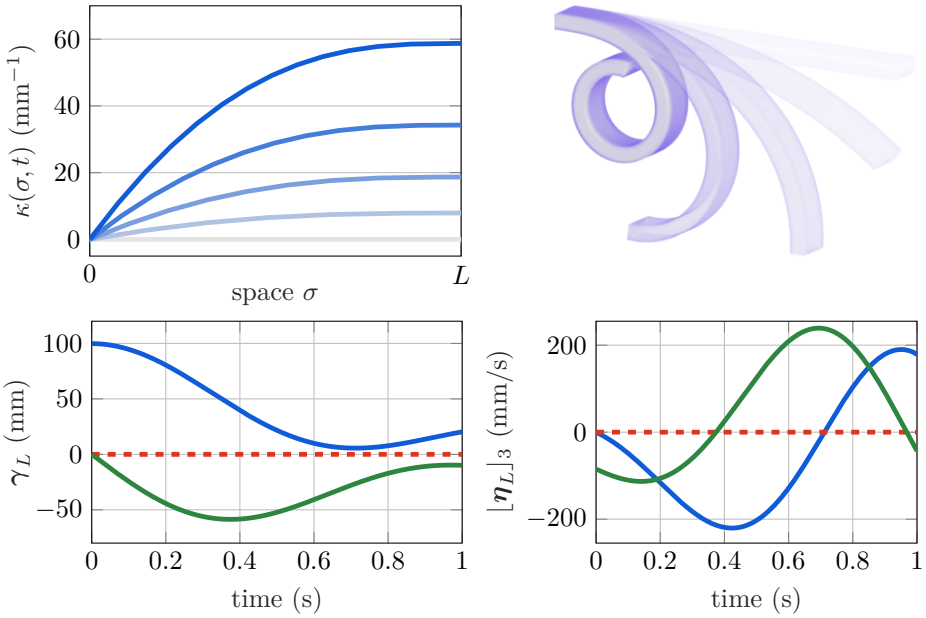


Figure 5.2. Spatio-temporal evolution of the cantilever beam obtained by solving the exact continuum forward kinematics described in (5.7). The figure shows the forward kinematic solutions of the end-effector position $\boldsymbol{\gamma}_l$ and its linear velocities $\lfloor \boldsymbol{\eta} \rfloor_3$, where the following color coding is used $\{x, y, z\} = \{$ –, –, –$\}$.

that satisfies the nonlinear equality $\cos(\beta L)\cosh(\beta L) = 1$. The physical interpretation β is the inverse wavelength thus it is inversely proportional to the beam length L. A numerical approximation of this value is $\beta \approx 0.597\,\pi L^{-1}$. The strain velocity $\dot{\boldsymbol{\xi}}$ and strain acceleration $\ddot{\boldsymbol{\xi}}$ are obtained straightforwardly by differentiation of $\kappa(\sigma, t)$ with respect to time. Hence, we have explicit expressions on the strain, its velocity and acceleration. Therefore, we can evaluate $\boldsymbol{s}_i(\sigma) := \boldsymbol{\xi}(\sigma, t_i)$, $\boldsymbol{v}_i(\sigma) := \dot{\boldsymbol{\xi}}(\sigma, t_i)$, and $\boldsymbol{a}_i(\sigma) := \ddot{\boldsymbol{\xi}}(\sigma, t_i)$ for any time sample $t_i \in \mathbb{T}$.

Then, substitution of $\boldsymbol{s}_i$, $\boldsymbol{v}_i$ and $\boldsymbol{a}_i$ into (5.7) leads to an ordinary differential equation that depends exclusively on σ. The forward kinematics can then be solved for the spatial horizon $\mathbb{X} = [0, L]$ using an explicit integration routine (fourth order Runge Kutta method), which produces $\boldsymbol{g}$, $\boldsymbol{\eta}$ and $\dot{\boldsymbol{\eta}}$. We repeat this process until the full forward kinematics is recovered for all $t \in \mathbb{T}$. The results for the Euler-Bernoulli cantilever beam are shown in Figure 5.2. Figure 5.2 shows for $\kappa(\sigma, t)$ evolves over space and time, and how they relate to the continuum deformations. It also shows the position $\boldsymbol{\gamma}_L$ and the linear velocities $\lfloor \boldsymbol{\eta} \rfloor_3$ of the end-effector.

Note that this approach can easily deal with non-constant curvature situations, in contrast to the approach discussed in Chapter 4. Namely, Chapter 4 subdivided the domain $\mathbb{X}$ into small PCC segments leading to notorious discontinuities at the interfaces (see Figure 5.3).

5.2.3 Finite-dimensional projection

The problem is, however, that the exponential coordinates of the geometric strain vector belong (in theory) to an infinite set of real-valued functions. Similar to many continuum mechanics problems, the state dimension for such multi-variable coordinate representation are of infinite dimensional nature. To solve this, we must project our solution space onto a finite dimensional subset that is countable and therefore computable in this setting. In the finite element method, the solution presents itself by dividing the continuum into a finite set of elements. Here we follow an approach similar to Chirikjian et al. [42, 43] that proposed to express the spatial deformations of slender (semi-rigid) continuum robots by finite set of basis functions called *spatial modes*. This idea also finds application is disturbance rejection for structural vibration of semi-rigid flexible manipulators whose modeling strategies are identical the Euler-Bernoulli assumptions. Here though we introduce a modal approximation to find a suitable finite-dimensional approximation of the strain twist $\boldsymbol{\xi}(\sigma, t)$ for all points along the material domain $\mathbb{X}$. To do so, we assume the following:

Assumption 5.3 Assuming the strain field has a separable spatio-temporal nature, any entry of the strain vector field $\boldsymbol{\xi} = (\xi_1, \xi_2, ..., \xi_6)^\top$ can be written as an infinite expansion of the following form:

$$\xi_i(\sigma, t) = \sum_{n=1}^{\infty} \theta_n(\sigma) q_{i,n}(t) + \xi_i^\circ(\sigma) \quad i \in \{1, \dots, 6\}, \tag{5.10}$$

where $\{\theta_n\}_{n=1}^{\infty}$ is a set of (orthogonal) basis functions $\theta_n : \mathbb{X} \to \mathbb{R}$ together with modal coefficients $q_{i,n} : \mathbb{T} \to \mathbb{R}$, and an intrinsic time-invariant strain $\xi_i^\circ : \mathbb{X} \to \mathbb{R}$. The basis functions $\theta_n(\cdot)$ and the modal coefficients $q_n(\cdot)$ are both smooth functions.

Assumption 5.4 Given infinite expansion (5.10), the k-th order truncation for any entry of the strain field, defined as

$$[\xi_i]_k(\sigma, t) := \sum_{n=1}^{k} \theta_n(\sigma) q_{i,n}(t) + \xi_i^\circ(\sigma) \quad i \in \{1, \dots, 6\}, \tag{5.11}$$

converges uniformly on $\mathbb{X}$ and $\mathbb{T}$ as the index $k \to \infty$. Moreover, we assume that uniform convergence holds for its partial derivatives $\frac{\partial}{\partial t}[\boldsymbol{\xi}]_k$ and $\frac{\partial}{\partial \sigma}[\boldsymbol{\xi}]_k$.

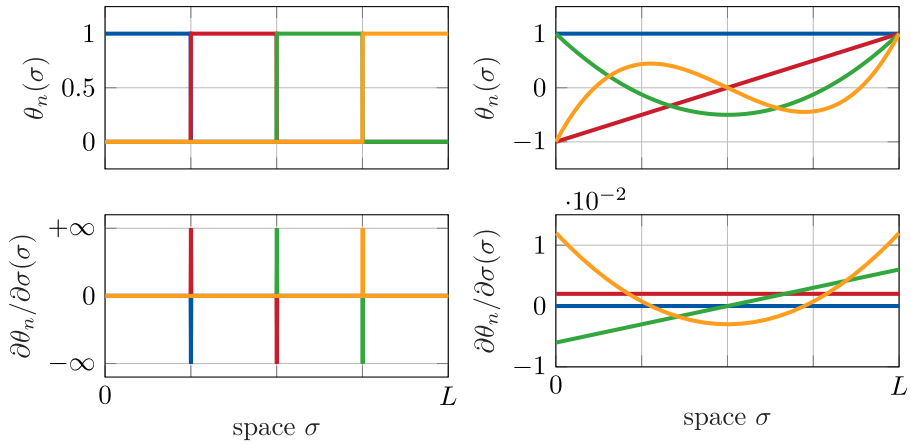


Figure 5.3. Example plot of the Constant-Strain parametrization in Chapter 3 (left), and the new strain parametrization (5.13) using Chebyshev polynomials (right). The ordering of the strain basis is as follows $\{\theta_1, ..., \theta_4\} = \{$ —, —, —, —$\}$. Notice that the spatialdiscontinuities in the PCC model induces spikes that directly violate the compatibility equation in (5.4).

Accordingly, we can rewrite the k-th order truncation of the complete strain field as a linear matrix operation as follows

$$\begin{aligned}[\boldsymbol{\xi}]_k &= \left(\boldsymbol{I}_6 \otimes \begin{bmatrix}\theta_1 & \dots & \theta_k\end{bmatrix}\right)\boldsymbol{q} + \boldsymbol{\xi}^\circ, \\ &= \underbrace{\begin{pmatrix}\theta_1 & \dots & \theta_k & \dots & 0 & \cdots & 0 \\ \vdots & \ddots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & \dots & \theta_1 & \dots & \theta_k\end{pmatrix}}_{\boldsymbol{\Theta}(\sigma)} \underbrace{\begin{pmatrix}q_{1,1} \\ \vdots \\ q_{6,k}\end{pmatrix}}_{\boldsymbol{q}(t)} + \boldsymbol{\xi}^\circ, \end{aligned} \tag{5.12}$$

where $\boldsymbol{\Theta} \in \mathbb{R}^{6\times 6k}$ is a sparse matrix-valued function whose columns are mutually orthonormal, the operator $\otimes$ denotes the Kronecker product, and the vector $\boldsymbol{q} \in \mathbb{R}^{6k}$ the collection of all time-varying modal coefficients related to the columns of $\boldsymbol{\Theta}$. Although a wide variety of bases are possible (see for instance [27, 60]), we have chosen a modified Legendre polynomial set:

$$\theta_n(\sigma) = \frac{2}{2^n(n-1)!}\frac{d^{n-1}}{d\sigma^{n-1}}\left[\left(\frac{2\sigma}{L} - 1\right)^2 - 1\right]^{n-1} \tag{5.13}$$

with $n \in \mathbb{Z}_{>0}$ the polynomial degree A comparison between the proposed basis and previous method in Chapter 4 is shown in Figure 5.3. Important to note

is that the previous PCC condition discussed introduces discontinuities in space that violate the compatibility equation in (5.4). Also note now that the inner product between elements of the set of modified Legendre functions $\{\theta_n\}_{n=1}^{k}$ satisfies $\langle \theta_i, \theta_j \rangle_{\mathbb{X}} := \int_{\mathbb{X}} \theta_i \theta_j d\sigma = 0$ for $i \neq j$, and 1 otherwise. An alternative option could be constructing the set of basis functions through the so-called '*snapshot decomposition method*' [12, 62, 124].

5.2.4 Reduced-order kinematics

Given the finite-dimensional truncation in (5.12), we can now find an expression for the finite-dimensional forward kinematics in terms of the generalized coordinates $\boldsymbol{q}$ and its velocities components $\dot{\boldsymbol{q}}$.

First, let us regard the configuration of the soft robot $\boldsymbol{g} \in \mathrm{SE}(3)$. Recall that the spatial evolution of the curve is described by $\partial \boldsymbol{g}/\partial\sigma = \boldsymbol{g}\boldsymbol{\xi}^{\wedge}$, see Eq. (5.2). Given the initial condition $\boldsymbol{g}(0,\cdot) = \boldsymbol{g}_0$, an approximation of the continuously deformable soft robot can be obtained by partial integration over the spatial domain:

$$[\boldsymbol{g}]_k(\sigma, \boldsymbol{q}) = \boldsymbol{g}_0 \exp_{\mathrm{SE}(3)} \left[\int_0^{\sigma} [\hat{\boldsymbol{\xi}}]_k(s, \boldsymbol{q}) \, ds \right]. \tag{5.14}$$

The computation of the mapping $\exp_{\mathrm{SE}(3)}$ is given in Appendix B.1. Note that the expression above nothing more than the reconstruction of the curve by integration of its tangent space along its spatial parameter σ. Next, lets regard the velocity kinematics $\boldsymbol{\eta}(\sigma, t)$ for any point σ on the backbone curve. Using the differential property of the adjoint action $\mathbf{ad}_{\boldsymbol{\xi}} = -\partial/\partial\sigma[\mathbf{Ad}_{g^{-1}}]\mathbf{Ad}_g$ [154], we can rewrite the continuumforward kinematics in (5.6) as

$$\frac{\partial \boldsymbol{\eta}}{\partial \sigma} = \frac{\partial}{\partial \sigma} \left(\mathbf{Ad}_{\boldsymbol{g}^{-1}} \right) \mathbf{Ad}_{\boldsymbol{g}} \boldsymbol{\eta} + \dot{\boldsymbol{\xi}}. \tag{5.15}$$

Now, given the initial condition $\boldsymbol{\eta}(0,t) = \mathbf{0}_6$ and the approximations $[\boldsymbol{\xi}]_k$ and $[\boldsymbol{g}]_k$, we can find an approximation to the velocity twist $\boldsymbol{\eta}$ by partial integration over space:

$$[\boldsymbol{\eta}]_k(\sigma, \boldsymbol{q}, \dot{\boldsymbol{q}}) = \mathbf{Ad}^{-1}_{[\boldsymbol{g}]_k(\sigma,\boldsymbol{q})} \int_0^{\sigma} \mathbf{Ad}_{[\boldsymbol{g}]_k(s,\boldsymbol{q})} \boldsymbol{\Theta}(s) \dot{\boldsymbol{q}} \, ds \,. \tag{5.16}$$

Observe that in (5.16) we can bring $\dot{\boldsymbol{q}}$ outside the integrand, hence the velocity twist has a linear dependency on the joint velocities – a property common to rigid robotics [154, 201]. This naturally gives rise to a linear matrix transformation called the *geometric manipulator Jacobian*. The geometric Jacobian matrix $[\boldsymbol{J}]_k \in \mathbb{R}^{6\times 6k}$ is defined by

$$[\boldsymbol{J}]_k := \mathbf{Ad}^{-1}_{[\boldsymbol{g}]_k(\sigma,\boldsymbol{q})} \int_0^{\sigma} \mathbf{Ad}_{[\boldsymbol{g}]_k(s,\boldsymbol{q})} \boldsymbol{\Theta}(s) \, ds, \tag{5.17}$$

such that $[\boldsymbol{\eta}]_k = [\boldsymbol{J}]_k \dot{\boldsymbol{q}}$. The geometric Jacobian plays an important role in obtaining the Lagrangian form of the reduced-order dynamic model. Finally, to express the acceleration twist, we take the time-derivative of (5.16) leading to

$$\begin{aligned}[\dot{\boldsymbol{\eta}}]_k &= [\boldsymbol{J}]_k \ddot{\boldsymbol{q}} + [\dot{\boldsymbol{J}}]_k \dot{\boldsymbol{q}}, \\ &= [\boldsymbol{J}]_k \ddot{\boldsymbol{q}} + \mathbf{Ad}^{-1}_{[\boldsymbol{g}]_k} \int_0^\sigma \mathbf{Ad}_{[\boldsymbol{g}]_k(s,\boldsymbol{q})} \mathbf{ad}_{[\boldsymbol{\eta}]_k(s,\boldsymbol{q},\dot{\boldsymbol{q}})} \boldsymbol{\Theta}(s)\, ds\, \dot{\boldsymbol{q}}, \end{aligned} \tag{5.18}$$

which gives rise to the analytic expression of the time-derivative of the geometric Jacobian $[\dot{\boldsymbol{J}}]_k$. Although the expression for the $\dot{\boldsymbol{J}}$ is compact, its derivation is fairly involved. We therefore refer the reader to Appendix B.2 on its derivation.

5.2.5 Reduced-order dynamics using Newton-Euler

Here, we detail the dynamics of the Cosserat beam. A majority of the dynamic framework presented here is adopted from Boyer et al. [27]; yet we introduce some modification to allow a pH-structure. Other works that adopt a similar approach include [76, 169, 170, 220]. First, let us consider an infinitesimal slice of continuum body that is perpendicular to the backbone curve. The kinetic momenta of this infinitesimal slice is then given by $\boldsymbol{\mu}(\sigma, t) := \boldsymbol{\mathcal{M}}\boldsymbol{\eta}(\sigma, t)$ in which $\boldsymbol{\mathcal{M}} \in \mathrm{se}^*(3) \times \mathrm{se}(3) \cong \mathbb{R}^{6\times 6}$ denotes the symmetric inertia tensor.

Remark 5.1 For some soft robots, the inertia tensor $\boldsymbol{\mathcal{M}}$ may have an explicit dependency on space or time (or both). A common phenomena in soft robotics that innately lead to such dependencies include for instance the *ballooning effect* of soft pneumatic networks. During actuation, matter expands outwards radially, changing the cross-sectional geometry and thus its inertial properties. That said, the inertial changes due to ballooning is negligibly small. Besides, ballooning is often an undesired phenomena that is reduced by appropriate structural design such as internal ribs, or bellow-like geometries. Nevertheless, examples where inertial changes cannot be neglected include the class of growing or morphing soft robots [87, 105, 106], self-healing soft robots [173, 213], origami soft robot manipulators [104, 126, 245]. Pressure-driven soft robots actuated by near incompressible fluids like water also suffer from these inertial changing effects. But, these robots are scarce and purposefully design for aquatic environments which is outside the scope of this work. For many soft robots, air or gas is preferred over liquids. For sake of simplicity, we limit ourselves to a diagonal invariant inertia tensor:

$$\boldsymbol{\mathcal{M}}(\sigma, t) \simeq \begin{pmatrix} \rho_0 \boldsymbol{\mathcal{J}} & \mathbf{0}_{3\times 3} \\ \mathbf{0}_{3\times 3} & \rho_0\, A_0 \boldsymbol{I}_3 \end{pmatrix} \succ 0,$$

in which the (cross-sectional average) density is $\rho_0 > 0$, the cross-sectional area of the soft robot $A_0 > 0$, and the second moment of area $\boldsymbol{\mathcal{J}} \in \mathrm{so}^*(3) \times \mathrm{so}(3) \cong \mathbb{R}^{3\times 3}$.

We emphasize here that considering an invariant cross-sectional inertia tensor is not an overly conservative assumption. It still holds true for a large class of soft manipulators, and it has limited effects on the generality of our modeling framework.

Using the expression of the kinetic momenta $\boldsymbol{\mu}(\sigma, t)$ of the infinitesimal rigid body in free-motion, we can write the equation of motion for a particular slice at σ using the Newton-Euler equations:

$$\frac{\partial}{\partial t}(\mathbf{Ad}_{\boldsymbol{g}}^{-\top}\boldsymbol{\mu}) = \mathbf{Ad}_{g}^{-\top}\boldsymbol{\mathcal{F}}, \tag{5.19}$$

where again $\mathbf{Ad}_{(\cdot)}$ stands for the adjoint action, and $\boldsymbol{\mathcal{F}} = \boldsymbol{\mathcal{F}}_c + \boldsymbol{\mathcal{F}}_u - \boldsymbol{\mathcal{F}}_{\mathrm{b}} - \boldsymbol{\mathcal{F}}_{\mathrm{e}}$ the resultant wrench that is composed of the constraint wrench $\boldsymbol{\mathcal{F}}_c$, the input wrench $\boldsymbol{\mathcal{F}}_u$, and the potential wrenches due to body accelerations and hyper- and visco-elasticity, $\boldsymbol{\mathcal{F}}_{\mathrm{b}}$ and $\boldsymbol{\mathcal{F}}_{\mathrm{e}}$, respectively. Further evaluation of (5.19) leads to

$$\boldsymbol{\mathcal{M}}\dot{\boldsymbol{\eta}} - \mathbf{ad}_{\boldsymbol{\eta}}^{\top}\boldsymbol{\mathcal{M}}\boldsymbol{\eta} = \boldsymbol{\mathcal{F}}, \tag{5.20}$$

where we used the fact that $\frac{\partial}{\partial t}\mathbf{Ad}_{\boldsymbol{g}}^{-1} = -\mathbf{ad}_{\boldsymbol{\eta}}\mathbf{Ad}_{\boldsymbol{g}^{-1}}$. Before continuing, we introduce a slight modification to the relation above. Using the fact that $\mathbf{ad}_{\boldsymbol{\eta}}\boldsymbol{\eta} = \mathbf{0}_6$, we can introduce the vector $\boldsymbol{\mathcal{M}}\mathbf{ad}_{\boldsymbol{\eta}}\boldsymbol{\eta}$ to (5.20) without affecting the dynamics. The importance of this modification originates from the preservation of passivity in the Lagrangian model, which is an important property in stability theorems for robotics. By substitution of the null vector, the equation of motion becomes

$$\boldsymbol{\mathcal{M}}\dot{\boldsymbol{\eta}} + \left(\boldsymbol{\mathcal{M}}\mathbf{ad}_{\boldsymbol{\eta}} - \mathbf{ad}_{\boldsymbol{\eta}}^{\top}\boldsymbol{\mathcal{M}}\right)\boldsymbol{\eta} = \boldsymbol{\mathcal{F}}, \tag{5.21}$$

which is nothing more than the Newton-Euler equation for rigid-body motion on $\mathbb{R}^3$. To introduce the (reduced-order) Cosserat kinematics and make the expression symmetric, we substitute (5.16) and (5.18) into (5.21) and pre-multiply by $[\boldsymbol{J}]_k^{\top}$:

$$[\boldsymbol{J}]_k^{\top}\left(\boldsymbol{\mathcal{M}}[\boldsymbol{J}]_k\ddot{\boldsymbol{q}} + \boldsymbol{\mathcal{M}}[\dot{\boldsymbol{J}}]_k\dot{\boldsymbol{q}} + \boldsymbol{\mathcal{C}}_{[\boldsymbol{\eta}]_k}\dot{\boldsymbol{q}}\right) = [\boldsymbol{J}]_k^{\top}\left(\boldsymbol{\mathcal{F}}_u - \boldsymbol{\mathcal{F}}_{\mathrm{g}} - \boldsymbol{\mathcal{F}}_{\mathrm{e}}\right), \tag{5.22}$$

where $\boldsymbol{\mathcal{C}}_{(\cdot)} = -\boldsymbol{\mathcal{C}}_{(\cdot)}^{\top} := \boldsymbol{\mathcal{M}}\mathbf{ad}_{(\cdot)} - \mathbf{ad}_{(\cdot)}^{\top}\boldsymbol{\mathcal{M}}$ is a skew-symmetric matrix. It is important to note that by pre-multiplication of the transpose Jacobian, we have eliminated the constraint wrenches, *i.e.*, $[\boldsymbol{J}]_k^{\top}\boldsymbol{\mathcal{F}}_c = \mathbf{0}_n$ [154]. Finally, the finite-dimensional dynamics of the deformable soft robot is found by spatial integration of (5.22) over the material domain $\mathbb{X}$. The overall dynamics can be written in the familiar Euler-Lagrangian (EL) form as follows

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{f}_{\mathrm{g}}(\boldsymbol{q}) + \boldsymbol{f}_{\mathrm{e}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{\tau}(\boldsymbol{q}, \boldsymbol{u}) \tag{5.23}$$

with the system matrices:

$$\boldsymbol{M}(\boldsymbol{q}) = \int_{\mathbb{X}} [\boldsymbol{J}]_k^\top \boldsymbol{\mathcal{M}} [\boldsymbol{J}]_k \, d\sigma, \tag{5.24}$$

$$\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \int_{\mathbb{X}} [\boldsymbol{J}]_k^\top \left(\boldsymbol{\mathcal{M}} [\dot{\boldsymbol{J}}]_k + \boldsymbol{\mathcal{C}}_{[\boldsymbol{\eta}]_k} [\boldsymbol{J}]_k \right) \, d\sigma, \tag{5.25}$$

$$\boldsymbol{f}_{\mathrm{g}}(\boldsymbol{q}) = \int_{\mathbb{X}} [\boldsymbol{J}]_k^\top \boldsymbol{\mathcal{F}}_{\mathrm{b}} \, d\sigma, \tag{5.26}$$

$$\boldsymbol{f}_{\mathrm{e}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \int_{\mathbb{X}} [\boldsymbol{J}]_k^\top \boldsymbol{\mathcal{F}}_{\mathrm{e}} \, d\sigma := \boldsymbol{K}\boldsymbol{q} + \boldsymbol{D}\dot{\boldsymbol{q}}, \tag{5.27}$$

$$\boldsymbol{\tau}(\boldsymbol{q}, \boldsymbol{u}) = \int_{\mathbb{X}} [\boldsymbol{J}]_k^\top \boldsymbol{\mathcal{F}}_{u} \, d\sigma := \boldsymbol{G}(\boldsymbol{q})\boldsymbol{u}, \tag{5.28}$$

where $\boldsymbol{M}$ is the generalized inertia matrix, $\boldsymbol{C}$ the Centripetal-Coriolis matrix, $\boldsymbol{f}_{\mathrm{g}}$ a vector of generalized potential forces with $\boldsymbol{\mathcal{F}}_{\mathrm{b}} = -\mathbf{Ad}_{[\boldsymbol{g}]_k}^\top \boldsymbol{\mathcal{M}} \boldsymbol{a}_b$ the external wrench acting on the body due to body-frame accelerations $\boldsymbol{a}_b$ (like gravity), and $\boldsymbol{\mathcal{F}}_e$ a vector of visco-elastic forces imposed by the stiffness matrix $\boldsymbol{K} \succ 0$ and damping matrix $\boldsymbol{D} \succ 0$. The stiffness matrix and damping matrix are computed through spatial integration similar to standard finite element methods:

$$\boldsymbol{K} = \int_{\mathbb{X}} \boldsymbol{\Theta}^\top \boldsymbol{\mathcal{K}} \boldsymbol{\Theta} \, d\sigma, \tag{5.29}$$

$$\boldsymbol{D} = \int_{\mathbb{X}} \boldsymbol{\Theta}^\top \boldsymbol{\mathcal{D}} \boldsymbol{\Theta} \, d\sigma, \tag{5.30}$$

where $\boldsymbol{\mathcal{K}} \in \mathrm{se}^*(3) \times \mathrm{se}(3)$ and $\boldsymbol{\mathcal{D}} \in \mathrm{se}^*(3) \times \mathrm{se}(3)$ are the stiffness and damping tensor, respectively. The vector $\boldsymbol{G}\boldsymbol{u}$ represents the distributed forces $\boldsymbol{n}_{\mathrm{u}}(\sigma, t)$ and moments $\boldsymbol{m}_{\mathrm{u}}(\sigma, t)$ generated by various kinds of the internal actuators (*e.g.*, tendons or pneumatics). We will later detail the derivation of $\boldsymbol{G}(\boldsymbol{q})$ in this chapter as it depends highly on the geometrical placement of the soft actuators. If $\mathrm{rank}(\boldsymbol{G}) < \dim(\boldsymbol{q})$, the system (5.23) is said to be underactuated. Within the context of soft robotics, whose infinite-dimensional configuration space cannot be matched by a finite number of actuators, these systems are often intrinsically under-actuated.

o compute the Lagrangian matrices (5.24)-(5.30), that all involve a spatial integration over $\mathbb{X} = [0, L]$, we rewrite these integrations into one global matrix differential equation and solve it efficiently using a Matrix-Differential solver proposed in Chapter 4. As such, these system matrices are not computed exactly but approximated with sufficient precision. The numerical precision depends on the stepsize of the discretized spatial domain $\mathbb{X}$.

Assumption 5.5 The inertia matrix $\boldsymbol{M}(\boldsymbol{q})$ is a symmetric, positive definite,

symmetric. and is uniformly bounded such that there exists positive constants $\lambda^- \leq \lambda^+$ such that $\lambda^- \boldsymbol{I}_n \preceq \boldsymbol{M}(\boldsymbol{q}) \preceq \lambda^+ \boldsymbol{I}_n < \infty$.

The uniform boundedness of the inertia matrix $\boldsymbol{M}(\boldsymbol{q})$ is a well-known property for rigid manipulators composed of purely rotational joints. According to Spong et al. [201] the inertia matrix in this case is composed solely of $\cos(\cdot)$ and $\sin(\cdot)$ terms, and thus it is bounded for all $\boldsymbol{q} \in \mathcal{Q}$. The physical interpretation is that the mass and moment of inertia of the robot must be finite and positive for all configurations. This does not always hold true for instance for robots with prismatic joints. Yet, bounds can be imposed on the configuration space $\boldsymbol{q} \in \mathcal{Q}$ that enable boundedness, see Ghorbel et al. [78]. In Della Santina et al. [59], it was shown that $\boldsymbol{M}(\boldsymbol{q}) = \boldsymbol{M}^\top(\boldsymbol{q}) \succeq 0$ for soft manipulators restricted to variable planar curvatures. The property holds for the infinite dimensional case where $k \to \infty$.

Lemma 5.1. *Given the inertia matrix $\boldsymbol{M}(\boldsymbol{q})$ and the Coriolis matrix $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ as described by (5.24) and (5.25), respectively, it can be shown that the matrix $\dot{\boldsymbol{M}} - 2\boldsymbol{C}$ is skew-symmetric.* ▲

Proof. To show $\dot{\boldsymbol{M}} - 2\boldsymbol{C}$ is skew-symmetric, we start by computing the time-derivative of the inertia matrix. For sake of clarity, we abbreviate the Jacobian matrices $[\boldsymbol{J}]_k = \boldsymbol{J}$ and $[\dot{\boldsymbol{J}}]_k = \dot{\boldsymbol{J}}$. Through chain differentiation, we find

$$\dot{\boldsymbol{M}} = \int_{\mathbb{X}} \left(\dot{\boldsymbol{J}}^\top \boldsymbol{\mathcal{M}} \boldsymbol{J} + \boldsymbol{J}^\top \boldsymbol{\mathcal{M}} \dot{\boldsymbol{J}} \right) d\sigma, \tag{5.31}$$

Then, calculating $\dot{\boldsymbol{M}} - 2\boldsymbol{C}$ leads to

$$\dot{\boldsymbol{M}} - 2\boldsymbol{C} = \int_{\mathbb{X}} \dot{\boldsymbol{J}}^\top \left(\boldsymbol{\mathcal{M}} \boldsymbol{J} - \boldsymbol{J}^\top \boldsymbol{\mathcal{M}} \dot{\boldsymbol{J}} - 2\boldsymbol{J}^\top \boldsymbol{\mathcal{C}} \boldsymbol{J} \right) d\sigma. \tag{5.32}$$

Since the matrix $\boldsymbol{J}^\top \boldsymbol{\mathcal{C}} \boldsymbol{J}$ is skew-symmetric, the remainder of the proof is to show that the matrix $\boldsymbol{S} = \dot{\boldsymbol{J}}^\top \boldsymbol{\mathcal{M}} \boldsymbol{J} - \boldsymbol{J}^\top \boldsymbol{\mathcal{M}} \dot{\boldsymbol{J}}$ also satisfies skew-symmetry. Since the inertia tensor is symmetric $\boldsymbol{\mathcal{M}} = \boldsymbol{\mathcal{M}}^\top$, we can easily show this holds true:

$$\begin{aligned} \boldsymbol{S} &= \dot{\boldsymbol{J}}^\top \boldsymbol{\mathcal{M}}^\top \boldsymbol{J} - \boldsymbol{J}^\top \boldsymbol{\mathcal{M}}^\top \dot{\boldsymbol{J}}, \\ &= -\left(\dot{\boldsymbol{J}}^\top \boldsymbol{\mathcal{M}}^\top \boldsymbol{J} - \boldsymbol{J}^\top \boldsymbol{\mathcal{M}} \dot{\boldsymbol{J}} \right)^\top = -\boldsymbol{S}^\top. \end{aligned} \tag{5.33}$$

Therefore, the matrix $\dot{\boldsymbol{M}}(\boldsymbol{q}) - 2\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ is skew-symmetric for all $\boldsymbol{q}, \dot{\boldsymbol{q}} \in \mathbb{R}^n$ □

In literature, Lemma 5.1 is often referred to as the passivity condition [154, 159, 201]. It implies that, in the absence of external dissipation, the total energy of the system (*i.e.*, the Hamiltonian) is conserved. It is also worth mentioning that this condition does not necessarily hold true for any representation of the Coriolis matrix only for particular computations of the Coriolis matrix $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$ (*e.g.*, through the Christoffel symbols).

5.2.6 Tendon and fluidic actuation

Up to now, we have not discussed the derivation of the actuation mapping $\boldsymbol{G}(\boldsymbol{q})$. The matrix namely depends on (i) the type of actuation, and (ii) how actuation fields are distributed along the backbone curve of the soft robot manipulator. Earlier, we introduced the notion of a distributed forces $\boldsymbol{n}_{\mathrm{u}}$ and distributed moments $\boldsymbol{m}_{\mathrm{u}}$ acting on the curve $\boldsymbol{g}$, which are collocated into a control wrench $\boldsymbol{\mathcal{F}}_{\mathrm{u}}$. Now, assuming that a general soft manipulator system is either actuated using a set of fluidic chambers or a set of tendons, the control wrench related this network of m number of soft actuators can be represented in the form [27, 169]:

$$\boldsymbol{\mathcal{F}}_{\mathrm{u}}(\sigma, \boldsymbol{u}) = \sum_{i=1}^{m} \begin{pmatrix} \boldsymbol{r}_i^{\times}(\sigma)\boldsymbol{t}_i(\sigma) \\ \boldsymbol{t}_i(\sigma) \end{pmatrix} u_i =: \boldsymbol{\Theta}_{\mathrm{u}}(\sigma)\boldsymbol{u}, \tag{5.34}$$

where $\boldsymbol{r}_i : \mathbb{X} \to \mathbb{R}^3$ is the radial distance between the actuator center-line and the backbone curve $\boldsymbol{\gamma}$, and $\boldsymbol{t}_i : \mathbb{X} \times \mathcal{Q} \to \mathbb{R}^3$ the tangent vector along the actuator center-line expressed that deforms with the continuum body prescribed by $[\boldsymbol{g}]_k(\sigma, \boldsymbol{q})$, $\boldsymbol{\Theta}_{\mathrm{u}} \in \mathbb{R}^{6\times m}$ a shape function matrix related to the spatial distribution of the soft actuators, and $u_i(t)$ a time-varying scalar related to the activation amplitude of the i-th soft actuator. Generally speaking, for tendon actuation $u_i = -\max\{0, \lambda_i\}$ with λ_i the cable tension and for pneumatic actuation we have $u_i = A_i p_i$ with $A_i > 0$ an effective pressure surface and p_i the applied pressure. Following (5.34), the generalized joint torques can then be computed by projecting the distributed control wrench onto the configuration space using the geometric Jacobian matrix. Then, $\boldsymbol{G}(\boldsymbol{q})$ is computed by integrating generalized distributed joint forces over $\mathbb{X}$:

$$\boldsymbol{\tau}(\boldsymbol{q}, \boldsymbol{u}) = \int_{\mathbb{X}} \boldsymbol{J}^{\top}(\boldsymbol{q}, \sigma)\boldsymbol{\Theta}_{\mathrm{u}}(\sigma)\, d\sigma\, \boldsymbol{u} =: \boldsymbol{G}(\boldsymbol{q})\boldsymbol{u}. \tag{5.35}$$

Similar to the other Lagrangian matrices, the actuation mapping is included into the matrix differential solver to allow fast and efficient computation of the system (5.23).

5.2.7 Port-Hamiltonian formulation

In this section, the Lagrangian model in (5.23) is rewritten in port-Hamiltonian form. To this end, we define the generalized momenta $\boldsymbol{p} := \boldsymbol{M}\dot{\boldsymbol{q}}$. Then, the (reduced-order) Hamiltonian is given by $\mathcal{H}(\boldsymbol{q}, \boldsymbol{p}) := \mathcal{K}(\boldsymbol{q}, \boldsymbol{p}) + \mathcal{U}(\boldsymbol{q})$ with $\mathcal{K} = \frac{1}{2}\boldsymbol{p}^{\top}\boldsymbol{M}^{-1}\boldsymbol{p}$ and $\mathcal{U}(\boldsymbol{q})$ the kinetic and potential energy, respectively.

Given the system's Hamiltonian $\mathcal{H}$, it can be shown that generalized velocities can be written in terms of partial derivatives of the Hamiltonian function

$$\dot{\boldsymbol{q}} = \nabla_{\boldsymbol{p}}\mathcal{H} \quad \Longrightarrow \quad \nabla_{\boldsymbol{p}}\mathcal{H} := \left(\frac{\partial \mathcal{H}}{\partial \boldsymbol{p}}\right)^{\top} = \boldsymbol{M}^{-1}\boldsymbol{p}. \tag{5.36}$$

where we denote $\nabla_{\boldsymbol{x}}(\cdot) := \partial(\cdot)^{\top}/\partial \boldsymbol{x}$ as the gradient w.r.t. a vector field $\boldsymbol{x}$. Note that $\boldsymbol{M}^{-1}$ is always exists due to the positive definiteness condition in Lemma 5.5. Similarly, we seek a differential description that relates the time evolution of $\boldsymbol{p}$ to a state-derivative of the Hamiltonian function. Applying the chain rule of differentiation to the generalized momenta:

$$\begin{aligned} \dot{\boldsymbol{p}} &= \dot{\boldsymbol{M}}\dot{\boldsymbol{q}} + \boldsymbol{M}\ddot{\boldsymbol{q}}, \\ &= \left(\dot{\boldsymbol{M}} - \boldsymbol{C} - \boldsymbol{D}\right)\boldsymbol{M}^{-1}\boldsymbol{p} - \boldsymbol{K}\boldsymbol{q} - \boldsymbol{f}_{\mathrm{g}} + \boldsymbol{G}\boldsymbol{u}, \end{aligned} \tag{5.37}$$

Taking the partial derivative of the Hamiltonian $\mathcal{H}$ w.r.t. the generalized coordinates $\boldsymbol{q}$, we find

$$\nabla_{\boldsymbol{q}}\mathcal{H} = \frac{1}{2}\nabla_{\boldsymbol{q}}\left[\dot{\boldsymbol{q}}^{\top}\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{q}}\right] + \nabla_{\boldsymbol{q}}\mathcal{U}. \tag{5.38}$$

To relate (5.37) and (5.38), we explores some structural properties in the Lagrangian model. To be more specific, we exploit the skew-symmetry condition as detailed in Lemma 5.1. According to the Spong et al. (2006, [201]), if the matrix $\dot{\boldsymbol{M}} - 2\boldsymbol{C}$ satisfies the skew-symmetrycondition in Lemma 5.1, it can be shown that

$$\left(\dot{\boldsymbol{M}} - 2\boldsymbol{C}\right)\dot{\boldsymbol{q}} = -\nabla_{\boldsymbol{q}}\left[\dot{\boldsymbol{q}}^{\top}\boldsymbol{M}(\boldsymbol{q})\dot{\boldsymbol{q}}\right] - \dot{\boldsymbol{M}}\dot{\boldsymbol{q}}. \tag{5.39}$$

Finally, by combining (5.36), (5.37), (5.38), and (5.39), we can show that the Lagrangian model in (5.23) can be equivalently rewritten as a standard port-Hamiltonian (pH) system:

$$\Sigma_{\text{beam}} : \begin{cases} \dot{\boldsymbol{q}} &= \nabla_{\boldsymbol{p}}\mathcal{H}, \\ \dot{\boldsymbol{p}} &= -\nabla_{\boldsymbol{q}}\mathcal{H} - \boldsymbol{D}\nabla_{\boldsymbol{p}}\mathcal{H} + \boldsymbol{G}\boldsymbol{u}. \end{cases} \tag{5.40}$$

The advantage of the port-Hamiltonian system Σ_{beam} over the standard EL structure in (5.23), and as also seen in Chapter 4, is the general applicability to different physical domains and the common formalism with energy-based control, which we will explore further in the next section.

5.3 Energy-based controller for tasks on SE(3)

Given the reduced port-Hamiltonian model in (5.40), our objective here is to find a controller $\boldsymbol{u}$ that ensures $\lim_{t\to\infty} \boldsymbol{g}(L,t) = \boldsymbol{g}_d$ in which $\boldsymbol{g}_d \in \mathrm{SE}(3)$ denotes the desired configuration of the end-effector. To achieve the control objective, the main idea is to reshape the potential energy function of the reduced-order finite-dimensional system using a standard energy-shaping techniques, common to standard port-controlled Hamiltonian models [160, 182].

We adopted an energy-based control strategy akin to the works of Franco et al. [70], Ortega et al. [159, 160] and Schaft [182]. Following the energy-shaping strategy, the model-based nonlinear controller becomes

$$\boldsymbol{u} = \boldsymbol{G}^{+}\left(\nabla_{\boldsymbol{q}}\mathcal{H} - \nabla_{\boldsymbol{q}}\mathcal{H}_d\right), \tag{5.41}$$

where $\boldsymbol{G}^{+} = \left(\boldsymbol{G}^{\top}\boldsymbol{G}\right)^{-1}\boldsymbol{G}^{\top}$ is the generalized inverse of the actuation map $\boldsymbol{G}$, and the desired Hamiltonian in closed loop $\mathcal{H}_d = \frac{1}{2}\boldsymbol{p}^{\top}\boldsymbol{M}^{-1}\boldsymbol{p} + \mathcal{U}_d$ that satisfies $\operatorname{argmin}_{\boldsymbol{g}_L}\mathcal{U}_d = \boldsymbol{g}_d$ with $\boldsymbol{g}_L = \boldsymbol{g}(L,\cdot)$ the pose of the end-effector. Note that we purposefully omitted any damping injection as the system's intrinsic visco-elastic damping is deemed sufficiently large to guarantee stability. Following the concept of a kinematic feedback controller [27, 32] that artificially mimic an elastic element between the end-effector and the desired configuration in SE(3), we have choose the gradient of the desired potential energy as

$$\nabla_{\boldsymbol{q}}\,\mathcal{U}_d = \lambda_1 \boldsymbol{J}^{\top}\left(\boldsymbol{J}\boldsymbol{J}^{\top} + \lambda_2 \boldsymbol{I}\right)^{-1}\boldsymbol{\mathcal{F}}_u, \tag{5.42}$$

where $\lambda_1 > 0$ is a proportional gain, $\lambda_2 > 0$ a controller gain related to the artificial damping of the pseudo-inverse, $\boldsymbol{\mathcal{F}}_u = k_p T_{\mathrm{SE}(3)}(\boldsymbol{\mathcal{E}})\boldsymbol{\mathcal{E}}$ an artificial control wrench with positive definite stiffness matrix k_p, $\boldsymbol{\mathcal{E}} = \log_{\mathrm{SE}(3)}([\boldsymbol{g}_L]_k^{-1}\boldsymbol{g}_d)$ where $\log_{\mathrm{SE}(3)}(\cdot)$ and $T_{\mathrm{SE}(3)}(\cdot)$ denote the logarithmic map (see Appendix B.1) and the tangent-space map, respectively. We refer the reader to Sonneville et al. (2014, [198]) for the numerical computations of the tangent map on SE(3). The vector $\boldsymbol{\mathcal{E}}$ may be regarded as the geometric error between the homogeneous transformations $\boldsymbol{g}_d$ and $\boldsymbol{g}_L$ such that if $\boldsymbol{g}_d = \boldsymbol{g}_L$ will simply yield $\|\boldsymbol{\mathcal{E}}\|_2 = 0$. Furthermore, the controller gains λ_1 and λ_2 can be tuned accordingly to tweak the desired transient behavior of the closed-loop system, similar to a classical PD controller.

5.4 Simulation studies of bio-inspired robots

In this section, we detail the numerical simulations of the port-Hamiltonian model in (5.40) together with the energy-shaping controller in (5.41).

Due to the partial differential nature, we have to employ a nested ODE routine to recover the trajectories for $\boldsymbol{q}$ and $\boldsymbol{p}$. First, we employ an implicit trapezoidal solver with a fixed stepsize of $dt = 30$ ms to solve (5.40). At each time increment, we have to evaluate the dynamic matrices (5.24)-(5.28). To efficiently compute these dynamic entities, we solve the spatial integration problem over the material domain $\mathbb{X}$ by using a second-order Runge-Kutta solver. The stepsize for the spatial solver is chosen sufficiently small $\Delta\sigma = 1$ mm. The spatial stepsize is related to the length of the manipulator L, but also the degree of geometric deformations along the backbone. In some cases, high-gain feedback control can excite high-order modes thus requiring a smaller stepsize to capture these phenomena accurately. However, by keeping the control gains small paired with the fact that structural damping is dominant in soft elastomers often lead to slow transients (order of seconds). As such, the spatial discretization has been shown to relatively nonrestrictive. . All simulation examples and underlying source code are provided publicly on the *Sorotoki* toolkit [34]. Here, the numerical integrations of the system matrices (5.24), (5.25), (5.27) and (5.28) are performed using a so-called Matrix-Differential solver (see [35]). The simulations are performed on a modern machine (Ryzen 7-5800H, 3.2GHz).

For the soft robotic simulations, we have chosen a linear isotropic Hookean material with shear constraints. Given these material properties, the inertia tensor and the stiffness tensor become diagonal matrices:

$$\boldsymbol{\mathcal{M}} = \text{blkdiag}\left\{\rho_0 \boldsymbol{\mathcal{J}},\, \rho_0 A \boldsymbol{I}_3\right\};$$

$$\boldsymbol{\mathcal{K}} = \text{blkdiag}\left\{\mu_1 \boldsymbol{\mathcal{J}},\, E A_0,\, \mu_2 A_0,\, \mu_2 A_0\right\},$$

where A_0 is the (average) cross-sectional area, and $\boldsymbol{\mathcal{J}}$ the second moment of area. Note that A_0 and $\boldsymbol{\mathcal{J}}$ depend on the cross-sectional geometry of the soft robot and is therefor unique for each system. The damping tensor chosen as $\boldsymbol{\mathcal{D}} = \zeta \boldsymbol{\mathcal{K}}$ with damping coefficient ζ. The visco-elastic matrices are precomputed using (5.29) and (5.30).

5.4.1 Soft robot manipulator inspired by octopus' tentacle

In the first study-case, we consider a soft robotic arm that is loosely inspired by the tentacles of an octopus. To introduce the under-actuation typically present in soft robotics, we have chosen an actuation matrix $\boldsymbol{G} = \text{blkdiag}\left\{\boldsymbol{I}_5, \boldsymbol{O}_3\right\}$ such that only the first five modes are actively controllable. We assume that the soft manipulator has a circular cross-section, thus the cross-sectional area is given by $A_0 = \pi r^2$ with radius r. The system properties are shown in Table 5.1.

Table 5.1. Parameters setting for the solver, the soft robot, and controller.

Parameter description	Symbol	Value	Unit
Finite horizon time	T	10	s
Intrinsic length	L	120	mm
Cross-section radius	r	8.25	mm
Uniform density	ρ_0	1250	kgm^{-3}
Young's modulus	E	25	MPa
Shear modulus	μ_1	10	MPa
Constraint modulus	μ_2	15	GPa
Rayleigh coefficient	ζ	0.40	-

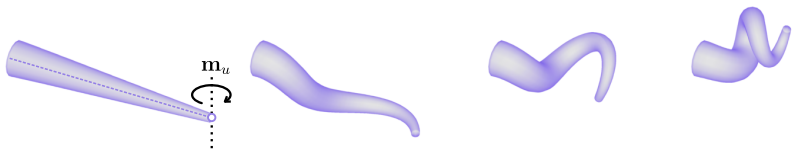


Figure 5.4. Three-dimensional deformation of the soft tentacle due to an external pure-moment wrench of $|\boldsymbol{m}_u| = 75$ N mm acting on its end-effector.

Example 5.2 (Solution convergence for increasing modal order)**.** Before investigating the performance of the controller, we study the affects of the truncation order k. Initially, it can be difficult find the optimal value for k that balances precision and computational speed. Consider the case where we subject the soft robot by a external tip wrench $\tau_{\text{ext}} = [J]_k^\top (\boldsymbol{m}(t), 0, 0, 0, 0, 0)^\top$ given by a predefined moment $\boldsymbol{m}(t) = 75 \cdot \text{erf}(t)$ N mm. The applied moment causes a spiraling deformation of the soft arm, as can be seen in Figure 5.4. Observe that the soft arm undergoes a rather complex nonlinear three-dimensional deformation. Now, we inclemently increase the model truncation parameter $k \in \{1, ..., 10\}$ and investigate the convergence of the solutions $\boldsymbol{\gamma}_L$ (*i.e.*the end-effector location). The numerical results are shown in Figure 5.5. Furthermore, we compare these simulation runs with a high-order mode of $k = 25$, where investigate the root mean square error, absolute error, and the ratio between actual simulation time T_{sim} and finite horizon T. The benchmark is shown in Figure 5.6. From these results, we observe that the solutions quickly converge to a narrow error band, namely for $k > 3$ the error does not seem to improve significantly. The RMS error is about 3.25 mm or 2.7% normalized by L. Regarding computation time, we observe a linear decrease for a linear increase in modal order k. Note that for $k > 8$, the simulations are close to losing real-time performance. We therefore choose a modal basis of order $k = 8$.

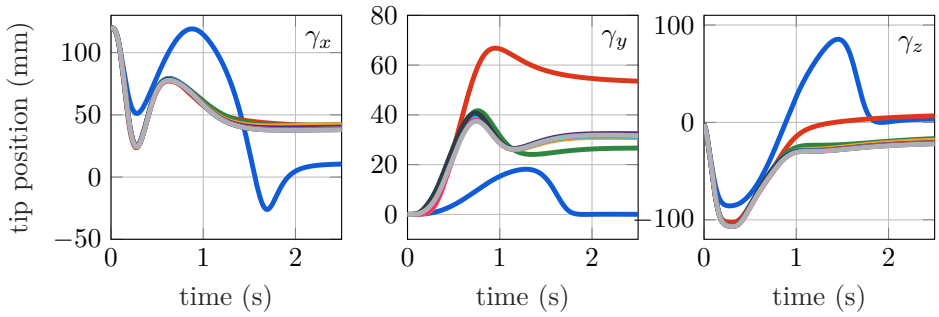


Figure 5.5. Temporal evolutions of the end-effector locations for increasing modal number k, where (left) is the x-position, (middle) the y-position, and (right) the z-position. The modal order is color-coded as follows: $k \in \{-1, -2, -3, -4, -5, -6, -7, -8, -9, -10\}$. Notice that the trajectories of the end-effector $\boldsymbol{\gamma}_L$ converge after $k > 3$.

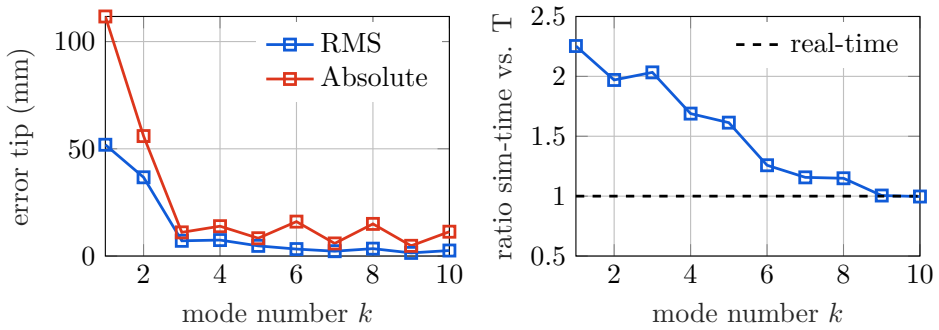


Figure 5.6. The benchmark results of the soft tentacle subjected to a tip-load for increasing model order k. (left) The root mean square error (RMSE) in (–) and absolute error in (–) for increasing k. Notice that the errors converge to an error band of about 3.25 mm. (right) The ratio between simulation time and finite horizon time. All simulations achieve real-time computation.

Example 5.3 (Energy-based controller)**.** Following, we now subjected the soft robot to the proposed energy-based controller in (5.41), where the control gains are tuned to produce a smooth transient: $\lambda_1 = 0.01$ and $\lambda_2 = 0.001$. The artificial spring stiffness is chosen as $\boldsymbol{k}_p = \text{blkdiag}\{0.01 \cdot \boldsymbol{I}_3, \boldsymbol{I}_3\}$. Lastly, the desired

configuration of the end-effector is chosen as follows:

$$\boldsymbol{g}_d = \begin{pmatrix} \boldsymbol{I}_3 & \boldsymbol{r}_d \\ \boldsymbol{0}_3^\top & 1 \end{pmatrix} \quad \text{with} \quad \boldsymbol{r}_d = \begin{pmatrix} 0.04 \\ 0.00 \\ -0.01 \end{pmatrix}.$$

The numerical results of the closed-loop system are shown in Figure 5.7 and 5.8. It is worth mentioning that these simulation run at ±120 Hz real-time.[1]

Figure 5.7 shows the evolution of the continuous deformation along the soft robotic body, whereas Figure 5.8 shows the modal coefficients $\boldsymbol{q}(t)$ and generalized momenta $\boldsymbol{p}(t)$. As can be seen, the end-effector of the soft robot manipulator converges to the desired set-point $\boldsymbol{g}_d \in \mathrm{SE}(3)$. Although the control gains could be increased to promote a faster transient, it was observed that high gains lead to undesired (propagating) oscillations of the flexible structure. A possible solution might be to introduce negative damping to the controller Hamiltonian $\mathcal{H}_d$, to overcome the soft robot's structural damping.

For the second simulation run, we modified the control gains to highlight an interesting property of the proposed controller. To be more specific, we increase the controller gains to $\lambda_1 = \lambda_2 = 0.1$. The numerical results for the increased controller gains are shown in Figure 5.9 and Figure 5.10. Although the control goal and the initial conditions are chosen identical, the soft robot converges to a different configuration – albeit, a shape with less '*complexity*'. The cause of less complicated bending patterns has two origins. First, increasing the control gains also artificially impacts the structural stiffness of the soft robot, resulting in soft robot with a higher perceived stiffness. Second, by increasing the stabilizing term λ_2 in the damped Jacobian inverse (5.42), more weight is given towards finding a solution that also minimizes the joint angles $\|\boldsymbol{q}\|_2$. This is to be expected, as intrinsically, it requires more energy to achieve higher-order modes deformations $\{\theta_n\}_{n=1}^k$ of the Legendre basis. Hence, the energy-based controller will thus find a minimizer that accounts for the ordering of the shape basis, penalizing higher-order modes. To some extent, the high intrinsic damping in soft robots acts as a spatial low filter which is beneficial for both modeling as control. This effect is not observed in classical semi-rigid robots since structural damping is often not dominant in metal composites. This result indicates that the proposed controller can be effectively tuned alter the structural compliance of the soft robot; and thus could be implemented carefully to preserve '*softness*'.

[1] Real-time bandwidth is determined by the ratio between finite horizon and CPU's computation time, *i.e.*, $f \approx T/T_{\text{sim}}$,which is affected most by spatial stepsize due to explicit integration.

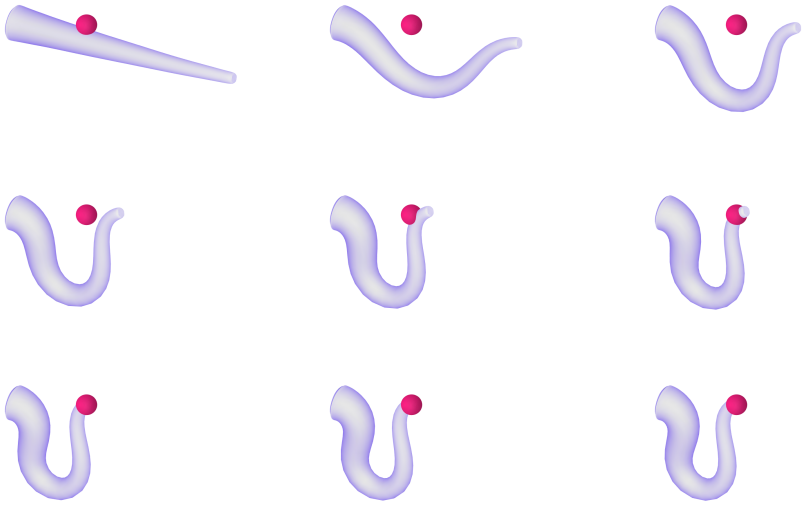


Figure 5.7. Three-dimensional evolution of the soft robot manipulators, converging to the desired set-point $g_d \in \mathrm{SE}(3)$ (indicated by the pink ball). Observe that the morphological motion that arises due to the energy-based controller has a close similarity to those an octopus' tentacle.

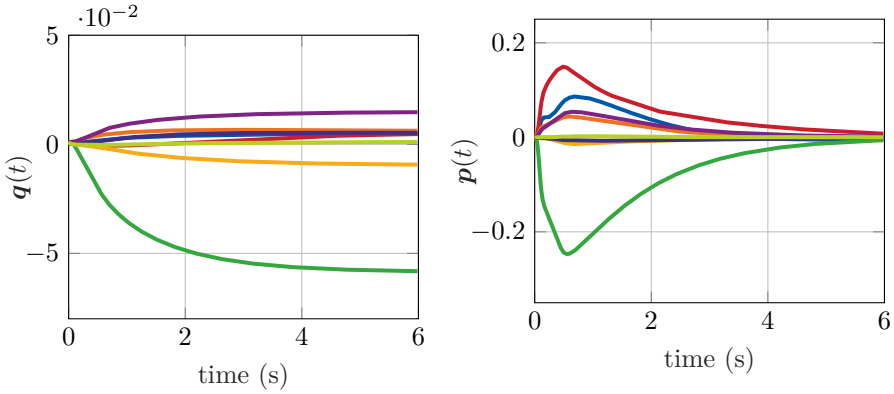


Figure 5.8. The evolution of the modal coefficients and the generalized momenta of the soft robot manipulator. The modal coefficients $\boldsymbol{q}$ are ordered as follows: $k \in \{-1, -2, -3, -4, -5, -6, -7, -8\}$. Observe that mainly mode 3 is dominant.

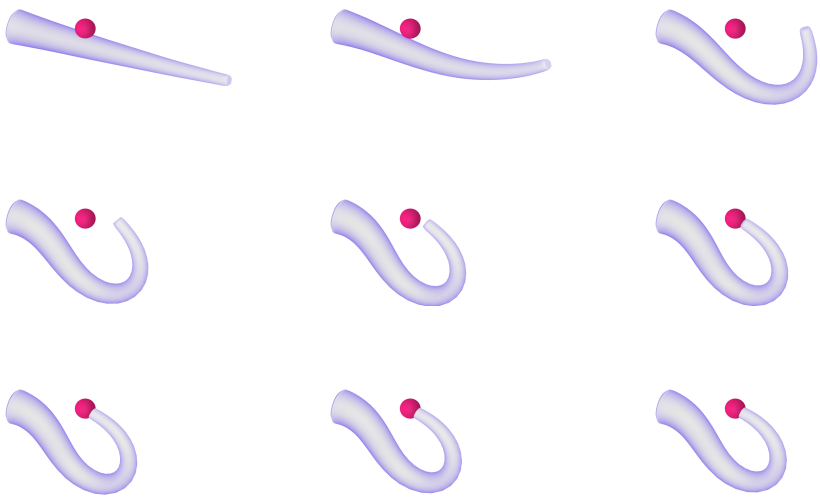


Figure 5.9. Three-dimensional evolution of the soft robot manipulators, converging to the desired set-point $g_d \in \mathrm{SE}(3)$ (indicated by the pink ball). Observe that a different morphology arises due to higher control gains, *i.e.*, $\lambda_1 = \lambda_2 = 0.1$, which is caused by the controller affecting the structural compliance of the soft robot.

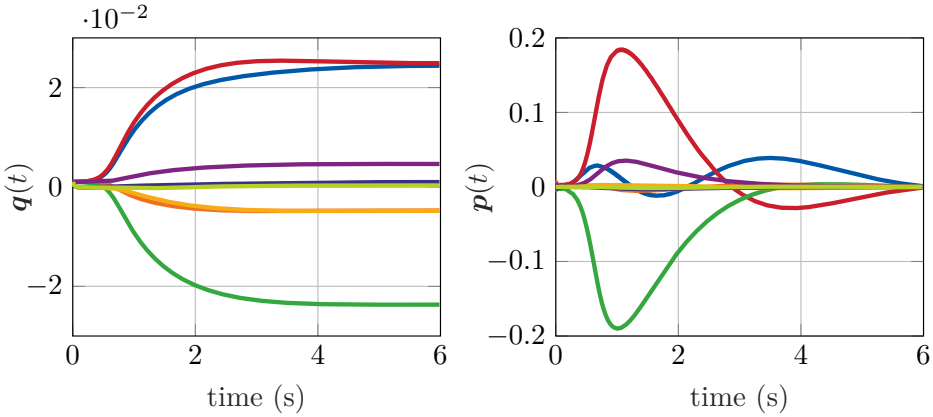


Figure 5.10. The evolution of the modal coefficients (left) and the generalized momenta (right) of the soft robot manipulator with the increased controller gains. The modal coefficients $\boldsymbol{q}$ are ordered as follows: $k \in \{-1, -2, -3, -4, -5, -6, -7, -8\}$. Observe now that modes $1-3$ are dominant.

5.4.2 Multi-link soft robot inspired by the elephant's trunk

In the second study-case, we consider a two-link soft robot that is inspired by the trunk of an elephant. A similar soft robotic system is considered in [65] (*i.e.*, the elephant-inspired bionic arm by `Festo`), where mobility of the bio-inspired robotic system is achieved through a pneumatic-network distributed along the continuous body of the robot. Therefore, considering a six-bellow network, the actuation matrix takes the form:

$$\boldsymbol{G}(\boldsymbol{q})\boldsymbol{u} = \sum_{n=1}^{6} \left[\int_{\mathbb{X}} [\boldsymbol{J}]_k^\top (\sigma, \boldsymbol{q}) \boldsymbol{\theta}_{\mathrm{u},i}(\sigma)\, d\sigma \right] u_i,$$

where $\{\boldsymbol{\theta}_{\mathrm{u},i}\}_{n=1}^{6}$ is a set of piecewise constant wrench functions related to the pneumatic actuation bellows distributed along the soft robotic body, and $\boldsymbol{u} = (u_1,\, u_2,\, u_3,\, u_4,\, u_5,\, u_6)^\top$ a vector of wrench amplitudes. The control input sets $\{u_1, .., u_3\}$ relate to the first link and $\{u_4, .., u_6\}$ to the second link of the robot. Given this input configuration, it also follows that $\mathrm{rank}(\boldsymbol{G}(\boldsymbol{q})) < \dim(\boldsymbol{q})$ for all $\boldsymbol{q} \in \mathbb{R}^{6k}$, *i.e.*, underactuated. The system and solver properties are given in Table 5.2. We again consider $k = 8$ spatial modes. To simulate the effect of the gripper, we added an inertial mass at the end-effector modelled by:

$$\boldsymbol{\tau}_{\mathrm{ext}} = \boldsymbol{\mathcal{M}}_{\mathrm{grip}}\, \boldsymbol{J}(\boldsymbol{q}, L)^\top \left[\left(\mathbf{0}_3^\top,\, \boldsymbol{a}_g^\top \mathbf{Ad}_{\boldsymbol{g}(\cdot,L)}^\top \right)^\top + \dot{\boldsymbol{\eta}}(\boldsymbol{q}, L) \right],$$

where $\boldsymbol{\mathcal{M}}_{\mathrm{grip}}$ inertia tensor related to the gripper placed at the end-effector of the robot located at $\sigma = L$. Again we apply the energy-based controller in (5.41) to the system, where the control gains are $\lambda_1 = 5$ and $\lambda_2 = 1$, while the artificial stiffness matrix $\boldsymbol{k}_p$ is kept identical to previous simulations. Lastly, the desired configuration of the end-effector is chosen as follows:

$$\boldsymbol{g}_d = \begin{pmatrix} \boldsymbol{\Phi}_d & \boldsymbol{r}_d \\ \mathbf{0}_3^\top & 1 \end{pmatrix} \quad \text{with} \quad \boldsymbol{r}_d = \begin{pmatrix} 0.125 \\ 0.100 \\ 0.175 \end{pmatrix} \quad \text{and} \quad \boldsymbol{\Phi}_d = \mathrm{Rot}_y\left(\tfrac{1}{4}\pi\right).$$

The numerical results of the closed-loop system are shown in Figure 5.11 and Figure 5.12 which could reach a real-time performance of ±65Hz. Figure 5.11 shows the continuous deformation along the soft robotic body. Figure 5.12 shows the trajectories of the modal coefficients $\boldsymbol{q}(t)$ and the generalized momenta $\boldsymbol{p}(t)$. As can be seen, the end-effector of the soft robot manipulator slowly converges to the desired set-point $\boldsymbol{g}_d \in \mathrm{SE}(3)$, even when dealing with piece-wise distributed actuation loads applied to the continuous backbone. To describe the discontinuous actuation profiles, however, higher order modes are required as can be seen in Figure 5.12. This might indicate there exist better tailored compact shape bases for this soft robotic system.

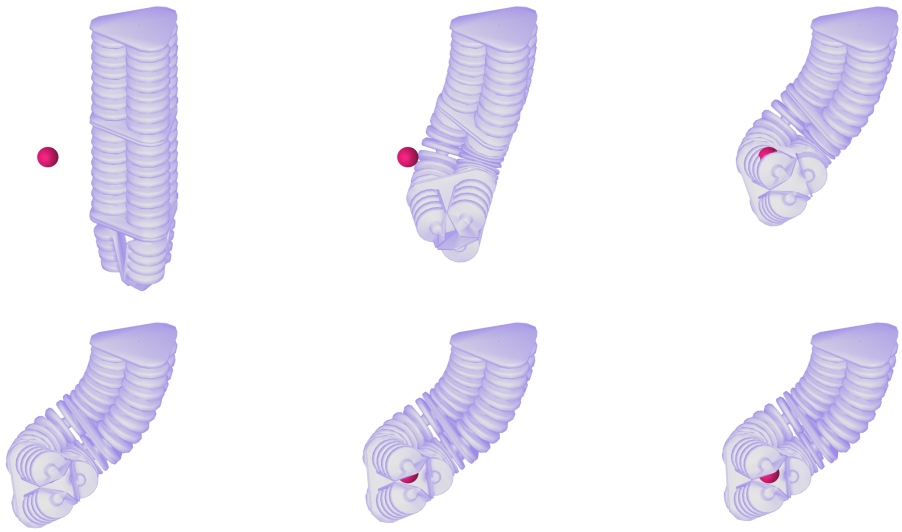


Figure 5.11. Three-dimensional evolution of the soft robot inspired by the elephant's trunk (whose muscular network is mimicked through six pneumatic bellows), slowly converging to the desired set-point $\boldsymbol{g}_d \in \mathrm{SE}(3)$ (*i.e.*, the pink ball).

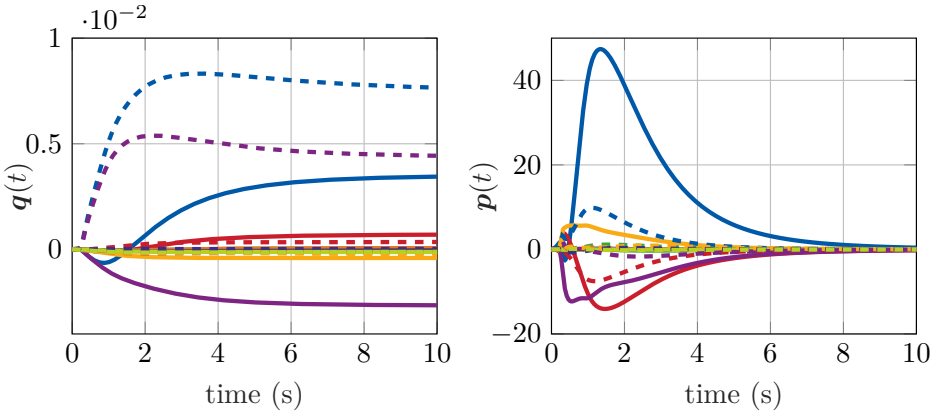


Figure 5.12. The evolution of the modal coefficients (left) and the generalized momenta (right) of the soft robot manipulator inspired by the elephant's trunk. The modal coefficients $\boldsymbol{q}$ are ordered as follows $k \in \{-1, -2, -3, -4, -5, -6, -7, -8\}$ where the full and dashed lines represent the first and second link, respectively. Observe that mainly modes 1 and 5 are dominant in both links.

Table 5.2. Parameters setting for the numerical solver, the soft manipulator by the elephant's trunk, and the energy-based controller.

Parameter description	Symbol	Value	Unit
Finite horizon time	T	20	s
Intrinsic length	L	360	mm
Cross-section radius	r	25	mm
Uniform density	ρ_0	250	kgm^{-3}
Young's modulus	E	35	MPa
Shear modulus	μ_1	20	MPa
Constraint modulus	μ_2	15	GPa
Rayleigh coefficient	ζ	0.45	-

5.5 Conclusion

Due to their intrinsic compliance of soft robots, they allow for complex morphological motions that mimic animals in nature. Achieving similar performance to biology highlights the need for more accurate dynamic models and control strategies. In this chapter, we provided a modeling framework for Cosserat beams that leads to a finite-dimensional system in a port-Hamiltonian structure. By exploiting the passivity, an energy-shaping controller was proposed that ensures the closed-loop Hamiltonian is minimal at the desired set-point. The numerical model was developed for a several bio-inspired soft robot (octopus' tentacle and elephant's trunk) with distributed control inputs. The key challenges here regarding both the model as the controller are their ability to capture the hyper-flexibility, deal with inherent under-actuation, and exploit its hyper-redundancy to achieve its control task. Given appropriate controller gains, the model-based controller yields smooth convergence of the soft robot's end-effector while accounting for under-actuation. It was shown that by tuning the controller gains, the intrinsic stiffness of the soft body can be adapted, resulting in significantly different quasi-static joint solutions of the set-point problem. To some extent, the mobility of the Cosserat model paired with the energy-based control has a (close) resemblance to the biological motion. There are, however, a few limitations to our approach. The strain parametrization of functional basis does not account for the geometry of the soft robot, meaning some systems require many spatial modes to *accurately* represent the true continuum dynamics. Second, regarding implementation, measuring these spatial modes in an experimental setting is difficult, and future research is required to find a suitable '*soft sensing*' technique that *(i)* has limited impact on the dy-

namics, and *(ii)* accounts for the continuity of the elastic body. A possible solution might be the optimal placement of a network of distributed localized sensors, *e.g.*, strain gauges or IMUs. Furthermore, the proposed controller is only suited for set-point regulation or slow-varying references. Exploring (fast)-dynamic control objectives will likely require more research. In particular, controllers that suppress natural resonances of continuum elastic body under fast motion. One could argue that this perhaps fights against the natural dynamics of the soft robot, yet such oscillations might be able to be explored for locomotion or soft manipulators throwing objects rather than traditional pick-and-place strategies.

Given these limitations, future work will focus on the following: *(i)* hyper-elasticity *(ii)* validating the controller experimentally, and *(iii)* constructing a set of basis functions through the so-called '*snapshot decomposition method*' using FEM-driven data. In particular, the latter two goals could be interesting to explore. Both advantages in FEM and Cosserat models, being accurate continuum deformations and computational efficiency; leading to a modeling strategy for *optimal* finite-dimensional state projection with insightful structure of the passive and active joints.

IV

Open-access Software for Soft Robots

6

Sorotoki: A Matlab Toolkit for the Design and Control of Soft Robots

Abstract - In this chapter, we present *Sorotoki*, an open-source toolkit in MATLAB that offers a comprehensive suite of tools for the design, modeling, and control of soft robots. The complexity associated with the study and development of soft robots frequently arises from the interdependence between the areas of design and control, which are rarely treated as a joint problem. To address the complex interdependencies in soft robotics, the *Sorotoki* toolkit provides a comprehensive and modular programming environment composed of seven Object-Oriented classes. These classes are designed to work together to solve a wide range of soft robotic problems, offering versatility and flexibility for its users. We provide a comprehensive overview of the *Sorotoki* software architecture to highlight its usage and capacities. The details and interconnections of each module are thoroughly described, collectively explaining how to build up complexity of modeling in soft robotics. The effectiveness of *Sorotoki* is also demonstrated through a range of case studies, including novel problem scenarios and established works widely recognized in the soft robotics community. These case studies cover a broad range of research problems in the field of soft robotics, including: inverse design of soft actuators, passive and active soft locomotion, object manipulation with soft grippers, meta-materials, model reduction, model-based control of soft robots, and shape estimation. Additionally, the toolkit provides access to four open-hardware soft robotic systems that can be fabricated using commercially available 3D printers.

This chapter is based on: B.J. Caasenbrood, A.Y. Pogromsky, and H. Nijmijer. *Energy-shaping Controllers for Soft Robot Manipulators through Port-Hamiltonian Cosserat Models.* SN Computer Science, Springer, 2022. doi: 10.1089/soro.2021.0035.

6.1 Introduction

Since the late 1980s, roboticists have been developing fluid-driven robots inspired by biological systems. Examples include the pneumatic three-link soft robot manipulator developed by Wilson et al.[238–240] and the fluidic four-fingered soft gripper presented by Suzumori et al.[203, 204], which both showed high dexterity for advanced object manipulation without the need for advanced (contact-aware) controllers. Although the design and control of these robots were simple, their level of dexterity and adaptability was previously unseen in rigid robotics and strongly resembled biological systems. These benefits were achieved through the use of "soft materials" paired with fluidic actuation, where the term "soft" refers to the collective mechanical properties of highly compliant materials such as flexibility, compressibility, and mechanical robustness. Just as nature exhibits diverse evolutionary solutions to environmental locomotion and manipulation, soft materials possess a plethora of beneficial mechanical properties that can be applied to robotics. Today, the philosophy of building robots from soft materials has significantly matured and has become a well-recognized field known as "*soft robotics*".

Due to their mechanical composition, soft robots offer several potential advantages compared to traditional rigid robots [177]. First, these robotic systems are less likely to cause injury during collisions, making them a more suitable alternative for tasks involving close human-robot interaction. Secondly, soft robots possess the ability to adapt to complex environments and manipulate a diverse array of objects through their ability to change shape and conform adaptively to their surroundings [100, 133]. These features are somewhat analogous to those found in nature, such as the trunk of an elephant grasping tree branches or the tentacles of an octopus squeezing through narrow spaces. Soft materials can be tuned to allow for delicate grasping [73, 189] or high-power densities [128]. Additionally, their high adaptability paired with tunable low compliance make them extremely robust towards abrupt impacts or high compression forces [16], making them suitable for complex tasks in harsh environments without breaking [87, 127, 222]. Moreover, soft robots are typically constructed from low-cost materials and fabricated through straightforward manufacturing processes, such as rubber casting, making them more cost-effective compared to traditional rigid robots. With recent advances in soft material Additive Manufacturing (AM), soft robots can even be fully 3D-printed [229, 246]. This not only reduces production time and cost, but also allows for the embedding of printed on-board logic [99, 237]. Additionally, soft robots tend to be lightweight, making them easier to handle and transport, making them ideal for wearable robotics.

6.1.1 Problem formulation

Although significant progress has been made since their inception, generalized solutions for the design and control of soft robots are still lacking in comparison to those available for rigid robots. This can be partially attributed to the inherently nonlinear and high-dimensional nature of the mathematical descriptions for deformable robotic bodies composed of soft materials. This presents major challenges in finding suitable models that enable fast simulation, which ultimately hinders efficient structural design and model-based controller design. Despite the numerous challenges in soft robotics, two major research trends can be recognized within the soft robotics community:

Design of soft actuators and sensors (a): A majority of the soft robots are actuated in two ways [177]: (*i*) local actuation through variable length tendons (e.g., cables [170] or shape-memory alloy wires [6]) or (*ii*) distributed actuation through responsive soft materials [228] or surface loads using fluidics [65, 72, 87], commonly implemented as a fluidic networks embedded within the soft body. The latter method is often referred to as Fluidic Elastomer Actuators (FEAs) or Soft Fluidic Actuators (SFA). FEAs can be designed through either geometric asymmetries in their structural design or by incorporating a composition of different materials, such as fibers or meshes, that induce the desired deformation when pressurized. While FEA designs date back to the 1980s, there remain significant gaps in understanding and applying established engineering principles to their design. FEAs frequently experience large deformation when actuated, leading to slow actuation due to material relaxation or, in more severe cases, fatigue or tearing caused by ballooning [133] – almost inherent to the elastomeric material effects of FEAs. Furthermore, low-compliance soft elastomer actuators often undergo parasitic deformation when exposed to external forces, like gravitational load. To efficiently solve the design cycle in soft actuators, it is essential to have a comprehensive understanding of the nonlinear deformation characteristics of soft materials under static and dynamic conditions.

On the contrary, proprioceptive soft sensing technology is still in early stages. The high compliance of soft robots often makes it difficult to apply common embedded sensors, such as encoders, capacitive sensors, strain gauges, and inertial sensors. These sensors are well-suited for rigid robots with articulated joints, as they are effective in measuring local joint displacement. However, in soft robots, displacement is often distributed, rendering these sensors less suitable. Furthermore, these sensors must be designed to minimize their impact on mechanical impedance, in order to minimize changes to the structural dynamics and operational workspace. A common approach is to incorporate microfluidic channels filled with a conductive liquids [161, 206, 212], such as Eutectic Gallium-Indium

(EGaIn), into the soft body, that are placed antagonistic to the soft actuator. Upon deformation, the resistance changes, allowing for the correlation of specific deformation profiles with the soft robot. Other solutions incorporate integrated Hall sensors to measure changes in the magnetic field of ferromagnets distributed throughout the body [14, 190], or utilize fiber-optic grating bending sensors [74]. Generally speaking, the relationship between sensor output and deformation is non-trivial and often requires the extensive acquisition of *a-priori* measurement data (sometimes synthetic) to be mapped onto a set of motion primitives.

The geometry of soft actuators and sensors plays a crucial role in determining their functionality and performance. Currently, most soft robotic components are designed using Computer-Aided Design (CAD) software similar to those used for rigid robots. However, as the geometric complexity of soft robots increases, particularly with the increasing trend towards bio-inspired and 3D-printed designs, there is a growing need for software that can handle free-form designs and have predictive capabilities for soft material deformation.

Closed-loop control of soft robots (b): With the aim of achieving comparable performance to rigid robots and eventually biological creatures, there is a strong demand for advanced closed-loop control in soft robotic systems. However, the challenges in soft actuation and sensing extend directly to several modeling and control paradigms for soft robots [10, 58]: (*i*) their high dexterity and adaptability are challenging to incorporate into a modeling framework, and (*ii*) due to their continuum elastic bodies composed of a finite number of actuators and sensors, soft robots are inherently under-actuated and under-sensed, a problem common to infinite-dimensional systems (*e.g.*, continuum systems). The field has presented two branches that can deal with the issues at hand related to closed-loop control of soft robots.

Model-based control uses first-principle mathematical models of the system being controlled to design and implement controllers. The derived models often have a conventional structures, *e.g.*, Lagrangian or Port-Hamiltonian, that is (closely) analogous to classic rigid robotics [10, 58, 154, 201]. As such, they extend (with minor modification) to existing control strategies including model-based feedback control [65, 144], impedance control [60], adaptive control [112, 227], iterative learning control [89], and energy-shaping control [26, 39, 69, 70]. Also, model-based approaches provide physical interpretations of the control gains, making controllers more transparent in terms of stability guarantees. Nonetheless, for some scenarios, first-principle models will not suffice. For example, during environment or self-contact, it can be challenging to select a finite-dimensional state representation of the soft robotic model that balances precision and computational efficiency. Also, *a-priori* unknown system uncertainties, such as unreliability of

sensors and actuators, model mismatches, and time-varying parameters, can impede the approach altogether.

In data-driven modeling, the emphasis is on utilizing available data to characterize the relationship between inputs and outputs, rather than relying on prior knowledge or assumptions to formulate a theoretical model. The training data can be obtained either from real measurements or high-fidelity surrogate models, enabling the model to adapt as new information becomes available similar to adaptive control. Although these models are often black box (or grey box) approximator, their low dimensionality allows for very efficient simulations. As an alternative, synthetic data generated from digital environments can be used to train learning controllers, such as Reinforcement Learning (RL) [183, 211]. Model-Predictive Control (MPC) can also be applied within a data-driven framework [31, 101]. However, in both cases, it is crucial that the training data be comprehensive enough to encompass the entire dynamic workspace. This requirement limits the generalizability of the method to unseen scenarios and necessitates retraining for specific control objectives. Additionally, control policies learned through virtual environments may not be effective in the physical system due to differences with reality, known as the Simulation-to-Reality (Sim2Real) barrier [121].

6.1.2 Contribution of Sorotoki software

To address some of these challenges, we introduce *Sorotoki* (short for Soft Robotics Toolkit), an open *MATLAB* toolkit for soft robotics that offers a range of tools for design, modeling, and control. *Sorotoki* aims to reduce barriers to entry in the field of soft robotics by providing a comprehensive software package that integrates various layers of modeling and control approaches, including continuum mechanics, dynamic systems and control theory, topology optimization, computer graphics, real-time control, and vision-based sensing. These diverse capabilities provide a highly flexible programming environment that can facilitate the development of innovative soft robotics research. The main feature of the *Sorotoki* are listed below:

1. **Design and fabrication** – Implicit modeling using Signed Distance Functions (SDFs), mesh generation, computational design, STL generation for 3D printing;
2. **Modeling and control** – Finite Element Models (FEM), high-efficiency reduced-order soft beam models (Lagrangian or port-Hamiltonian), programmable interconnections of a network of dynamic systems, e.g., soft robots, pressure reservoirs, and inertial rigid bodies.
3. **Actuation and sensing** – Real-time, high-precision, fluidic control platform using Raspberry Pi, vision-based sensing using RGB-depth camera;

4. **Visualization** – Fast and responsive 3D graphics rendering, mesh deformation modifiers, FK/IK-rigging.
5. **Accessibility** – A minimal programming syntax, characterized by the ability to express complex problems with a minimal number of lines of code.
6. **Open hardware** – Four 3D-printable soft robots (e.g., soft hand, soft manipulator).

6.1.3 Organization of the paper

We briefly detail the organization and navigation of the paper. Section 6.2 reviews existing open-source soft robot software packages. Section 6.3 then assists the reader in getting started with the toolkit and introduces the open-source soft robotic systems. In Section 6.5, the reader is informed of the software architecture, the theory underlying the *Sorotoki* functions, and how the theory can be applied through coding examples in *Sorotoki*. Once the reader is familiar with the basic software architecture, Section 6.6 presents advanced study cases based on seminal works in soft robotics research. The paper concludes in Section 6.7 with a summary and outlook for future work.

6.2 Related works

Over the past two decades, significant advancements have been made in the field of soft robotics. To support the growing community, researchers in the field have made efforts to provide open-source software tools alongside their scientific contributions. This section provides a review of related work on open-source software packages for soft robotics, comparing these software packages and discussing how *Sorotoki* addresses any gaps in functionality.

One widely used tool is the SOFA (Stand Alone Open Framework for Animation) software [62], which is an open-source framework for real-time physically-based simulations of mechanical systems. Relevant to soft robotics, SOFA is commonly used to simulate the behavior of soft robots and to design and test control algorithms on real platforms [51, 62]. SOFA employs the Finite Element Method (FEM) to describe the continuum deformations of inertial elastic bodies. FEM is a numerical technique that solves partial differential equations (PDEs) that describe physical systems by dividing the domain of the system into small elements and approximating the PDEs with a set of algebraic equations [95, 117]. FEM models generally provide high-accuracy volumetric deformation simulations of soft materials, but their high state dimension, which can often be in the thousands or millions

of degrees of freedom, can render them computationally expensive for state feedback. To enhance efficiency, Goury et al. [82] have explored model reduction using snapshot Proper Orthogonal Decomposition (POD).

Snapshot POD [12, 18] is a method for significantly reducing the dimensionality of a model by collecting snapshots of its state and utilizing Singular Value Decomposition (SVD) to identify the principal components. The projection is achieved by taking a linear combination of the principal components, weighted by their corresponding coefficients (also known as "modes"). The resulting projection is then a reduced-order model of the original system, which can be used for faster simulation. In addition to improving speed, this approach also provides accurate, robust, and efficient models suitable for closed-loop controller design [5, 216, 223, 242]. The numerical FEM model in *SOFA* incorporates both the structural geometry and material properties, facilitating the transfer of control policies to physical systems. This has enabled successful control synthesis using *SOFA* in various experimental settings [111, 243]. *SOFA* also includes tools for real-time visualization and data analysis, making it a valuable platform for testing and debugging control algorithms. Recently, Schegg et al. [183] introduced an interface between *SOFA* and *OpenAI* called *SofaGym*. This wrapper enables the training of reinforcement learning (RL) policies using real-time simulation models, and it incorporates model reduction to further improve the efficiency of RL that otherwise suffer from computationally-intensive simulations.

Another software package that utilizes the nonlinear finite element approach similar to *SOFA* is the Gibbon toolbox, developed by Moerman et al. [149]. *Gibbon* is a MATLAB-based pre-processor and post-processor for FEBio [132]. The toolkit has recently been used to solve the nonlinear deformation of bending soft pneumatic actuators using finite shell elements [195], generate designs using a multi-objective heuristic [196], and analyze soft bending actuators composed of an adaptive fiber-elastomer composite [108]. It also features various tools for image segmentation, meshing, and visualization, with a focus on biomedical engineering.

Despite the availability of open-source FEM packages for modeling and controlling soft robots, challenges still exist in using FEM for design-based optimization of these systems, particularly due to the complexity of hyper-elastic materials and fluidic actuation, which are crucial in the field of soft robotics. Currently, there are limited options for frameworks that effectively and efficiently address these issues, although recent developments are promising. Smith et al. [197] recently proposed a versatile free-form design and fabrication workflow called *SoroForge*, which builds upon [195, 196]. Unlike volumetric representation, their approach can design complex soft actuator exteriors using a highly-flexible and fast node-tree interface of implicit function primitives. However, these generative CAD solutions

are limited to only addressing quasi-static deformations and do not consider the deformation induced by control.

There are several software packages specifically designed for the dynamic locomotion of soft robots, which take into account the structural design, actuator placement within the soft body, control actions, and even adapt the body's topology accordingly. One such example is *EvoSoro* developed by Kriegman et al. [121], which builds on the work of Hiller et al. [88] and Cheney et al. [40]. This study discretizes a soft continuum body into small voxels, which can be assigned different cell types: soft or hard passive cells, or two different muscle cell types that undergo periodic contraction with an $+\pi$ phase offset. The dynamic behavior of the system is modeled through a network of mass particles and springs, and a Compositional Pattern-Producing Network (CPPN) is utilized to determine the optimal combination of material type and placement within a specified domain, enabling locomotion. This concurrent optimization of topology and control policy, referred to as *"co-design"*, is a subject of active research within the field of soft robotics [246]. Another example of co-design in soft robotics is *EvoGym* [24], which optimizes for a wide range of tasks such as locomotion and object manipulation (e.g., carrying and throwing). More recently, *DiffTachi* (the successor to *QueenChain* [98]) is a differential programming environment that allows users to directly provide gradient-based information into a neural network controller using a least-squares Material Point Method (MPM). Unlike FEM, MPM is a mesh-free approach that describes the continuum using a finite number of hybrid Euler-Lagrangian elements referred to as "material points". In terms of learning control policies, *SoMoGym* by Graule et al. [83] uses reinforcement learning (RL) to teach locomotion and object manipulation in soft robots, and has successfully bridged the gap between simulation and reality (Sim2Real). It is important to note that *QueenChain*, *DiffTachi*, and *SoMoGym* focus purely on learning control and not design. Simultaneous optimization of (free-form) design and control for soft robots remains an open challenge.

Parallel to volumetric-based FEM or MP soft robotic software, there also exists a branch of dynamic beam (or rod) models for soft robots. These beam model approaches for soft robots have long been a viable alternative to FEM-based models – examples include the Piecewise-Constant Curvature (PCC) model [35, 65, 202], the augmented PCC rigid-body model [60, 227**?**], and various non-constant curvature descriptions [27, 60, 169, 234]. As the formulations of these models are often synonymous to rigid robot models, they have a rich basis of control-oriented research [26, 35, 67**?**]. *TMTDyn* by Sadati et al. [178] is a *MATLAB* toolkit that automates the derivation of dynamic models for hybrid rigid-continuum body soft robots, based on discretized lumped systems and reduced-order models. More

recently, *SoroSim* was developed by Mathew et al. [139], which is a *MATLAB* toolbox with a graphical user interface for modeling, analysis, and control of soft, rigid, and hybrid robots. *SoroSim* is based on the Geometric Variable Strain (GVS) approach applied to the geometric Cosserat beam theory in SE(3), introduced relatively recently by Renda et al. [169] and Boyer et al. [27]. Its Lagrangian formulation also allows for various complex control designs, such as a geometrically-exact inverse kinematic controller that accounts for underactuation in tendons [139]. *SoroSim* has been used for dynamic models of flexible flying rods, hybrid rigid-soft manipulators, design optimization for soft robot swimmers, and inverse dynamic control. The toolbox has also been employed for the simultaneous swimming and grasping dynamics of underwater soft-rigid hybrid robots [137].

Alternatively, *PyElastica* (a wrapper for *Elastica* [158]) by Tekinalp et al. [211] is an open-source software package written in *Python* that provides the capability to simulate an assembly of Cosserat beams. The software is based on the work of Gazzola et al. [77]. Unlike the Geometric Variable Strain (GVS) approach proposed in [27, 139, 169], *PyElastica* employs a discrete formulation of the Cosserat partial differential equation (PDE) through a finite number of line elements referred to as Discrete Elastic Rods (DERs). Additionally, its `c++` architecture enables it to handle problems with higher computational complexity. As demonstrated in [251], *Elastica* has been used to model snake muscular systems, bird wing flapping, and bio-hybrid robots using soft contractile filaments. Furthermore, *PyElastica* has been utilized in energy-based control and even extended to model the full muscular-skeletal system of an octopus' tentacle modeled as an array of Cosserat beams [38], and mimic biological movements accordingly [232]. It also provides wrappers for *OpenAI* to enable reinforcement learning.

To summarize, Table 6.1 provides an overview of the functionalities of the previously discussed software packages for soft robotics. The table highlights the diversity of software options available, each with specialized solution approaches for specific sub-problems in soft robotics. However, there are limited tools that address the combined issues of design and control in an interdependent, holistic manner. For instance, developing a model-based controller for a specific soft robot heavily relies on its structural geometry, material composition, network of soft actuators, and their dynamic interaction with the soft body. On the other hand, finding a suitable soft robot design requires a priori knowledge of the material properties and their deformation under the admissible control inputs. This complex interplay between design, modeling, and control makes studying and developing soft robots a challenging task. `Sorotoki` aims to address these challenges by integrating the scientific disciplines of design and control into a unified toolkit.

Table 6.1. Comparison between different open-source software provided by the soft robotics community that are tailored either towards design, modeling, or control of soft robots. *Inverse design here refers to automated algorithms that freely optimize the topology of the soft body. †Sim2Real here implies the software has been used on a real soft robotic platform – eitehr successfully transferring open or closed-loop control policies to reality; or optimized design solution to real soft systems.

Software	Front-end	Model(s)	Hyperelastic	Tendon	Fluidic	Locomotion	Manipulation	Inv. Design*	Control	Sim2Real†
SOFA [51, 63]	Python	(Reduced) FEM	✓	✓	✓	✓	✓	✗	✓	✓
Gibbon [149]	Matlab	FEM	✓	✓	✓	✗	✗	✓	✗	✗
SoRoForge[197]	Matlab	Shell-FEM	✓	✗	✓	✗	✗	✓	✗	✓
SoftIK[21]	C++	FEM	✓	✓	✗	✓	✓	✗	✓	✓
EvoGym[24]	Python	Mass-spring	✗	✗	✗	✓	✓	✓	✗	✗
EvoSoro[40, 121]	Python	Mass-spring	✗	✗	✗	✓	✓	✓	✗	✓
DiffTachi [98]	C++	MPM	✗	✗	✗	✓	✓	✓	✗	✗
TMTDyn [139]	Matlab	ROM + EBA	✗	✓	✓	✗	✓	✗	✓	✓
SoroSim [139]	Matlab	Cosserat	✗	✓	✓	✓	✓	✗	✓	✗
(Py)Elastica [77]	Python	Cosserat	✗	✓	✗	✓	✓	✗	✓	✗
SoMoGym [83]	Python	Rigid-link	✗	✗	✓	✓	✓	✗	✓	✓
Sorotoki (ours)	Matlab	FEM + Cosserat	✓	✓	✓	✓	✓	✓	✓	✓

6.3 Getting started with Sorotoki

In the following section, we briefly detail a starter's guide for Sorotoki. The software package *Sorotoki* is available via git:

```
$ git clone https://github.com/BJCaasenbrood/SorotokiCode.git
```

To install the toolkit, request the documentation, or preview demonstrations of the toolkit, we call

```
> sorotoki install      % installation toolkit
> sorotoki doc          % provides documentation
> sorotoki demo         % provides example scripts
```

in the *Matlab* command prompt, respectively. The online documentation provides information on the features and capabilities, installation instructions, general use and syntaxing. The documentation assists users in comprehending and efficiently navigating it for their research requirements. It serves as a complement to the work presented herein.

6.4 Open-source soft robots of Sorotoki

Before discussing the software, we present a selection of open-source soft robotic systems, as illustrated in Figure 6.1, which are part of the *Sorotoki* toolkit. These systems all feature fluidic actuation and can be fabricated using conventional additive manufacturing techniques such as selective laser sintering (SLS), stereolithography (SLA), or direct light projection (DLP). Formlabs Elastic 80A resin or a flexible TPU with a shore hardness of $\leq$ 80A are suggested for the elastically deformable bodies in SLA/DLP and SLS printing, respectively. Moreover, these soft robots can also be realized using multi-material Fused Deposition Modeling (FDM) with both flexible filaments (e.g., NinjaFlex) and water-soluble filaments. Nevertheless, FDM-fabricated systems may suffer from inferior quality, air-tightness, and anisotropy when compared to those produced by SLS or SLA/DLP. For that reason, we recommend SLS/SLA/DLP over FDM for its simplicity and reliability. For further details concerning the SLS/DLP manufacturing process, consult Proper et al. [**?**]. The 3D models (in `.stl` format) are publicly available through Git:

```
> git clone https://github.com/BJCaasenbrood/SorotokiBots
```

A brief description of these soft robotic systems and their functionality is provided below.

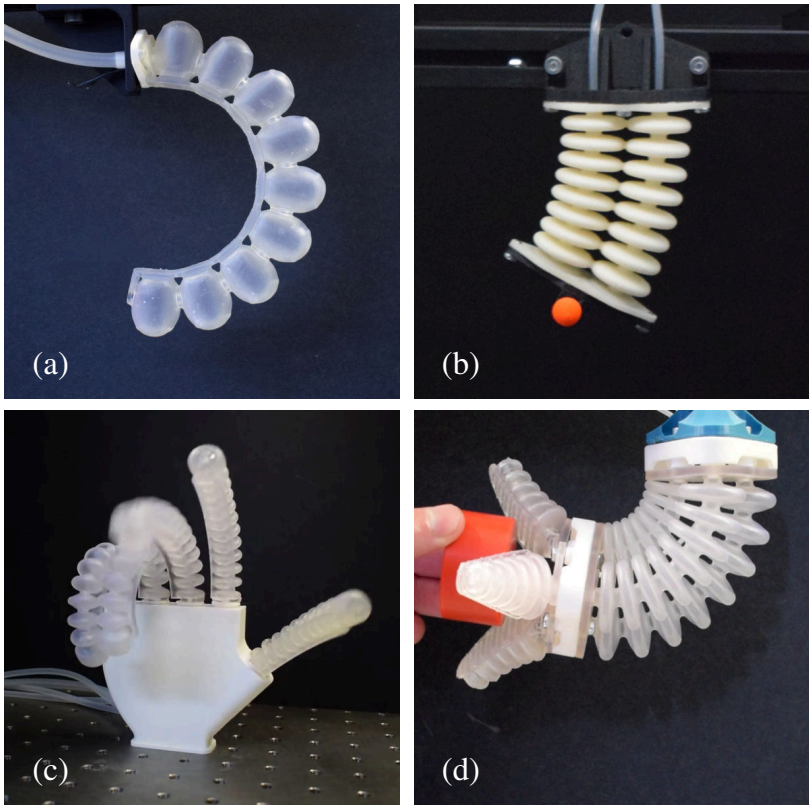


Figure 6.1. Open-source soft robots and soft actuators that are included within the *Sorotoki* toolkit. All systems are fully 3D-printed using either Selective Laser Sintering (SLS) or Stereolithography (SLA) and their 3D files can be found on the repository. All systems are driven by pneumatics. (a) A two-bellow soft robot suitable for planar motion. (b) An optimized PneuNet bending actuator. (c) A soft robotic hand composed of five soft bending actuators, whose fingers are easily replaceable. (d) A three-bellow soft robot manipulator with a mounted soft gripper at the end-effector. The center axis is hollow, allowing for electronic cables when compact sensors (e.g., IMUs) are mounted on the soft gripper.

A: Soft bending actuator. The first system is a soft bending actuator (Figure 6.1a), an alternative to the PneuNet actuator proposed by Mosadegh et al. [152]. Like the PneuNet actuator, our soft actuator consists of an array of bellows placed on a relatively inextensible elastic medium. The stiffness gradient in the actuator allows for pure bending to occur when the network of bellows is pressurized. The geometry of the PneuNet-based soft robot was optimized using Sorotoki's topology optimizer, which was specifically tailored for use with Formlabs Elastic 80A resin. The soft actuator is fully 3D-printed using SLA and can accept pressures in the range of $-10 \leq u \leq 100$ kPa at its central pressure input.

B: Planar soft actuator. The second system is a planar soft actuator that comprises two pneumatic bellow networks that are connected in parallel (Figure 6.1b). Similar to the previous soft actuator, bending occurs due to a pressure differential between the two pneumatic networks. However, the system is also capable of pure elongation and contraction if the pressure in both networks is equal. This enhances the motion capabilities of the soft robot, enabling it to move within a planar workspace of approximately 100 × 100 mm. The system has two pressure range of $-10 \leq u \leq 50$ kPa.

C: Composable soft robotic hand. The third system provided by *Sorotoki* is a soft robotic hand with a higher level of complexity compared to the previous soft robots (Figure 6.1c). This design was inspired by the work of Laake et al. [**?**] and Fras et al. [72]. The soft robotic hand consists of five independently controlled soft fingers that can be actuated using pneumatics or fluidics. Each finger is fabricated using SLS with Elastic 80A, while the base is fabricated using FDM with PLA. The dimensions and scale of the soft robotic hand are similar to those of a human hand, with approximate dimensions of 190 × 100 × 40 mm. All fingers have a length of 90 mm except for the thumb, which is slightly shorter at 80 mm. The soft robotic hand has five independent inputs that accept pressures in the range of $-10 \leq u \leq 60$ kPa.

D: Full soft manipulator with soft gripper. The final soft robot provided by *Sorotoki* is a soft robotic manipulator that features three independent bellow networks and a three-fingered soft robotic gripper attached to the end effector (Figure 6.1d). With independent actuation of each bellow network, the manipulator has a full 3D workspace of approximately 150 × 150 × 150 mm. The soft elements of the manipulator are fabricated using Elastic 80A resin, while the rigid connector pieces are made using Rigid 10K resin. The gripper has demonstrated the ability to successfully grip objects with a diameter of 40 mm, with a maximum payload of 100 g without significant parasitic deformation. The central axis of the manipulator is designed to be hollow, enabling the pneumatic tubing of the gripper and the cables for state estimation sensors (e.g., IMUs) to be embed-

ded. The manipulator has three inputs that accept pressure values in the range of $-10 \leq u \leq 30$ kPa, and the gripper has one input that accepts values in the range of $-30 \leq u \leq 60$ kPa.

Remark 6.1 High-resolution 3D models are available on the following repository at https://github.com/BJCaasenbrood/SorotokiBots. These models can be sliced for 3D printing using popular slicing software such as Cura, PrusaSlicer, or PreForm. Detailed assembly instructions can be found in the documentation on the repository. Additionally, low-resolution 3D models are provided on the repository under *./assets/stl/redux* for real-time visualization of the soft body's deformations.

6.5 Software architecture

In this section, we will present the software architecture of the *Sorotoki* toolkit. The toolkit consists of seven Object-Oriented classes, each designed to address a specific sub-problem within the field of soft robotics. We will introduce each class in the following sequence:

- In Section 6.5.1 we will discuss the class `Sdf`: a Signed Distance Function (SDF) class that are used to build spatial geometries – *"Implicit CAD"*;
- In Section 6.5.2 we will discuss `Mesh` responsible for mesh generation;
- In Section 6.5.3 we discuss the class `Fem`: a Finite Element Model (FEM) solver required for high-detail soft robot simulations;
- In Section 6.5.4, we detail the class `Shapes` responsible for soft beam models required for fast soft robot simulations;
- In Section 6.5.5 we explain `Model` – a model composer to interconnect various dynamic model, and the control synthesis;
- Following, in Section 6.5.6, we highlight `Control` that serves as a control interface for fluidic platform communicating to *Matlab* via TCP/IP;
- Finally, Section 6.5.7 will explain `Vision` – a Vision-based tool for state estimation of soft robots through optical markers.

To assist the reader, we have included a software architectural flowchart in Figure 6.2 that illustrates each class. The flowchart demonstrates how the classes can be interconnected to increase system complexity while maintaining the structured

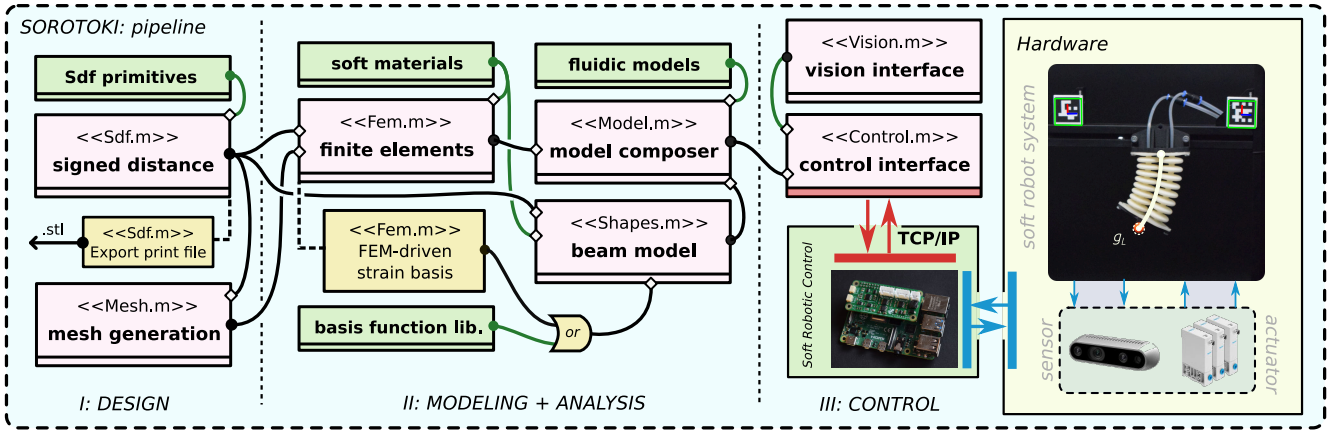


Figure 6.2. The Sorotoki software toolkit is structured according to a problem-solution pipeline, consisting of seven Object-Oriented classes that address common subproblems in the field of soft robotics research. These classes are: `Sdf`, `Mesh`, `Fem`, `Shapes`, `Model`, `Control`, and `Vision`. The software architecture flowchart employs the symbol (•) to represent class outputs and the symbol (◇) to denote inputs. It is important to note the interplay between these classes.

and separable nature of the subproblems. To further clarify their individual functionality, we will provide illustrative examples and corresponding MATLAB executable scripts. The topics covered include design, modeling and analysis, model reduction, and control and vision-based sensing, presented in this chronological order.

6.5.1 Signed Distance Function (`Sdf`)

Signed Distance Fields (SDFs) have been widely applied in various areas of computer graphics, including the representation of implicit surfaces [**?** **?**], collision detection in robotics [**?** **?**]. In particular, SDFs have gained attention for their use in implicit modeling [197], a technique for representing 3D shapes as continuous functions, rather than discrete mesh descriptions.

In *Sorotoki*, SDFs are implemented in the class `Sdf.m` and can be used to construct general 2D and 3D geometries. They can also be utilized to model static or dynamic contact environments, generate 3D models of soft actuators that are suitable for 3D printing, and compute inertia tensors for continuum bodies in $\mathbb{R}^2$ and $\mathbb{R}^3$.

A: Implicit modeling using SDFs. In this section, we briefly outline the mathematical foundations underpinning the `Sdf` class. As the name suggests, signed distance functions are a type of function that encodes distance information relative to an object defined implicitly. Adopting the notation used in [**?**], given a domain $\Omega \subset \mathbb{R}^n$ and its boundary $\partial\Omega$, these functions can be written in the following general form:

$$\texttt{sdf}(\boldsymbol{p}) = \begin{cases} -d(\boldsymbol{p}, \Omega) & \text{if } \boldsymbol{p} \in \Omega, \\ +d(\boldsymbol{p}, \Omega) & \text{if } \boldsymbol{p} \in \mathbb{R}^n \backslash \Omega, \end{cases} \tag{6.1}$$

where $d(\boldsymbol{p}, \Omega) := \inf_{\boldsymbol{y} \in \Omega} ||\boldsymbol{p} - \boldsymbol{y}||_2$ is a scalar function that returns the smallest Euclidean distance from a sample point $\boldsymbol{p} \in \mathbb{R}^n$ to the boundary $\partial\Omega$. SDFs provide a simple and efficient way of determining the location of a set of points relative to a domain Ω defined implicitly. The SDF is a scalar function that encodes the Euclidean distance of a sample point $\boldsymbol{p} \in \mathbb{R}^n$ to the boundary $\partial\Omega$ of the domain. By evaluating the sign of the SDF, it is possible to classify the set of points as being within or outside the boundary of the domain. This enables set operations such as union, difference, and intersection to be performed.

In the *Sorotoki* software package, these operations are implemented using Matlab's arithmetic operators between two or more instances of the *Sdf* class, including '`+`' (union), '`-`' (difference), '`/`' (intersection), `*` (scaling), and '`.*`' (repeating). By

utilizing these set operations and a library of basic SDF primitives, it is possible to construct a wide range of complex geometries with relative ease. Subsequently, the SDFs can be transformed into a `.stl` file using the Marching Cube algorithm [**?**], enabling 3D printing. This functionality is implemented in the command `sdf.export`.

Example 6.1 (Implicit CAD using SDFs)**.** To demonstrate the use of signed distance functions in *Sorotoki*, we present an example of 2D and 3D implicit modeling scheme as shown in Figure 6.3. This example illustrates the utilization of various SDF primitives, which are combined through standard set operations, such as union, difference, and intersection, to generate complex geometries. The accompanying code is provided below:

```
%% EXAMPLE: Sdf class
Y = 1; % design parameter

% generate 2D sdf
c   = sCircle(Y);
r   = sRectangle(Y);
sdf = r.rotate(pi/4)-c;

% generate 3D sdf
S1  = sSphere(Y/2,[0,0,Y]);
S2  = sSphere(Y,[0,0,Y/2]);
C   = sCube([0,0,0],[Y,Y,Y]);
sdf = (C - S1)/S2;
```

B: SDF differentiablility Contrary to mesh-based geometries, signed distance functions (SDFs) possess closed-form differentials. Specifically, if Ω is a

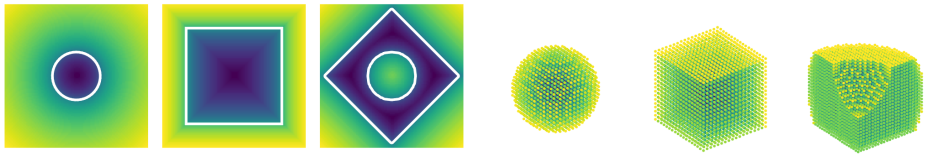


Figure 6.3. Examplary functionality of the Signed Distance Function (Sdf) operators in *Sorotoki*. The top figures are two-dimensional Sdfs, whereas below are three-dimenional Sdfs. *Sorotoki* allows the user to combine Sdf using Matlab's arithmetics, like '+', '−', and '/', to perform unions, differences, and intersections, respectively. These set operations like union, difference, and intersect lead to new (differentiable) SDFs.

subset of $\mathbb{R}^n$ with piecewise smooth boundaries, the SDF is (i) differentiable almost everywhere, and (ii) its gradient satisfies $|\nabla \texttt{sdf}| = 1$. As a result, the unit-normal vector $\boldsymbol{n}(\boldsymbol{p})$ pointing away from the boundary $\partial\Omega$ can be expressed as $\boldsymbol{n}(\boldsymbol{p}) := \nabla \texttt{sdf}(\boldsymbol{p})$. The gradient can be estimated using a finite-difference scheme:

$$\boldsymbol{n}_i(\boldsymbol{p}) \approx \frac{1}{\varepsilon}\Big[\texttt{sdf}(\boldsymbol{p} + \varepsilon\boldsymbol{\delta}_i) - \texttt{sdf}(\boldsymbol{p})\Big], \tag{6.2}$$

where $\boldsymbol{\delta}_i$ is a vectorized Kronecker delta and ε a sufficiently small incremental displacement.

In *Sorotoki*, this finite difference routine is efficiently implemented such that the normal, tangent, and bi-normal vector computations can be called using `[N,T,B] = Sdf.gradient(p)`. These gradient vector computations are crucial for contact dynamics with the environment whose topology may be arbitrarily complex. The normal vector can also be useful in finding the closest-point projection onto the surface $\partial\Omega$, namely $\text{proj}_{\partial\Omega}(\mathbf{p}) := \mathbf{p} - \texttt{sdf}(\mathbf{p}) \cdot \nabla \texttt{sdf}(\mathbf{p})$. The projection operator is implemented as `[P,d] = Sdf.project(p)`, which takes a point cloud `p` and returns a point cloud `P` that is mapped onto the boundary of the SDF. It also returns the Euclidean distance $d(\mathbf{p}, \partial\Omega)$ from the surface. This can be extremely useful in simulations of soft robotic grippers for grasping, or obstacle avoidance for soft manipulators.

6.5.2 Mesh generation (`Mesh`)

In finite elements and computer graphics, mesh tessellation is a common language used to describe the structural geometry through a finite collection of vertices and edges. In *Sorotoki*, meshes and mesh generation features are packaged into the class `Mesh.m`. In general, a mesh defines a discrete representation of a continuum body that is subdivided into smaller convex sub-volumes, referred to as "*elements*". The nodal and elemental information are stored in data structures that can be accessed using `msh.Node` and `msh.Element`, respectively. For two-dimensional FEM problems, it is common to use linear elements such as `Tri3` and `Quad4` or quadratic elements like `Tri6` and `Quad8`. For three-dimensional FEM problems, the common practice is to use hexahedron elements (*i.e.*, `Hex8`) or tetrahedral elements (*i.e.*, `Tet4` and `Tet12`). There are also polygonal tessellations, often denotes as `PolyN` finite elements [**?**]. The *Sorotoki* toolkit supports all of the above element types.

A: Mesh generation from SDFs. The *Sorotoki* toolkit explores several routines for mesh generation, which are all contained in the class `Mesh.m`. Our primary focus is on using a modified version of the *PolyMesher* software developed by Talischi et al. [**?**]. Their work provided a stable foundation for generating

unstructured meshes of `PolyN` elements. The approach starts by defining a material domain implicitly using SDFs (as discussed in Section 6.5.1). The number of elements is chosen a priori, and then repeated random sampling of Equation (6.1) is performed until the number of samples that fall within the specified domain matches the number of elements. A bounded Voronoi diagram is generated using the samples and the centers of the Voronoi cells are updated using Lloyd's algorithm [**?**]. To generate a mesh from an `Sdf` class, one can call `msh = Mesh(Sdf)` followed by `msh = msh.generate()`.

Example 6.2 (Meshing of SDFs)**.** We provided a mesh generation example in Figure 6.4 where we used the SDF function from the previous example to generate our tessellation. The code is given below

```
%% EXAMPLE: Mesh class
%  generate sdf mesh domain
c   = sCircle(1);
r   = sRectangle(1);
sdf = r.rotate(pi/4)-c;

% sdf conversion to mesh
msh = Mesh(sdf,'NElem',150);
msh = msh.generate();
```

In Figure 6.4, we see the evolution of the Voronoi cells that produce the `PolyN`-type mesh. Observe that after a few iterations of Lloyd's algorithm, the centroids are distributed homogeneously over the compact domain Ω (as shown by the color distribution).

B: Mesh from common file formats An alternative option is to use the mesh generation tools provided by the *Partial Differential Toolbox* in Matlab. Such

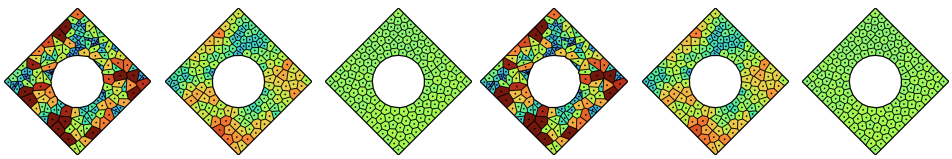


Figure 6.4. Example of mesh generation in *Sorotoki*. The figure shows the evolution of an unstructured polygonal mesh based on Lloyd's algorithm. The colors relate to the relative element size with respect to the mean element size, given by $\in [0, 10]$ mm^2. Notice that only after a few iterations, the centers of the Voronoi cells become homogeneously distributed within the meshing domain Ω.

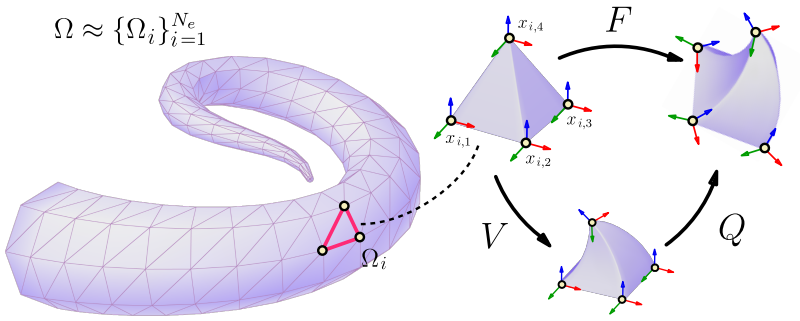


Figure 6.5. Illustration of the Finite Element Method (FEM), where a solid geometry Ω is subdivided into N_e finite elements Ω_e. Each element has $\dim(\boldsymbol{x}_e)$ DOFs, which allows the computation of the deformation gradient $\boldsymbol{F}$. The deformation gradient can be decomposed into $\boldsymbol{F} = \boldsymbol{Q}\boldsymbol{V}$, an isochoric deformation part $\boldsymbol{V}$ and rigid-body rotation part $\boldsymbol{Q}$.

function is also included in the `Mesh.generate()` routine. SDFs can also be used in this process, although an intermediate step is required. For two-dimensional domains, SDFs are first converted into binary images and then the image boundary detection is used to convert them to either a linear mesh (`Tri3`) or a quadratic mesh (`Tri6`). Direct input of black-and-white `.jpg` or `.png` images is also supported. For three-dimensional domains, SDF functions are converted to an `.stl` file using the Marching Cube algorithm [**?**] and then provided to the Matlab PDE toolbox to generate the tessellation. Importing `.stl` or `.obj` files directly is also possible. Meshes can also be export

6.5.3 Finite element modeling (`Fem`)

Following the mesh generation process, *Sorotoki* offers a nonlinear finite element solver for both quasi-static and fully dynamic simulations. An illustration of the FEM approach is given in Figure 6.5. FEM-based tools are crucial when describing large deformations in soft robots, which also accounts for hyperelastic materials and geometric nonlinearities. The FEM package is provided in a class called `Fem.m` and can be instantiated using `fem = Fem(Mesh)`. This class serves two main purposes: (*i*) to solve static or dynamic continuum problems with high accuracy, and (*ii*) to solve gradient-based optimization problems, also known as inverse design problems. It is important to note that, unlike *SOFA*, the focus of *Sorotoki* is on high-detail simulations rather than real-time implementation for control. The presented FEM simulation models are not intended for real-time applications, but rather for system identification and analysis.

A: High-detail finite element model. The nonlinear dynamics of the finite element model in *Sorotoki*, similar to *SOFA* and *Gibbon*, can be described by the general Newton-Euler equation of motion:

$$\boldsymbol{M}\ddot{\boldsymbol{x}} + \boldsymbol{f}_{\text{mat}}(\boldsymbol{x}, \dot{\boldsymbol{x}}) + \boldsymbol{f}_{\text{g}} = \boldsymbol{f}_{\text{u}}(\boldsymbol{x}, \boldsymbol{u}, t) + \boldsymbol{f}_{\Omega_{\text{env}}}(\boldsymbol{x}, \dot{\boldsymbol{x}}, t), \tag{6.3}$$

where $\boldsymbol{x}$, $\dot{\boldsymbol{x}}$, and $\ddot{\boldsymbol{x}}$ are the global nodal displacement, velocities and accelerations of the mesh tesselation, respectively; $\boldsymbol{M}$ the constant generalized mass matrix, $\boldsymbol{f}_{\text{mat}}$ the internal soft material forces, $\boldsymbol{f}_{\text{g}}$ the constant gravitational forces, $\boldsymbol{f}_{u}$ a user-defined input, and $\boldsymbol{f}_{\Omega_{\text{env}}}$ the normal reaction forces and tangent friction forces imposed by the dynamic contact with a (possibly time-dependent) environment Ω_{env}. The environment Ω_{env} can be described using the SDF functionality (see Section 6.5.1) using the syntax `fem.addContact(sdf)`. A broad collection of generalized external inputs can be added using: `fem.addLoad`, `fem.addDisplace`, `fem.addGravity`, and `fem.addTendon`. Pressure inputs can be added using the command `fem.addPressure`.

Without loss of generality, the material force can be decomposed into a position-dependent and velocity-dependent part: $\boldsymbol{f}_{\text{mat}}(\boldsymbol{x}, \dot{\boldsymbol{x}}) = \boldsymbol{f}_{\text{e}}(\boldsymbol{x}) + \boldsymbol{f}_{\text{d}}(\dot{\boldsymbol{x}})$, *i.e.*, an elastic and dissipation contribution, respectively. We assume that the dissipation is given by $\boldsymbol{f}_{\text{d}} = \zeta \mathbf{M}\dot{\boldsymbol{x}}$ with damping coefficient $\zeta > 0$. Materials can be assigned using `fem.addMaterial`. Note that the conservative elastic material forces $\boldsymbol{f}_{\text{e}}$ require more involved computation. Since this computation is not straightforward, we briefly explain the derivation of the nonlinear hyper-elastic material forces in (6.3), which follows standard nonlinear finite element procedures [95, 117, 194].

Intermezzo 6.1 (Deformation gradient)**.** A fundamental measure of deformation in continuum mechanics is the deformation gradient, denoted by $\mathbf{F}$. The deformation gradient characterizes the local deformation for a neighborhood of the continuum body Ω. Since a subvolume of the continuum body cannot be reduced to a point, it follows that $\det(\mathbf{F}) = J > 0$ and $\mathbf{F}^{-1}$ exists. The term J is called the relative volume change and it is equal to 1 for isochoric deformations, such as rigid body deformations. Given these properties, the deformation gradient can then be factorized into $\mathbf{F} = \mathbf{QV}$, where $\mathbf{V} \succ 0$ is the right-handed stretch tensor and $\mathbf{Q} \in \text{SO}(3)$ is a rotation matrix belonging to the special orthogonal group [95, 117, 194]. For convenience, we summarize the derived quantities of $\mathbf{F}$ in Table 6.2 that will be used throughout this section.

Intermezzo 6.2 (Derivation of hyperelastic forces)**.** Let Ω_i denote the subspace spanned by the i-th element of the finite element mesh, and let $\boldsymbol{x}_i$ denote its

Table 6.2. Table of deformations measures relevant for continuum mechanics problem. All measures can be related to the first-order deformation tensor $\boldsymbol{F}$, following the works [95, 117, 194].

Deformation measure	Derivation
Relative volume change	$J = \det(\boldsymbol{F})$
Polar decomposition	$\boldsymbol{F} = \boldsymbol{Q}\boldsymbol{V}$
Right Cauchy-Green tensor	$\boldsymbol{C} = \boldsymbol{F}^\top \boldsymbol{F}$
First strain invariant	$I_1 = \text{trace}(\boldsymbol{C})$,
Second strain invariant	$I_2 = \frac{1}{2}\left[\text{trace}(\boldsymbol{C})^2 - \text{trace}(\boldsymbol{C}^2)\right]$
First strain invariant	$I_3 = \det(\boldsymbol{C})$

nodal displacement vector. The elasticity of the constitutive soft material can be described by a strain-energy density function $\Psi : \boldsymbol{F} \rightarrow \mathbb{R}_{\geq 0}$. A comprehensive discussion on common constitutive models for Ψ will be provided later in Section **??**. The elastic potential energy of the continuum body is given by $\mathcal{U}_e = \int_\Omega \Psi(\cdot)\, dV$, and the conservative hyper-elastic force contribution can be computed as $\boldsymbol{f}_e := \nabla_{\boldsymbol{x}} \mathcal{U}_e$. This contribution can be approximated using piecewise finite element interpolation and integrated using the Gauss quadrature rule [117] as follows:

$$\begin{aligned} \boldsymbol{f}_{\text{e}}(\boldsymbol{x}) &= \sum_{i=1}^{N_e} \frac{d}{d\boldsymbol{x}_i} \left\{ \int_{\Omega_i} \Psi(\boldsymbol{F}(\boldsymbol{x}_i, s))\, ds \right\}, \\ &\approx \sum_{i=1}^{N_e} \sum_{j=1}^{N_w} w_j \underbrace{\frac{\partial \Psi}{\partial \boldsymbol{F}}(\boldsymbol{F}(\boldsymbol{x}_i, s_j))}_{\text{PK1}} \frac{\partial \boldsymbol{F}}{\partial \boldsymbol{x}_i}(\boldsymbol{x}_i, s_j) \end{aligned} \tag{6.4}$$

where the Gauss weights are denoted by $w_j > 0$, and the number of finite elements and Gauss samples are represented by N_e and N_w, respectively.

The term $\frac{\partial \Psi}{\partial \boldsymbol{F}}$ is also referred to as the first Piolla-Kirchhoff (PK1) stress tensor, which can be represented in closed-form for many constitutive models. The term $\frac{\partial \boldsymbol{F}}{\partial \boldsymbol{x}_e}$ denotes the deformation Jacobian, which can also be given in closed-form but depends on the choice of element type. In addition to the first Piolla stress tensor, we also introduce the Cauchy stress tensor (i.e., true stress) $\boldsymbol{\sigma} := J^{-1} \frac{\partial \Psi}{\partial \boldsymbol{F}} \boldsymbol{F}^\top$, which is a symmetric second-order tensor whose components represent the true stress. It should be noted that these tensor calculations are highly nonlinear, making their computation the most time-consuming aspect of the finite element assembly. To enhance computational efficiency, the toolkit employs `.mex` executable code that is generated during installation (`Matlab Coder` toolkit is required).

Table 6.3. Table of material models included in *Sorotoki*, including the Neo-Hookean model (NH), Mooney-Rivlin model (MR), and Yeoh (YH) model.

Material model	Parameters	Energy-density potential Ψ
Neo-Hookean (NH)	(μ)	$\Psi_{\mathrm{NH}} := \frac{\mu}{2}(I_1 - 3)$
Mooney-Rivlin (MR)	(c_1, c_2)	$\Psi_{\mathrm{MR}} := \sum_{i=1}^{2} c_i (I_i - 3)$
Yeoh (YH)	(c_1, c_2, c_3)	$\Psi_{\mathrm{YH}} := \sum_{i=1}^{3} c_i (I_1 - 3)^i$

B: Hyperelastic models and soft material presets. An important aspect of soft robotics in general is to accurately describe large nonlinear deformations of inertial continuum bodies in motion. Yet, due to these large deformations, many classical Hookean elasticity models may not be accurate for elastomer materials.

To address this, *Sorotoki* provides a library of hyper-elastic constitutive material models: Neo-Hookean (NH), Mooney-Rivlin (MR), and Yeoh model (YH). The strain energy densities for these models are derived based on the strain invariants I_1, I_2, and I_3 provided in Table 6.2 and are shown in Table 6.3. The material models presented in Table 6.3 are implemented in *Sorotoki* under the class `Material`, but have specific constructors tailored towards each material, `NeoHookeanMaterial`, `MooneyMaterial`, and `YeohMaterial`. Regarding their parameters, the work of Marechal et al. [**?**] provides an open-source database that includes a broad collection of soft materials commonly used in soft robotics, gathered through uniaxial material tests. Based on their dataset and relevant other literature [82, 117, 194, 246], *Sorotoki* offers some preset material models of soft materials commonly used in soft robotics, such as the Ecoflex30/50 series, Dragonskin10/30 series, NinjaFlex, and Formlabs Elastic50A/80A material. These material classes also include the physical data for density, viscosity, and tangential contact friction. Following (6.4), the first Piola-Kirchhoff (PK1) stress tensor is evaluated analytically using the function call `P = Material.PiollaStress(F)`.

C: Finite element solvers and (nonlinear) modal analysis To solve the structural forward dynamics of the system (6.3), the toolkit uses an implicit Newmark-β solver [**?**], which is briefly outlined in Appendix C.1. Implicit solvers offer improved stability compared to explicit methods, such as the Runge-Kutta solver (`ode45`), particularly when larger time steps are employed. However, the cost of larger time steps is a decreased numerical precision. Alternatively, for quasi-static problems when $\ddot{\boldsymbol{x}} = \dot{\boldsymbol{x}} = \mathbf{0}_n$, we aim to seek the solutions to the static force equilibrium $\boldsymbol{r}(\boldsymbol{x}) = \mathbf{0}_n$, where the force residual vector is given by $\boldsymbol{r} := -\boldsymbol{f}_{\mathrm{e}} + \boldsymbol{f}_{\mathrm{g}} + \boldsymbol{f}_{\mathrm{u}} + \boldsymbol{f}_{\Omega_{\mathrm{env}}}$. For quasi-static problems with $\ddot{\boldsymbol{x}} = \dot{\boldsymbol{x}} = \mathbf{0}_n$,

the aim is to find the solution to the static force equilibrium $\boldsymbol{r}(\boldsymbol{x}) = \mathbf{0}_n$, where $\boldsymbol{r} := -\boldsymbol{f}_{\mathrm{e}} + \boldsymbol{f}_{\mathrm{g}} + \boldsymbol{f}_{\mathrm{u}} + \boldsymbol{f}_{\Omega_{\mathrm{env}}}$ is the force residual vector. The nonlinear equality for nodal displacements $\boldsymbol{x}$ is solved using an iterative Newton-Raphson solver. To call these solvers, dynamic simulations are executed with `fem.simulate()` and quasi-static simulations with `fem.solve()`. Upon completion of a simulation, all displacements, velocities, forces, and stress information are stored in the `fem.Log` data structure. This log file can be accessed for data analysis or during simulation to facilitate state feedback control.

Alternatively, we can explore nonlinear modal analysis at any quasi-static equilibrium configuration $\boldsymbol{x}^* \in \mathcal{X}$ of the system (6.3). Let $\boldsymbol{K}_T := \nabla_{\boldsymbol{x}} \boldsymbol{f}_{\mathrm{e}}$ be the Jacobian matrix of the (nonlinear) elastic potential forces, also referred to as the tangent stiffness. Then, the local eigenvalue problem for the linearized FEM model around the point $\boldsymbol{x}^*$ is given by

$$[\boldsymbol{K}_T(\boldsymbol{x}^*) - \lambda_i \boldsymbol{M}]\, \boldsymbol{\theta}_i = \mathbf{0}_n, \tag{6.5}$$

where λ_i is a real scalar eigenvalue and $\boldsymbol{\theta}_i$ is its corresponding eigenmode. The dynamic analysis is implemented in Sorotoki using `fem = fem.analysis(x)`, which stores the necessary data in `fem.Log`. It is important to note that, unlike linear finite element models, the set of eigenmodes $\boldsymbol{\theta}_i$ obtained from the eigenvalue decomposition in (6.5) is highly dependent on the linearization point $\boldsymbol{x}^*$ and may thus not be unique for all $\boldsymbol{x}^* \in \mathcal{X}$.

Example 6.3 (Nonlinear buckling analysis via decomposition)**.** An excellent case study of the eigenvalue problem in nonlinear elasticity systems is the buckling behavior of patterned elastomer metamaterials, as studied by Bertoldi et al. [**?**] and later by Overvelde et al. [**?**]. In their studies, an elastomer specimen with a periodic circular porous structure was subjected to uniaxial compression. The specimen displayed an inward buckling phenomenon at a critical loading point, resulting in the specimen exhibiting a negative Poisson ratio, *i.e.*, auxetic behavior. To be specific, the structure undergoes a so-called "bifurcation" where solutions switch stability or new solutions arise for a critical parameter value. In this case, the bifurcation parameter is the compression ratio ε.

In accordance with [**?**], a square elastomer specimen with circular holes was modeled using a Neo-Hookean material model, with Young's modulus $E = 19$ (kPa) and Poisson ratio $\nu = 0.45$. As reported in [**?**], the critical buckling point was observed to occur at approximately $\varepsilon = -12.5\%$ uniaxial compression.

A numerical solution is obtained through quasi-static analysis using the function `fem.solve`. To model compression, a displacement load was added using

`fem.addDisplace('Top')`. The resulting equilibrium configuration was then utilized in the eigenvalue problem via the function `fem.analysis(x)`. The eigenmodes for the zero-stress and $\varepsilon = -12.5\%$ compression cases are illustrated in Figure 6.6. It is worth noting that the first three eigenmodes of the elastomer structure at $\varepsilon = 0\%$ compression exhibit no buckling modes. Conversely, the first eigenmode $\boldsymbol{\theta}_1$ at $\varepsilon = -12.5\%$ compression displays a buckling mode, wherein the collapse of the holes is periodically oriented either vertically or horizontally. This buckling mode is in accordance with the experiments from [**?**] and [**?**]. The supplementary code is provided below:

```
%% EXAMPLE: Fem class
msh = Mesh(sdf, 'NElem', 5e3);
fem = Fem(msh, 'TimeStep', 1/60);

% assign material
E = 1.0; Nu = 0.45;
fem.Material = NeoHookeanMaterial(E, Nu);

% add displacement and forces
fem = fem.addConstraint('Bottom', [1,1]);
fem = fem.addConstraint('Top', [1,0]);
fem = fem.addDisplace('Top', [0,-0.125 * W]);

%  quasi-solve and eigen analysis
fem = fem.solve();
fem = fem.analysis( fem.Log.x(:,end) );
```

Example 6.4. Locomotion dynamics of soft crawling robot To demonstrate a dynamic finite element method (FEM) simulation that incorporates contact, we will utilize *Sorotoki* to model the locomotion of a multi-gait soft robot crawler inspired by the work of Shepard [185]. The study by Shepard et al. [185] presents a soft robot system that consists of five pressure chambers - four for each leg and one for the spine. The pressure chambers are actuated in a sequential manner to produce an undulating motion. The work of Shepard et al. [185] demonstrates that complex locomotion can be achieved through the use of open-loop controllers and the dynamic interaction between the soft robot and its environment.

To simplify the model, we assume general plane motion. The geometry of the soft crawler's cross-section is first provided to `Mesh.m` to generate a triangular mesh. Then, a finite element method (FEM) model is generated, with the material model `fem.Material = Ecoflex0030`. To model the environment, the function `fem = fem.addContact(sLine)` is utilized, which simply creates an unbounded horizontal line. In accordance with [185], a harmonic excitation is applied to each

chamber, as expressed by the following equation:

$$u_i = A\text{sat}\left[\sin(\omega t - \phi_i)\right],$$

where the index $i \in 1, 2, 3$ represents the front, middle, and back pressure chambers embedded in the soft body. The excitation signal parameters are set as follows: $A = 45$ (kPa), $\omega = 5\pi$ (rad), and $\phi_i = \frac{\pi}{3} \cdot (i - 1)$ (rad). The saturation function is defined as $\text{sat}(x) = 0$ for $x < 0$, and $\text{sat}(x) = x$ for $x \geq 0$. Gravitational acceleration is added, and the dynamic simulation solver is invoked using `fem.simulate`. Figure 6.7 presents a comparison between the soft robot described in [185] and the dynamic simulation performed by *Sorotoki*.

The results of the simulation performed using *Sorotoki* show a morphological behavior that is consistent with the experimental recordings. Figure 6.8 depicts the trajectory of the center of mass (CoM) of the soft robot during the undulating locomotion. Note that an identical stair-like evolution of the CoM is also observed in the work of Shepard et al. [185]. The code written in *Sorotoki* is provided below, and its pipeline is shown in Figure **??**.

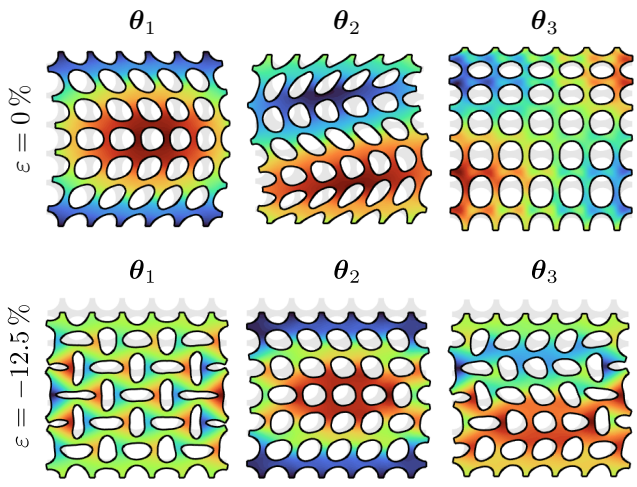


Figure 6.6. Nonlinear buckling mode analysis of periodic circular porous elastic structure inspired by [**?** **?**]. The horizontal displacements are indicated by ∈ $[-5, 5]$ mm. (top) The first three eigenmodes of the elastomer structure for $\varepsilon = 0\%$ compression, no buckling modes appear. (bottom) The first three eigenmodes for $\varepsilon = 12.5\%$ compression. Notice that the first mode $\boldsymbol{\theta}_1$ is a buckling mode where the collapes holes orient periodically either vertically or horizontally.

```
%% EXAMPLE: Fem class
msh = Mesh('MultiGait.png', 'MeshSize', 1.0, ...
           'BdBox', [0, 150, 0, 12]);

% -- mesh to fem
fem = Fem(msh,'TimeStep',1/500);
fem.Material = Ecoflex50();

% -- add inextensible bottom layer
E1  = fem.FindElements([0 150 0 2]);
fem = fem.addMaterialModifier(E1, 5);

% -- add forces and constraints
Y  = 'BoxSelect';
C1 = fem.FindEdges(Y,[0   50 0 15]);
C2 = fem.FindEdges(Y,[50  100 0 15]);
C3 = fem.FindEdges(Y,[100 150 0 15]);

fem = fem.addPressure(C1, @(x) Pressure(x,1));
fem = fem.addPressure(C2, @(x) Pressure(x,2));
fem = fem.addPressure(C3, @(x) Pressure(x,3));

fem = fem.addGravity();
fem = fem.addContact(sLine());

% -- solve dynamics
fem = fem.simulate();

% -- open-loop controller
function y = Pressure(fem, k)
    phi = (4/11) * pi;
    y = clamp(sin(5*pi*fem.Log.t - phi), 0, Inf);
end
```

Gradient-based computational (inverse) design Besides modeling, the field of computational inverse design can also benefit from the use of FEM models. Building up the `Fem` class, the objective is to find a topolgical structure of a continuum system based on a desired deformations or compliance. One widely adopted method is the Solid Isotropic Material with Penalization (SIMP) approach, which is a commonly used material interpolation technique in topology optimization [**?**]. In the SIMP method, each finite element $e \in \{1, 2, ..., N_e\}$ is assigned a continuous density variable $\rho_e \in (0, 1]$, which serves as an indicator of the material distribution within the mesh. If $\rho_e = 1$, the element is considered solid, while if $\rho_e = 0$, the element is considered void. This assignment of density variables enables the

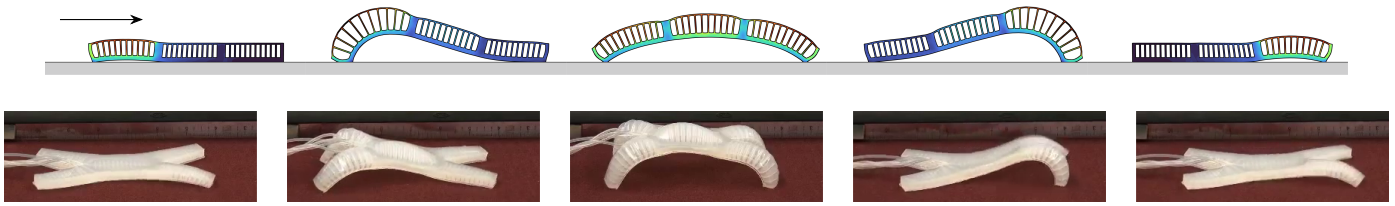


Figure 6.7. (top) Experiment of soft multi-gait crawling soft robot developed by Shepard et al [185] performing an undulating forward motion by periodic pressurizarion of its internal pressure chambers, back legs → middle → front legs. The soft robot is made from an elastomer material with strain-inhibit layer at the bottom to enhance bending. The Von Mises stresses are shown as ∈ [0, 10] MPa. (bottom) Simulation recreation of the experiments performed by Shepard et al [185] using the Finite Element solver in the *Sorotoki* toolkit. Images sourced from public resources, with intellectual property belonging to cited authors, sources, and publishers.

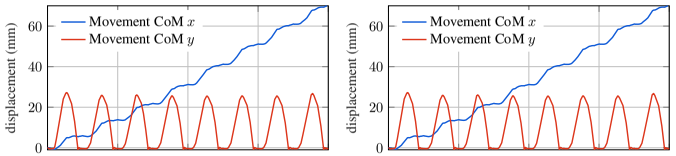


Figure 6.8. Numerical simulation of the center of mass displacement and gait for undulating soft crawler. The horizontal displacement is given by (—), and the vertical displacement as (—).

modification of the strain energy density in (6.4):

$$\tilde{\Psi}_e = [\varepsilon + (1-\varepsilon)\rho_e{}^p]\,\Psi, \tag{6.6}$$

where $0 < \varepsilon \ll 1$ a lower bound on the densities, and $p > 1$ a penalty factor for penalizing intermediate densities during the optimization process. By collecting the density values $\boldsymbol{\rho} = \text{vec}\{\rho_1, \rho_2, ..., \rho_{N_e}\}$, the inverse design problem can be formulated in terms of two unknowns: the displacement field $\boldsymbol{x}$ and the density field $\boldsymbol{\rho}$. Consequently, the computational design problem for general soft material structures can be expressed as a nonlinear topology optimization problem of the following form:

$$\begin{aligned} \underset{\boldsymbol{\rho}}{\text{minimize}} \quad & \Phi = -\beta_1\, \boldsymbol{L}^\top \boldsymbol{x} \;+\; \beta_2 \boldsymbol{f}_{\text{e}}^\top \boldsymbol{f}_{\text{u}} \\ \text{subject to} \quad & \boldsymbol{r}(\boldsymbol{x}, \boldsymbol{\rho}) = 0, \\ & \boldsymbol{v}^\top \boldsymbol{\rho} \leq v^\star, \\ & \boldsymbol{\rho} \in \mathcal{P}, \end{aligned} \tag{6.7}$$

where $\boldsymbol{L}$ a sparse unit-vector composed of nonzero entries for the degrees-of-freedom corresponding to the desired morphology of the soft robot, $\boldsymbol{v}$ the element volumes, $v*$ the desired volume infill, $\mathcal{P} = \{\boldsymbol{\rho} \in \mathbb{R}^{n_e} \mid 0 < \rho_i \leq 1\}$ admissable set for the design variables, and β_1 and β_2 are positive scalars that can be adjusted to vary the optimization problem, with $\beta_1 \ll \beta_2$ resulting in compliance minimization and $\beta_1 \gg \beta_2$ leading to a compliant mechanism. To solve the optimization problem in Equation (6.7), we utilize the Method of Mixed Asymptotes (MMA) proposed by Svanberg [**?** **?**]. Earlier work on this computational design approach was presented in [**?**], and the reader is referred to this work for the analytic gradients required for the MMA solver.

The optimization routine in the *Sorotoki* framework is incorporated into the `Fem` class and can be invoked by utilizing the command `fem.optimize('type')`, where `'type'` represents the optimization problem at hand. For minimizing compliance, the cost function is inherent to the finite element method (FEM) problem. However, when dealing with compliant mechanisms, it is necessary to specify the selection vector $\boldsymbol{L}$, which can be defined using the `fem.addOutput(id)` command. The value of `id` represents the nodal indices of interest, which can be identified using the `fem.Mesh.findNode` functionality.

6.5.4 Reduced-order soft beam models (`Shapes`)

While the finite element method (FEM) is known for producing reliable and highly accurate results, its high-dimensional state can make it computationally slow,

making direct applications for closed-loop control challenging. To address this issue, the Sorotoki toolkit offers reduced-order models based on Cosserat beam theory [**? ? ? ? ?**]. In Cosserat beam theory, deformable solids are modeled as elastic strings that are governed by finite strain theory. This formulation can be applied to the dynamic modeling of slender soft robots as one-dimensional spatial curves passing through the geometric center of the deformable soft body. As shown in Figure 6.9, a (slender) soft robot can be described using geometric Cosserat beam models, representing it as a parameterized curve on the group of rigid-body transformations SE(3):

$$\boldsymbol{g} : [0, L] \times [0, T] \rightarrow \mathrm{SE}(3), \tag{6.8}$$

where SE(3) composed of an orthogonal rotation matrix and a translation vector.

The objective of this approach, similar to the finite element method, is to solve a dynamic system in a continuous manner, often through projecting the problem onto a finite-dimensional subspace. To address the infinite dimensionality of the curve $\mathbf{g}$ and make the continuum kinematics computationally tractable, various methods have been proposed, including elemental discretization [**? ?**] (which is analogous to Section 6.5.3). A widely adopted alternative is modal approximation [45**? ?**]. The concept of modal decomposition for describing the kinematics of continuum robots dates back to the early 1990s [45**?**], and some modal representations (*e.g.*, first-order fourier series) even provide closed-form solutions to the inverse kinematics.

The method for constructing soft beam models in the *Sorotoki* toolkit is expressed using the syntax `shp = Shapes(pod,dof)`. In this expression, `pod` is a modal interpolation matrix that is derived from a modal basis selected by the user, and `dof` is a vector of six unsigned integers (`uint8`) that couples the beam degrees of freedom, including extension, bending, torsion, and shear, to their modal representation. The `Shapes` class serves two primary purposes: (*i*) to enable fast forward dynamic simulation of soft robots and (*ii*) to simplify the design of model-based controllers for both online and offline environments. Compared to the FEM model in (6.3), the soft beam models implemented in *Sorotoki* typically have a significantly lower dimensional representation, resulting in improved computational speed and, in some cases, real-time performance. This enables model-based controllers on real platforms.

A: Computationally-efficient soft beam models. Following the geometric Cosserat beam frameworks [169**? ?**], the reduced nonlinear dynamics of a soft beam model, fixed to a non-inertial base, can be represented using a Lagrangian

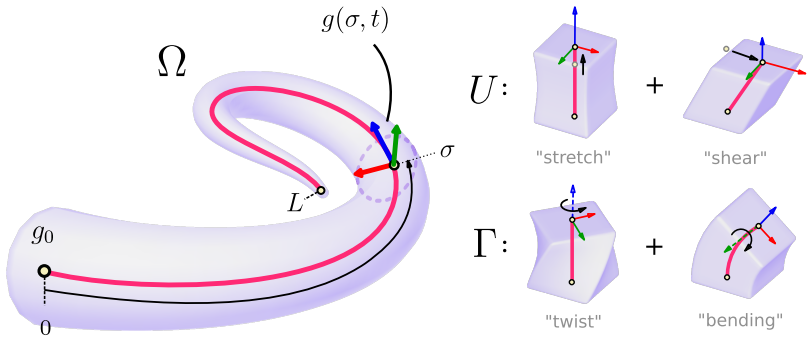


Figure 6.9. Illustration of the soft beam model using geometric Cosserat beam theory, where the backbone curve is $\boldsymbol{g}(\sigma, t) \in \mathrm{SE}(3)$ shown as (—). The geometric strain vector $\boldsymbol{\xi} := \mathrm{vec}\{\boldsymbol{\Gamma}, \boldsymbol{U}\}$ a vector of size 6 consisting of stretch-shear strains $\boldsymbol{U}$ and twist-bending strains $\boldsymbol{\Gamma}$.

formulation:

$$\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{f}_{\mathrm{mat}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{f}_{\mathrm{g}}(\boldsymbol{q}) = \boldsymbol{f}_{\Omega_{\mathrm{env}}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{\tau}(\boldsymbol{q}, \boldsymbol{u}), \quad (6.9)$$

where $\boldsymbol{q}$, $\dot{\boldsymbol{q}}$, and $\ddot{\boldsymbol{q}}$ represent the modal coefficients, velocity, and acceleration, respectively; $\boldsymbol{M}$ denotes a state-dependent generalized inertia matrix, and $\boldsymbol{C}$ denotes the Coriolis matrix. The material forces are expressed as $\boldsymbol{f}_{\mathrm{mat}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) = \boldsymbol{K}(\boldsymbol{q})\boldsymbol{q} + \boldsymbol{R}\dot{\boldsymbol{q}}$, where $\boldsymbol{K}$ is a generalized stiffness matrix and $\boldsymbol{R}$ is a generalized damping matrix. The environmental forces are represented by the vector $\boldsymbol{f}_{\Omega_{\mathrm{env}}}$, and the generalized input is given by $\boldsymbol{\tau} := \boldsymbol{G}\boldsymbol{u}$, where $\boldsymbol{G}(\boldsymbol{q})$ is the input mapping. As in Section 6.5.3, material and contact models can be assigned using comparable syntax, `Shapes.addMaterial` and `Shapes.addContact`, respectively. The intrinsic length of a curve can be altered by utilizing the function `Shapes.setLength`, while its cross-sectional geometry can be modified through the function `Shapes.setGeometry(sdf)`, which accepts a two-dimenional SDF function that may be arbitrarily complex.

Due to the complexity of deriving the forward kinematics and dynamics in the Cosserat model, the following subsections provide a clear summary of the finite-dimensional basis representation and its relationship to reduced kinematics. The system matrices, on the other hand, are notoriously lengthy expressions and thus omitted in this work. The reader is referred to [**?**] for a full derivation of model (6.9).

B: Finite dimensional projection. To start, our aim is to obtain a finite-dimensional approximation of the local geometric strain vector, denoted as $\boldsymbol{\xi} := (\boldsymbol{g}^{-1}\frac{\partial \boldsymbol{g}}{\partial \sigma})^{\wedge} := (\boldsymbol{\Gamma}^{\top}, \boldsymbol{U}^{\top})^{\top}$, where $\sigma \in [0, L]$ is a spatial coordinate and $(\cdot)^{\wedge} : \mathrm{se}(3) \to$

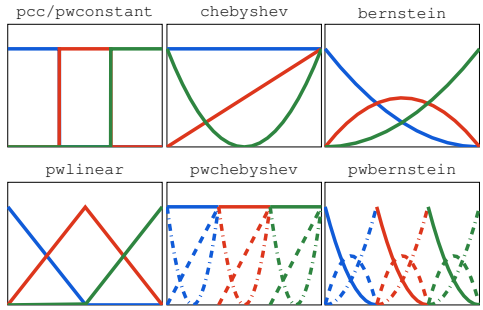


Figure 6.10. The library of modal basis functions is implemented in the Sorotoki toolkit, with a modal ordering of {—, —, —}. These function bases include Piecewise Constant Curvature (PCC), Piecewise Linear (PWL), full and piecewise Chebyshev, and full and piecewise Bernstein polynomials.

$\mathbb{R}^6$ (see [**?**]). Here, Γ_i and U_i are the torsion-curvature and elongation-shear curve parameters, respectively. To achieve this, we employ a Ritz-Galerkin modal discretization approach following the work of Boyer et al. [**?**]. This approach assumes that the strain can be accurately represented through a finite series of orthonormal basis functions:

$$\begin{aligned}[\xi_i]_{\boldsymbol{\theta}_i}(\sigma, \boldsymbol{q}_i) &= \sum_{j=1}^{k_i} \theta_{i,j}(\sigma) q_{i,j} + \xi_i^\circ(\sigma), \\ &= \underbrace{\begin{bmatrix}\theta_{i,1}(\sigma) & \dots & \theta_{i,k_i}(\sigma)\end{bmatrix}}_{\boldsymbol{\theta}_i^\top(\sigma)} \boldsymbol{q}_i + \xi_i^\circ(\sigma) \end{aligned} \tag{6.10}$$

where $\boldsymbol{\theta}_i$ is the modal approximation vector related to the i-th strain component, $\boldsymbol{q}_i$ is its corresponding modal coefficient, and $[\cdot]_{\boldsymbol{\theta}}$ denotes the subspace projection operator. By collecting all terms $q_{i,j}$ and $\theta_{i,j}$, we compactly express the finite-dimensional approximation as an affine operation:

$$[\boldsymbol{\xi}]_{\boldsymbol{\Theta}}(\sigma, \boldsymbol{q}) = \boldsymbol{\Theta}^\top(\sigma)\boldsymbol{q} + \boldsymbol{\xi}^\circ(\sigma) \tag{6.11}$$

where $\boldsymbol{\Theta} := \text{blkdiag}\{\boldsymbol{\theta}_1, \dots, \boldsymbol{\theta}_6\}$ is referred to as the *"modal approximation matrix"*, and $\boldsymbol{q} := \text{vec}\{\boldsymbol{q}_1, \dots, \boldsymbol{q}_6\}$ is the generalized coordinate vector of the global soft beam model in (6.9). Note that a geometric strain entry may be constrained and therefore not contribute to the overall continuum dynamics, thus $\boldsymbol{\theta}_i, \boldsymbol{q}_i \in \emptyset$ without loss of generality. The choice of basis plays a crucial role and often relies on *ad-hoc* approaches. It is therefore critical to choose an appropriate basis for optimal performance of the soft robot model.

C: Library of modal strain bases. In *Sorotoki*, the general constructor for creating a modal basis is defined as `pod = Basis(N,M,'type')`, where `N` represents

the number of samples (i.e., the level of discretization of the spatial curve), `M` is the degree of the basis, and `'type'` is a character input that specifies the basis type.

The literature presents various types of modal bases, with the Piecewise Constant Curvature (PCC) approach being the most commonly used [64**?**]. The Piecewise Constant Curvature approach is suitable for certain conditions, for example, when homogeneous bending moment and homogeneous material properties are considered. However, it lacks the ability to ensure the continuity of the strain field at the boundaries between sections, resulting in jumps in the strain profile. As a result, researchers have been exploring alternative representations that more effectively preserve the continuity conditions of the deformable continuums. Examples of alternative representations of bases include piecewise linear [**?**], affine curvature [**? ?**], Fourier cosine/sine series [45**?**], Legendre or Chebyshev [**? ?**], and actuation load bases [**?**]. The *Sorotoki* package offers access to a library of anonymous functions, facilitating the utilization of a range of basis functions. As an illustration, a collection of basis functions is displayed in Figure 6.13.

Remark 6.2 (On the modal order) Generally, finding a suitable reduction basis and reduction order can be a challenging task. The general assumption is that if the basis belongs to a regular function space (*i.e.*, Sobolev space) and the modal index k_i goes to infinity, the strain approximation converges (uniformly) to the exact solution on the interval $[0, L]$. However, as increasing the modal order enhances precision, it also greatly impacts computational performance. Thus, finding a balance between accuracy and computational speed is of utmost importance for the successful implementation of soft robotic models, often requiring an *ad-hoc* approach.

C: Data-driven strain basis from offline FEM simulations. To address the challenges of improving efficacy in the modal reduction of soft beam models, we propose a novel approach that merges the finite element method and soft beam modeling. This approach involves extracting geometric modal information from FEM simulation data to construct a low-dimensional strain basis, which we refer to as the Data-driven Variable Strain (DVS) basis. The DVS basis is similar in concept to the snapshot method presented by the SOFA toolkit [51, 63, 82], but adapted for use with one-dimensional curves. It takes into account the underlying geometric features of the soft robot and represents them in a minimal subspace representation. This approach leads to a substantial reduction in the number of states while still maintaining high accuracy in deformations, high computational efficiency, and providing a clear structure for passive and active joints. The derivation has two steps:

Step 1: Recovery of geometric strain from FEM: The reconstruction of the MIVS basis begins with obtaining geometric strain data from a Finite Element Method (FEM) simulation. The simulation is carried out using either `Fem.simulate` or `Fem.solve`, and the resultant information is stored in the `fem.Log` data structure. Mathematically, the simulation retrieves the states $\boldsymbol{x}^{(i)} := \boldsymbol{x}(t_i)$ at discrete time instances $t_i \in \{0, ..., T\}$, which in turn provides us with nodal position vectors $\boldsymbol{p}$ and the deformation gradient $\boldsymbol{F}$ at any point in the mesh. Using the polar decomposition $\boldsymbol{Q} = \boldsymbol{F}\boldsymbol{V}^{-1} \in \mathrm{SO}(3)$, see Table 6.2, we can retrieve the rigid body transformation of the FEM mesh

$$\boldsymbol{g}_{\mathrm{FEM}}(\boldsymbol{s}, \boldsymbol{x}^{(i)}) = \begin{pmatrix} \boldsymbol{Q}(\boldsymbol{s}, \boldsymbol{x}^{(i)}) & \boldsymbol{p}(\boldsymbol{s}, \boldsymbol{x}^{(i)}) \\ \boldsymbol{0} & 1 \end{pmatrix},$$

where $\boldsymbol{s} \subseteq \Omega$ is a spatial location within the undeformed mesh. It is important to note that if $\boldsymbol{s}$ does not correspond to a nodal location of the mesh, interpolation using elemental shape functions is necessary. Now, let $\bar{\gamma} : [0, L] \to \Omega$ be a unit-speed reference backbone curve that is contained within the mesh domain Ω. Then, we can retrieve $\boldsymbol{g}_{\mathrm{FEM}}(\bar{\gamma}, \boldsymbol{x}^{(i)})$. Subsequently, the geometric strain can be approximated as $\boldsymbol{\xi}_{\mathrm{FEM}} \approx (\boldsymbol{g}_{\mathrm{FEM}})^{-1} \delta \boldsymbol{g}_{\mathrm{FEM}}$. Here, $\delta \boldsymbol{g}_{\mathrm{FEM}}$ represents the spatial derivative of the reference curve w.r.t. σ, which is calculated using the central difference method. It is worth noting that the choice of $\bar{\gamma}$ is free, allowing for the estimation of geometric strain for many complex structures. The full procedure is outlined in Algorithm 3.

Step 2: POD snapshot basis: Next, we employ the "Snapshot Proper Orthogonal Decomposition" (POD) as described in [63, 82]. This data-driven approach determines a suitable orthonormal basis from simulated or experimental data [12]. Let $y_i(\sigma, t) := \xi_{\mathrm{FEM},i}(\sigma, t)$ represent the measurement of the i-th entry of the strain $\boldsymbol{\xi}_{\mathrm{FEM}}$. For each discrete time t_i, the sample is condensed into a column vector $\boldsymbol{y}_i^{(t)} := \mathrm{col}\{y_i(0, t), \ldots, y_i(L, t)\}$ and then stacked into the "snapshot matrix" $\boldsymbol{S}_i = \mathrm{row}\left\{\boldsymbol{y}_i^{(0)}, \ldots, \boldsymbol{y}_i^{(T)}\right\}$ where T is the finite horizon time. The correlation matrix $\boldsymbol{\mathcal{C}}_i = \frac{1}{m}\boldsymbol{S}_i^{\top}\boldsymbol{S}_i$ is then computed with $m = \dim(\boldsymbol{y}_i)$, and the spectral decomposition is performed:

$$\boldsymbol{\mathcal{C}}_i \boldsymbol{V}_i = \boldsymbol{\lambda}_i \boldsymbol{V}_i, \tag{6.12}$$

where $\boldsymbol{V}_i = \mathrm{row}\{\boldsymbol{v}_{i,1}, , ...\boldsymbol{v}_{i,m}\}$ is a basis of eigenvectors and $\boldsymbol{\lambda}_i = \mathrm{diag}\{\lambda_{i,1}, , ..., \lambda_{i,m}\}$ is a diagonal matrix of sorted eigenvalues. By selecting $k_i \leq m$ such that $\lambda_{i,k_i} \leq \delta$, where δ is a desired threshold, we obtain a truncated orthonormal basis $\{\boldsymbol{v}_{i,j}\}_{j=1}^{k_i}$. This process is repeated until the modal interpolation matrix $\boldsymbol{\Theta}$, required for

Algorithm 3: Recover geometric strain field $\boldsymbol{\xi}_{\text{FEM}}$ from offline FEM simulation data.

Input: Nodal displacements $\boldsymbol{x}$, mesh tesselation $\mathcal{T}$, reference curve $\bar{\boldsymbol{\gamma}}$, and sample set $\mathcal{S}$

Output: Geometric strain field $\boldsymbol{\xi}_{\text{FEM}}$ at time t_i

for i = *each spatial sample* $\sigma_i \in \mathcal{S}$ **do**
  get reference position $\bar{\boldsymbol{p}} \leftarrow \bar{\boldsymbol{\gamma}}(\sigma_i)$;
  get element $\mathcal{E} \leftarrow \texttt{inElement}(\bar{\boldsymbol{p}}, \mathcal{T})$;
  if $\mathcal{E} == \emptyset$ **then**
    get edge $\mathcal{E} \leftarrow \texttt{onClosestEdge}(\bar{\boldsymbol{p}}, \mathcal{T})$;
  end
  initialize $\boldsymbol{\Phi}^{(0)} \leftarrow \boldsymbol{I}_3$;
  initialize $\delta\boldsymbol{\gamma}^{(0)} \leftarrow \boldsymbol{0}_3$;
  for j = *each vertex spanned by element* $\mathcal{E}$ **do**
    get nodal displacement $\boldsymbol{X} \leftarrow \texttt{FEM}(\boldsymbol{x}_j)$;
    get deformation gradient $\boldsymbol{Y} \leftarrow \texttt{FEM}(\boldsymbol{x}_j)$;
    $[\boldsymbol{Q}, \boldsymbol{V}] \leftarrow \texttt{PolarDecomposition}(\boldsymbol{Y})$;
    $\alpha \leftarrow \texttt{ElementInterpolation}(\bar{\boldsymbol{p}})$;
    update $\boldsymbol{\Phi}^{(j)} \leftarrow \texttt{AverageSO3}(\boldsymbol{\Phi}^{(j)}, \alpha\boldsymbol{Q})$;
    update $\delta\boldsymbol{\gamma}^{(j)} \leftarrow \delta\boldsymbol{\gamma}^{(j)} + \alpha\boldsymbol{X}$;
  end
  $\boldsymbol{g}^{(i)}_{\text{FEM}} \leftarrow \texttt{SE3}(\boldsymbol{\Phi}^{(j)}, \bar{\boldsymbol{p}} + \delta\boldsymbol{\gamma}^{(j)})$;
end
for i = *each spatial sample* $\sigma_i \in \mathcal{S}$ **do**
  $\delta\boldsymbol{g}^{(i)}_{\text{FEM}} \leftarrow \texttt{CentralDiff}(\boldsymbol{g}^{(i-1)}_{\text{FEM}}, \boldsymbol{g}^{(i+1)}_{\text{FEM}})$;
  assemble strain $\boldsymbol{\xi}^{(i)}_{\text{FEM}} \leftarrow (\boldsymbol{g}^{(i)}_{\text{FEM}})^{-1}\delta\boldsymbol{g}^{(i)}_{\text{FEM}}$;
end

(6.11), is fully obtained. Finally, a Gram–Schmidt orthogonalization procedure is performed to ensure that its columns are mutually orthogonal.

Example 6.5. Data-driven basis from PneuNet simulation To demonstrate the reconstruction of the DVS basis, we consider a soft bending actuation, also known as the PneuNet actuator. A FEM simulation model is constructed where the soft actuator is subjected to a linearly increasing pressure up to 40 kPa and curls around a cylindrical object when pressurized. Figure 6.11 (top) illustrates the true system and the FEM simulation obtained through `fem.simulate`. The FEM class is then

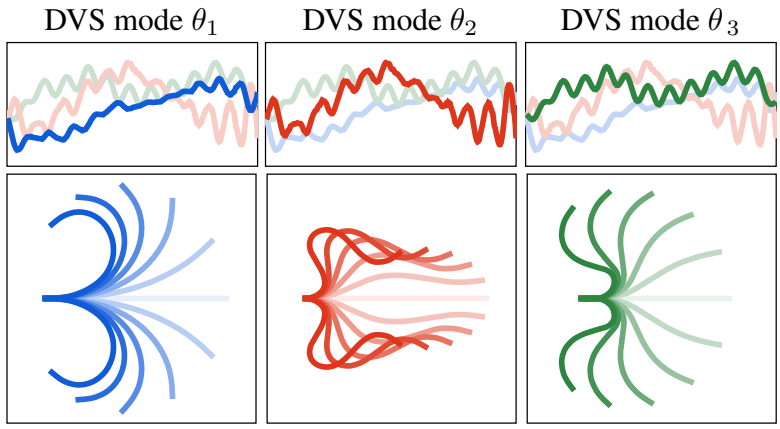


Figure 6.11. Example reconstruction of the Data-driven Variable Strain (DVS) basis for a PneuNet actuator grasping a cylindrical object. (top) The first three modes of the DVS basis related to planar bending, *i.e.*., planar curvature. The ordering is {—, —, —}. (bottom) A comparison between the true physical system, the FEM model, and the soft beam model shown in (—).

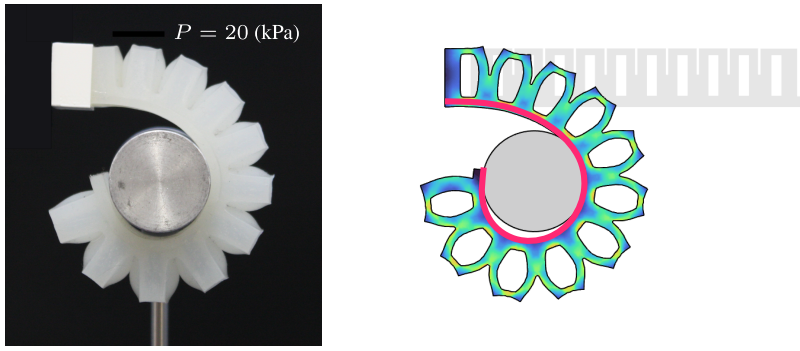


Figure 6.12. Example reconstruction of the Data-driven Variable Strain (DVS) basis for a PneuNet actuator grasping a cylindrical object. A comparison between the true physical system, the FEM model, and the soft beam model shown in (—).

integrated into the `Shapes` class through the syntax `shp = Shapes(fem,dof)`, where `dof = [0,3,0,0,0,0]` indicates the desire to recover the first three curvature bending modes from the `fem` object class. The length and base orientation are specified using `shp.setLength` and `shp.setBase`, respectively. The basis is then reconstructed by calling `shp.reconstruct`. The first three curvature bending modes are displayed in Figure 6.11 (bottom). It can be observed that the

geometrical features of the PneuNet actuator are encoded in the basis, with the 12 embedded pressure chambers represented by the DVS strain basis. The associated code is provided below:

```
%% EXAMPLE: Shapes class (DVS basis)
fem = load('femPneuNet.mat');    % load fem model

dof = [0,M,0,0,0,0];  % pure planar curvature bending
shp = Shapes(fem, dof, 'NNode', 200, 'Length',105);

% generate DVS basis
shp = shp.reference(@(s) [s,0,0].' );
shp = shp.reconstruct();
```

Forward beam kinematics. Once a basis representation $\boldsymbol{\Theta}$ has been selected, the forward kinematics of the continuum body can be efficiently solved using exponential maps for the group SE(3). As such, the backbone curve is be approximated by

$$[\boldsymbol{g}]_{\boldsymbol{\Theta}}(\sigma, \boldsymbol{q}) = \boldsymbol{g}_0 \exp_{\mathrm{SE}(3)} \left[\int_0^{\sigma} [\hat{\boldsymbol{\xi}}]_{\boldsymbol{\Theta}}(s, \boldsymbol{q}) \, ds \right] \tag{6.13}$$

On the other hand, the local velocity twist is represented by $\boldsymbol{\eta} := (\boldsymbol{g}^{-1}\dot{\boldsymbol{g}})^{\vee}$, which, similarly to rigid robotics, is linear in the joint velocities $\dot{\boldsymbol{q}}$. Regarding computation, the velocity twist of a point σ on the curve $\boldsymbol{g}$ can be represented as follows:

$$\begin{aligned} [\boldsymbol{\eta}]_{\boldsymbol{\Theta}}(\sigma, \cdot) &= \left[\boldsymbol{Ad}^{-1}_{[\boldsymbol{g}](\sigma,\boldsymbol{q})} \int_0^{\sigma} \boldsymbol{Ad}_{[\boldsymbol{g}](s,\boldsymbol{q})} \boldsymbol{\Theta}(s) \, ds \right] \dot{\boldsymbol{q}}, \\ &=: \boldsymbol{J}(\sigma, \boldsymbol{q}) \dot{\boldsymbol{q}}, \end{aligned} \tag{6.14}$$

where $\boldsymbol{J}(\sigma, \boldsymbol{q})$ denotes the geometric Jacobian that maps the joint velocities $\dot{\boldsymbol{q}}$ to velocity twist. For concisness, we write $\boldsymbol{J}_{\sigma} := \boldsymbol{J}(\sigma, \cdot)$. The Jacobian matrix holds significance not only for inverse kinematics but also for mapping external wrenches onto the generalized joint torques. For instance, it can be used to calculate the environmental forces as $\boldsymbol{f}_{\Omega_{\mathrm{env}}} = \int_0^L \boldsymbol{J}_{\sigma}^{\top} \boldsymbol{\mathcal{F}}_{\mathrm{env}} \, d\sigma$, where $\boldsymbol{\mathcal{F}}_{\mathrm{env}}$ represents a wrench related to the environment Ω_{env} described by SDFs.

Given the expressions in (6.11), (6.13), and (6.14), we can numerically evaluate the forward kinematics. We use a two-step Runge-Kutta integration solver that approximates the spatial integration. The forward kinematics solver is called by `shp = Shapes.string(q,dq)`, which stores all necessary numerical evaluations into a data structure `shp.Log.FK`. To further improve computation speed, `.mex` executable files are utilized.

Inverse beam kinematics (shape control). The inverse kinematics problem for soft continuum manipulators involves finding a solution $\boldsymbol{q}$ such that either (i) the end-effector reaches a specified setpoint, or (ii) shape control of the backbone is achieved. These manipulators often exhibit high levels of redundancy, so called hyper-redundancy [45]; leading to different solution approaches common to rigid robotics. Few modal basis representations possess a closed-form solution to the inverse kinematics, and they are typically solved using an iterative numerical method (*e.g.*, Newton Raphson). In *Sorotoki*, the inverse kinematics solver for sof beam models is implemented as `Shapes.IK`. We briefly detail the theory.

Suppose the desired shape of the soft manipulator is $\boldsymbol{g}_\star(\sigma) \in \boldsymbol{\mathcal{W}}_\sigma$ with

$$\boldsymbol{\mathcal{W}}_\sigma := \left\{ \boldsymbol{X} \in \mathrm{SE}(3) \mid \boldsymbol{X} = [\boldsymbol{g}]_{\boldsymbol{\Theta}}(\sigma, \boldsymbol{q}),\, \boldsymbol{q} \in \mathcal{Q} \right\} \tag{6.15}$$

the set of possible configurations of the backbone curve at σ. Note that $\boldsymbol{\mathcal{W}}_L$ spans the workspace of the end-effector, and $\boldsymbol{\mathcal{W}}_\Omega := \{\boldsymbol{\mathcal{W}}_\sigma \mid \sigma \in [0, L]\}$ the workspace of the entire soft body. For sake of readability, we redefine $\boldsymbol{g}_i(\boldsymbol{q}) := [\boldsymbol{g}]_{\boldsymbol{\Theta}}(\sigma_i, \boldsymbol{q})$ and $\boldsymbol{g}_i^\star := \boldsymbol{g}^\star(\sigma_i)$. We also redefine the geometric Jacobian by $\boldsymbol{J}_i(\boldsymbol{q}) := \boldsymbol{J}(\sigma_i, \boldsymbol{q})$.

Then, the inverse shape kinematics problem for the discretized soft manipulator can be formualted as an optimization problem of the form:

$$\begin{aligned} \underset{\boldsymbol{q}}{\text{minimize}} \quad & \Phi = \sum_{i=1}^{N_p} \left\| \boldsymbol{K}_p \mathrm{Log}_{\mathrm{SE}(3)} \left[\boldsymbol{g}_i^{-1}(\boldsymbol{q}) \boldsymbol{g}_i^\star \right]^\vee \right\|_2 \\ \text{subject to} \quad & \boldsymbol{g}_i, \boldsymbol{g}_i^\star \in \mathcal{W}_\Omega, \end{aligned} \tag{6.16}$$

where $\mathrm{Log}_{\mathrm{SE}(3)}$ denotes the logarithmic mapping from the Lie group to its algebra, see [**?**]. Despite the highly nonlinear nature of the optimization problem, its solution procedure is relatively straightforward two-step procedure:

Given an initial guess $\boldsymbol{q}^{(0)} \in \mathcal{Q}$, the aim is to compute an incremental update step that brings us closer to a local minimizer of the objective function Φ. For clarity, let $\boldsymbol{\Xi}_i := \boldsymbol{g}_i^{-1} \boldsymbol{g}_i^\star$ represent the geometric error between the soft robot and the desired shape. The state increment can then be expressed as:

$$\boldsymbol{\lambda}_i^{(k)} = \boldsymbol{J}_i^\top(\boldsymbol{q}^{(k)}) \left[\boldsymbol{K}_p \mathrm{T}_{\mathrm{SE}(3)}(\boldsymbol{\Xi}_i) \mathrm{Log}_{\mathrm{SE}(3)}(\boldsymbol{\Xi}_i) \right]^\vee, \tag{6.17}$$

$$\boldsymbol{q}^{(k+1)} = \boldsymbol{q}^{(k)} + \sum_{i=1}^{N_p} \boldsymbol{\Lambda}_i \left[\boldsymbol{\lambda}^{(k)} - \boldsymbol{\mathcal{N}}_i(\boldsymbol{q}^{(k)}) \nabla \Phi_{\mathrm{sub}} \right], \tag{6.18}$$

where $\mathrm{T}_{\mathrm{SE}(3)}$ denotes the tangent operator map on the group SE(3), see [32], $\boldsymbol{K}_p$ an artifical stiffness tensor, $\boldsymbol{\mathcal{N}}_i = (\boldsymbol{I} - \boldsymbol{J}_i \boldsymbol{J}_i^\top)$ represents the null-space projection, and $\boldsymbol{\Lambda}_i$ a diagonal activation matrix. The trivial choice is $\boldsymbol{\Lambda}_i = \boldsymbol{I}_n$. The nullspace projector can be extremely useful in exploring the high redundancy in

soft robots, allows us to consider subtasks described by Φ_{sub}. Classical examples of such subtasks include: minimizing elastic energy or obstacle avoidance. The iterative solver in (6.17) and (6.18) until convergence is $\boldsymbol{q}^{(k)}$ is achieved.

Example 6.6 (Contact kinematics of ultra-gentle soft gripper)**.** To showcase the forward and inverse kinematic solvers of Sorotoki, we will describe the ultra-gentle underwater soft gripper developed by Sinatra et al. [189]. The soft gripper consists of six soft fingers attached to a rigid palm base, where each soft gripper was designed to apply low contact pressure and minimize harm to common jellyfish species. An illustration of the sysem is shown in Figure 6.13. The delicate compliance of the soft gripper is achieved through the use of an extremely low durometer silicone matrix (Shore 20A). The actuator has a simple rectangular geometry, with a narrow cross-section of approximately 10×2 mm and an internal off-center rectangular hole. The thinnest part of the soft actuator, called the membrane, is approximately 0.35 mm thick, and length of about 130 mm.

In this study, we aim to reproduce a grasping scenario of a jellyfish modeled as a static SDF object. To achieve this, we first initiate a soft finger by utilizing the `Shapes` class. We employ a third-order Chebyshev basis to approximate the strain field. The geometric properties of the soft finger are specified through the functions `Shapes.setLength` and `Shapes.setGeometry`. Using a for-loop, we generate each soft finger sequentially, defining the spatial location of the fixed base with `Shapes.setBase`. Prior to deployment, each soft finger undergoes predeformation, which is calculated through the application of the forward kinematics solver `Shapes.FK(q)`. The joint configuration, `q`, is selected to match the experiments presented in Figure 6.13.

Subsequently, the deformed backbone is projected onto the surface of the SDF jellyfish through the use of the function `sdf.project`. The inverse kinematics solver is then invoked with `Shapes.IK`, resulting in the image depicted in Figure 6.13. Note that the inverse kinematics solver effectively places the soft finger onto the surface of the SDF object without causing penetration. The code for the forward and inverse kinematics is presented below:

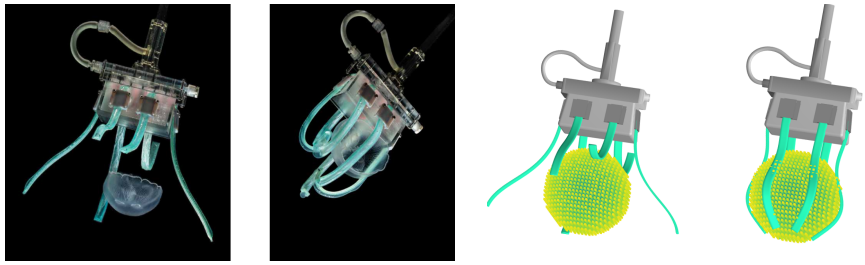


Figure 6.13. (Top) A snapshot of the ultra-gentle soft robot gripper developed by Sinatra et al. [189] is shown, demonstrating its ability to grasp a delicate jellyfish. (Bottom) The reconstructed soft gripper using *Sorotoki* is presented, where each tentacle finger is modeled individually using the *Shapes* class. It can be observed that the soft tentacles envelop the SDF object without penetration, indicating the successful accomplishment of task and subtask.

```
%% EXAMPLE: Shapes class (for/inv kinematics)
sdf = sSphere(0,0,-60,40)    % jellyfish SDF

% -- generate Shapes class
N = 200;                          % discretization
M = 3;                            % number of modes
pod = Basis(N,M,'chebyshev'); % build orth. basis
dof = [0,M,M,0,0,0];              % geom. strain dof

shp = Shapes(pod,dof,'Length',120);
shp = shp.setGeometry(sRectangle(4,1));

% -- inverse kinematics on SDF topology
for k = 1:6
    shp = shp.setBase(G{k});  % set base SE(3)

    pos = shp.FK(q0{k});     % forward kin.
    prj = sdf.project(pos); % project points

    qd  = shp.IK(prj);       % inverse kin.
    shp.render(qd);          % render shape
end

sdf.show();  % render jellyfish
```

6.5.5 Model composer (`Model`)

In many instances, soft robots comprise multiple dynamic components that are interconnected to form the overall system. For instance, the soft robotic hand depicted in Figure 6.1c comprises five actively controlled soft fingers connected to a rigid palm base, each of which exhibits its own fluid-structure interaction. While each soft finger can be modeled through the `Shape` class, the class itself lacks a composer or solver that addresses the interconnections between a network of dynamic systems.

To address the issue, we propose the `Model.m` class, which concatenates dynamic systems to systematically increase complexity. The class is equipped with an implicit solver that facilitates communication of state information between subsystems. The goal of the `Model.m` class is two-fold: (*i*) to facilitate the composition of multiple dynamic components that form a soft robotic system, and (*ii*) to leverage the dynamic network structure to design controllers through interconnection of subsystems. For example, a soft robot, the fluidic actuation, and the model-based controller can be modelled as three seperate entities and be composed into one global closed-loop system. This approach also enables the development of adaptive controllers that require additional state dynamics for online estimation of parameters.

A. Interconnected network of dynamical systems. The class `Model.m` allows users to compose an arbitrarily large network of dynamical systems that are presented in the state-space structure $\Sigma_i : \ \dot{\boldsymbol{z}}_i = \boldsymbol{f}_i(\boldsymbol{z}_i, \boldsymbol{u}_i, t)$. Then, the network of dynamical systems can written as

$$\Sigma_{\text{net}} : \dot{\boldsymbol{z}} = \boldsymbol{F}_{\text{net}}(\boldsymbol{z}, \boldsymbol{u}, t) \tag{6.19}$$

The network system matrices are assembled as: $\boldsymbol{F}_{\text{net}}(\boldsymbol{z}) := \text{blkdiag}(\boldsymbol{f}_1, \boldsymbol{f}_2, ..., \boldsymbol{f}_n)$. The implementation in *Sorotoki* is relatively straightforward. Let `f0 = @(z,u,t) ode(z,u,t)` be an anonymous function that defines a state space model, where `z`, `u`, `t` are the state vector, the input vector, and a time variable, respectively. In contrast to standard ODE solvers in *MATLAB* (*e.g.*, `ode45.m`), the input `u` is treated as an additional input to the ODE. This simplifies the design of controllers as it enables the definition of control laws outside of the ODE function caller, as opposed to the standard ODE packages in *MATLAB*.

To proceed, we first convert the function to a `sys0 = StateSpace(@f0)` class, which stores information on the state dimension, input dimension, and numerically computes the Hessian. Then, the model class is constructed using `mdl = Model(@f0)`. Other dynamic systems, represented by the `StateSpace` class `sys1`, `sys2`, and `sys3`, are added to the network by `mdl.addSystem(sys1,sys2,sys3)`. The `Model` class is also compatible with other classes, such as the `Fem` and `Shapes`

class, allowing for the interconnection between a FEM model and a soft beam model, for example. To solve the dynamics, we call `mdl.simulate([0,T])` which solves the state trajectories on the finite horizon domain $[0, T]$ using an implicit Trapezoidal solver (*i.e.*, similar to `ode23t` in `MATLAB`).

B: Assigning the controller. Once a network of dynamic systems has been composed in the `Model` class, assigning a control is straightforward. Controllers can be defined as auxiliary anonymous functions `@(mdl) Controller(mdl)` and added using `mdl.addController(@Controller)`. At each time instance of the implicit solver, this controller function is called and prompted with the current instance of the `Model`. Once prompted, the solver retrieves the global input vector $\boldsymbol{u}$ as in (6.19). All system information of the network can be retrieved within the function by `mdl.getState[I,J]`, where `I` is the system index and `J` the indices of the states of subsystem `J`. Such implementation presents a highly dexterous and efficient controller design platform previously not standard included in *MATLAB*.

C. Fluidic reservior with volume-variance The majority of soft robots are actuated through fluidics. In recognition of this, *Sorotoki* offers a variant of the `StateSpace` class, called `Fluidic`, which incorporates fluid dynamics of an enclosed pressure reservoir. The dynamics of such fluidic reservoir is determined by the compressibility of the fluid and the capacitance of the reservoir that is related to its volume $V > 0$. As a soft robot deforms, so does its internal volume. As such, consider a scalar variable $J : \mathcal{Q} \to (0, +\infty]$ such that we can describe the volume by $V(J) = JV_0$ where V_0 is the intrinsic volume. From a physical point of view, the scalar J can be seen as the relative volumetric change of the reservoir. Assuming constant temperature T and polytropic coefficient n_k, the pressure dynamics can be described by

$$\dot{p} = \frac{n_k}{V(J)} \left(RT\dot{m} - \frac{dV}{dJ}(J)\dot{J}p \right) - \mu_{\text{leak}} p \tag{6.20}$$

where the input is $\dot{m}$ the mass flux into the reservoir, R the ideal gas constant, and μ_{leak} a pressure leakage coefficient. For a perfectly enclosed system, the parameter $\mu_{\text{leak}} = 0$. Note however, that the volumetric change $\dot{J}$ might not always be available or it is difficult to derive. As an alternative, we substitute $\dot{J}$ with an "dirty derivative" approximation $\dot{\tilde{J}}$ following the work of Loria et al. [**?**]:

$$\dot{z} = \alpha_1[z + \alpha_2 J], \tag{6.21}$$

$$\dot{\tilde{J}} = z + \alpha_1 J, \tag{6.22}$$

where $\alpha_1, \alpha_2 > 1$ are filter gains. Hence, the system of equations (6.20), (6.21), and (6.22) leads to the full pressure dynamics. In case of an isochoric compression

(*i.e.*, no volume change $\dot{V} = 0$) with $J = 1 \Rightarrow \dot{J} = 0$, we can revert to a single-input-single-output (SISO) system with $u = \dot{m}$.

For many control applications for soft robotics, fluidic reservoirs are internally pressure regulated. Hence, the mass flow $\dot{m}(p, p_d)$ depends on the internal pressure p and a (possibly time-varying) pressure trajectory p_d. The mass flow controller of a proportional pressure-regulated reservoir is modelled here as PI-type controller:

$$\dot{m} = \Psi_{\mathrm{v}}(\nu, p), \tag{6.23}$$

$$\nu = k_p(p - p_d) + k_i \int p(\tau) - p_d(\tau) d\tau, \tag{6.24}$$

where $\Psi_{\mathrm{v}}(\nu, p)$ i a valve flow function related to mass flow saturation of the pressure valves, and k_p, $k_i > 0$ the proportional and integral control gains, respectively. In *Sorotoki*, such fluidic system is provided as `Fluidic` class. The volume of the reservoir can be set using `Fluidic.setVolume(@V)` where `V = @(x)...` is an auxiliary anonymous function. The mass flow function can be modified by the user using a similar command `Fluidic.setMassFlow(@M)`. By default, the flow function is chosen as $\Psi_{\mathrm{v}}(\nu, p) = \nu$, and the internal regulation is `Fluidic.isRegulated = true`.

Example 6.7. Simulation soft hand with fluidic soft fingers To demonstrate the versatility of the `Model` class and the `Fluidic` class, consider the example of modeling the soft robotic hand previously shown in Figure 6.1. We conducted an experiment where a predefined harmonic pressure signal was introduced to each of the soft fingers, with each harmonic having a $+\frac{\pi}{6}$ offset relative to its neighboring soft finger. The pressure signal is sinusoidal whose upper and lower bounds are -5 and 80 (kPa), respectively. As depicted in Figure 6.14, an oscillatory motion arises, where the fingers sequentially undergo bending.

The objective is to recreate the experimental oscillatory behavior of the soft hand, incorporating the fluid dynamics and continuum dynamics of the soft fingers. Each soft finger can be modeled separately using the `Shapes` class and its internal fluidic network using the `Fluidic` class. We start by sculpting the model for the soft fingers, assuming a fifth-order Chebyshev polynomial basis and considering planar bending curvature only. We assume that only the first mode can be actively controlled by the fluidic network, thus the input map is manually assigned using `Shapes.setInputMap(@(q) ... )`. The `Shapes` class is duplicated five times using a for-loop routine, with the base frame assigned accordingly at each iteration using `Shapes.setBase`.

For each soft finger, it is assumed that it has its own fluidic network that is equipped with internal pressure regulation. However, as each soft finger deforms,

the internal volume of the pressure reservoir also changes. This volumetric change is modeled as $V(\alpha) = V_0(1 + \tanh(\alpha))$, where it is assumed that $\alpha = 0.06q_1$ (i.e., the first joint of the soft beam) and V_0 the initial volume. Then, a for-loop is used to include the both `Shapes` and `Fluidic` systems to the global network of dynamical systems.

Finally, `mdl.addController(@Control)` is used to add the control law. In this auxilary function, we specifiy the pressure reference for the fluidic network, and we return the state deformations of the soft fingers to the fluidic network; required for the computation of the volumetric change. The dynamic simulation is solved implicity using the `mdl.simulate` command. The code for the dynamic simulation of the soft robotic hand is shown in Figure 6.14.

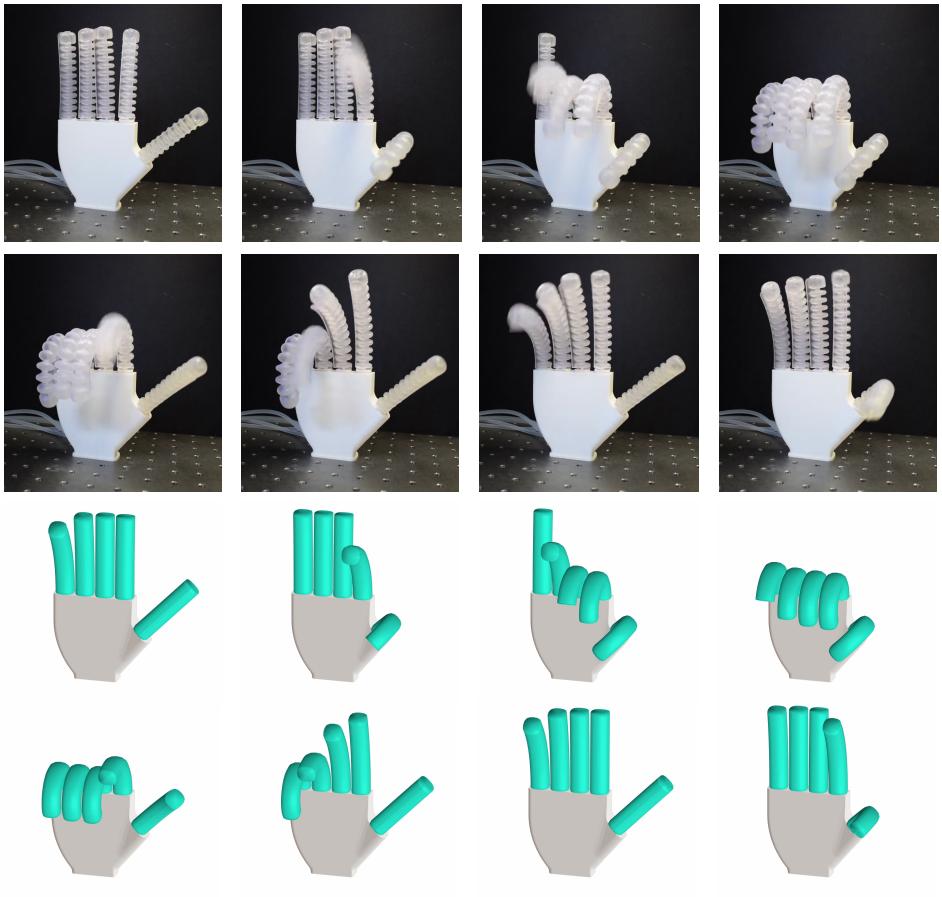


Figure 6.14. (top) Experimental snapshots of the open-source soft robotic hand provided by the *Sorotoki* toolkit. The five-fingered robot is subjected to a harmonic oscillator that commands oscillatory pressure to the individual fingers. Since each oscillator has a preset phase difference ($\phi = \pi/6$), the fingers of the soft robot hand undergo a periodic swinging motion. (bottom) Reconstructed soft robotic hand using *Sorotoki*, where each finger is modelled using the *Shapes.m* class and then composed using the *Model.m* class. Following, each oscillator is added using *mdl.addSystem* and then they ouputs are connected to the inputs of each *Shapes.m* class. As can be seen, the dynamics between the numerical model and the experiments are in close agreement.

```
%% EXAMPLE: Model class
shp = Shapes(pod,dof,'Length',90);

shp.setInputMap( @(x) [5; 0; 0; 0; 0]) ;
shp.Material = NeoHookeanMaterial(1.5,0.3);

for ii = 1:5
    SHP{ii} = shp.setBase( G{ii} );
end

%% building fluidics model
V0 = (4*pi/3) * 35^3;
V = @(x) V0 * (1 + tanh(x/0.06));

fld = Fluidic('NInput',3);
fld = fld.setVolume(V);

%% building model
mdl = Model([]);

for ii = 1:10
    if ii < 5    % add soft finger
        mdl = mdl.addSystem( SHP{ii} );
    else         % add fluidic network
        mdl = mdl.addSystem(fld);
    end
end

%% assign control and simulate
mdl.addController( @(x) Control(x) );
mdl = mdl.simulate([0, 5.0]);

% -- controller function block
function u = Control(mdl)
    phi = @(k) (k-1) * pi/6;
    Pd  = 80 * sign(sin(7.0 * mdl.t + phi(1:5)));

    u = zeros(20,1);
    u(6:3:end) = Pd;
    u(7:3:end) = mdl.getState(1:5,[1]);
    u(1:5)     = mdl.getState(6:10,[1]);
end
```

As depicted in Figure 6.14, the dynamics between the numerical model and the experiments are in close agreement. This serves as a testament to the efficacy of the dynamic model composer `Model` in building dynamic complexity through

its modular functionalities. Additionally, the ability to represent the controller as an auxiliary function that can retrieve state information at any given time confers *Sorotoki* with a high degree of flexibility in offline controller design.

6.5.6 Fluidic control hardware (`Control`)

The implementation of online controllers for physical soft robotic systems is a crucial aspect of *Sorotoki*. Although there are various options available in the research community [246], *Sorotoki* has a specific focus on fast closed-loop control. Drawing from our prior work [**?**], *Sorotoki* incorporates a TCP communication wrapper (`tcpip`) that enables real-time communication with a host computer, such as a Raspberry Pi (RPI). This host computer is connected to six pressure control boards, each capable of supporting up to two proportional pressure control valves from Festo. As a result, *Sorotoki* offers up to twelve pressure-regulated control ports with a range of -100 to 100 kPa that can be directly controlled using script-based programming in *MATLAB* . The entire system, including the software, is open-source and readily reproducible by researchers with diverse technical backgrounds.

By calling `brd = Control('ip','pwd')`, connection with the fluid control platform is establish, where `'ip'` is the IP address and password of the RPI. On the RPI, the Python script `ConnectToMatlab.py` must be executed that makes connection with *MATLAB* and awaits control commands. To initiate the control loop, a while-loop is used whose condition statement is `brd.loop(T)` where `T` is finite horizon time. Within the while-loop, all functionalities of *Sorotoki* are available, thus model-based controller design is possible for instance using the `Shapes` and `Model` classes. Each pressure regulation can be controlled using the command `brd.setPressure(id,P)` or an internal pressure measurement can be retrieved using `P = brd.getPressure(id)`.

Example 6.8 (Pick-and-place control of soft robot manipulator)**.** As an example of the capabilities of the fluid control platform, we used it for a pick-and-place application involving the aforementioned soft robot manipulator with soft gripper. The soft robot has four independent pressure inputs: three for the bellows network embedded into the soft body and one for the soft gripper. The *Sorotoki* toolkit communicates a desired pressure profile to a Raspberry Pi board computer, which is interfaced with an expandable array of proportional pressure regulators. As shown in Figure 6.15, the system successfully manipulates a 40 mm cylinder of 45 (g) into its container. The system has also been successfully simulated using the `Shapes` and `Model` class, shown in Figure 6.16, where the cylinder is modelled as a Newton-Euler rigid body system.

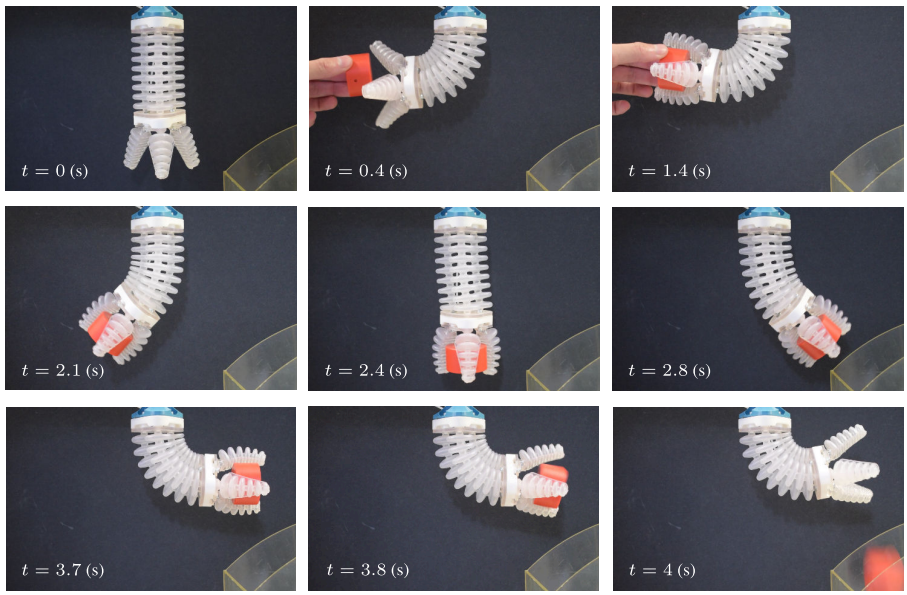


Figure 6.15. Implementation of open-loop control of a 3D-printed soft robot manipulator with a soft gripper using the *Sorotoki* toolkit. The soft robot has four independent pressure inputs: three for the bellows network embedded in the soft body and one for the soft gripper. The *Sorotoki* toolkit communicates a desired pressure profile to a Raspberry Pi board computer, which is interfaced with an expandable array of proportional pressure regulators. A straightforward pick-and-place task can then be easily programmed using the `Control` interface, using auxiliary *MATLAB* functions.

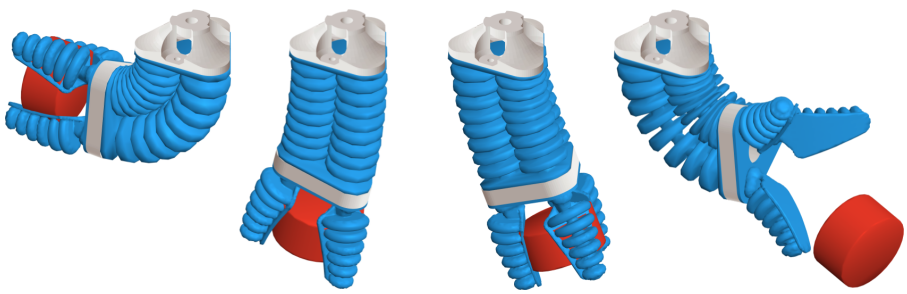


Figure 6.16. Simulation of the soft robot manipulator with gripper using the `Shapes` and `Model` class. The rigid-body is modelled by the Newton-Euler equation of motions implemented via `StateSpace`.

6.5.7 Computer vision (`Vision`)

The final class in *Sorotoki* addresses the challenge of soft sensing through vision. The class, named `Vision`, can be instantiated using `cam = Vision(Id)`, where `Id` is a user-specified index obtained from the list of available webcams using `webcamlist`. Alternatively, the `Vision` class is capable of reading sensor data from the RealSense D400 series RGB-Depth camera from Intel. The class is equipped with a suite of vision techniques that make use of the `OpenCV` Python implementation. It features three key functions: (*i*) extraction of optical markers from an image using RGB and depth data, (*ii*) calibration of the world coordinate frame using Aruco markers, and (*iii*) monitoring of a soft robot in real-time using camera feedback. The detection of color markers utilizes a circular Hough transform [**?**], which provides the pixel location of a circle within the specified search conditions. These tools provide a broad range of options for state estimation of a soft robot, which can be easily incorporated into closed-loop control schemes.

6.6 Soft robotics study cases

In the subsequent section, we will delve into the capabilities of the toolkit, *Sorotoki*. To provide a comprehensive overview of the toolkit, various problem scenarios will be considered, each with specific problem settings aimed at the design, modeling, or control of soft robots. We will also focus on widely cited academic works in the field of soft robotics and demonstrate these, mostly experimental, works using the *Sorotoki* toolkit. It is noteworthy that only brief code snippets will be presented, and the full code can be accessed in the repository under *./scripts/paper/*.

6.6.1 Multi-legged soft passive walker

In the first case study, the *Sorotoki* toolkit will be used to examine the dynamics of a multi-legged soft passive walker. The work of Suzumori and Saito [**?**] served as a key source of inspiration for this modeling problem. They proposed using a specialized soft structure that consists of an array of V-shaped soft legs, which exhibit stable intrinsic locomotion when placed on an inclined surface. This behavior was observed in experiments, as shown in Figure 6.17. The natural locomotion is driven by the elastic deformations of the V-shaped legs and their interaction with the environment, and is propelled forward by gravity. To increase the amplitude of these harmonics, small weights are placed at intermediate locations on the connecting soft body between pairs of V-shaped legs. Each pair of soft legs is tuned to a natural resonance frequency, and when coupled in parallel through a central deformable elastic body, synchronization occurs between the legs during

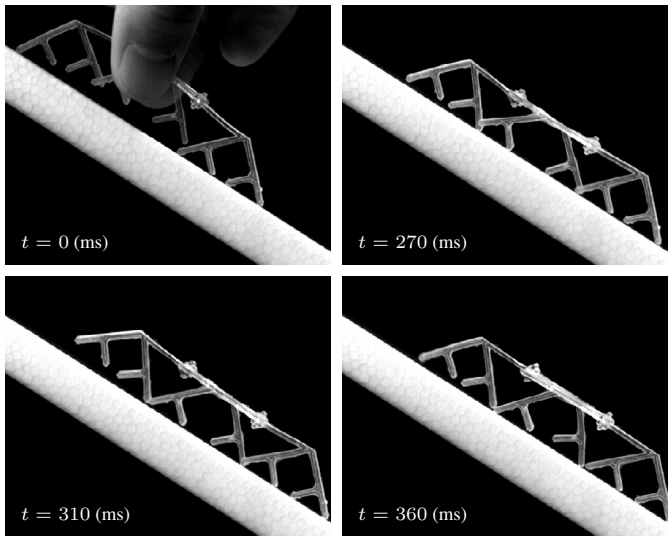


Figure 6.17. Snapsnots of the multi-legged passive walker from Suzumori and Saito [**?**] can be observed. The soft walker is placed on an inclined surface with a slope of $\varphi = -30°$ and initiates locomotion from an offset in the gravitational potential. These video frames were captured using a high-speed camera.

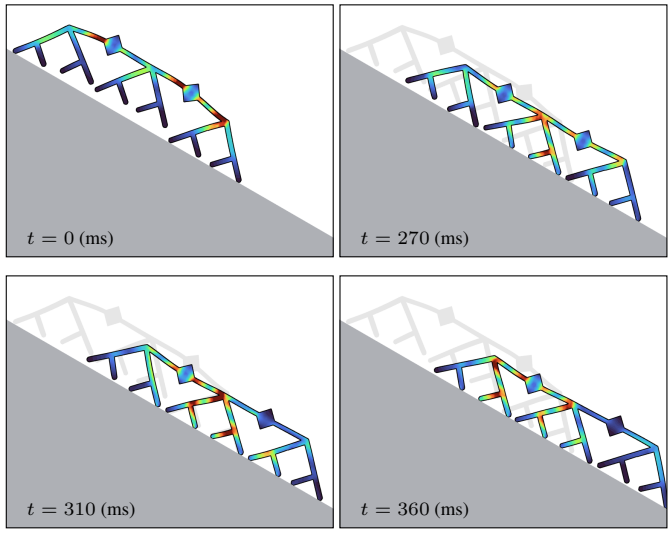


Figure 6.18. Snaphots of the multi-legged passive walker from *Sorotoki*. The experimental setup is similar to that described in [**?**]. By comparing the gaits in Fig. 6.17, a resemblance can be seen between the results obtained in *Sorotoki* and those reported in [**?**].

locomotion. In other words, after a transient period, each leg pair will converge to a similar limit cycle, but with a unique phase offset relative to its neighbors.

The objective of this study is to reproduce the dynamics of the soft passive walker described in Suzumori et al. [**?**] using *Sorotoki*. In the work of Suzumori et al. [**?**], the material parameters of the soft passive walker were given as: Young's modulus $E_0 = 1.2$ (MPa), Poisson ratio $\nu_0 = 0.49$, and density $\rho_0 = 3600 \cdot 10^{-12}$ ($\mathrm{kg\,mm^{-3}}$). Since the material model is not exactly specified in [**?**], a Neo-Hookean model was utilized with Rayleigh damping $\zeta = 1.5$.

To design the geometry of the V-shaped soft legs, the `sStrut(V1,V2,W)` function was utilized. This function generates an element of the `Sdf` class, requiring two nodal positions `V1,V2` and the strut's width `W` as input. The function was used to assemble a pair of legs iteratively, using the union operator implemented as *MATLAB*'s '`+`' arithmetic. The legs were then horizontally repeated three times with a uniform spacing of 25 mm. A coupling soft body was added, along with two weights at intermediate locations. The resulting SDF is first converted to an `.png` template and the imported in the `msh = Mesh('SDF.png','ElementSize',1.0)` to generate the finite element mesh. The code for the implicit modeling routine is given below

```
sdf = Sdf([]);
for i = 1:numel(Pts)
    sdf = sdf + sStrut(Pts{i,1},Pts{i,2},1);
end

R = sRectangle(2).rotate(pi/4);
W = R.translate([25,11]) + R.translate([50,11]);

sdf = sdf.repeat([25,0],3) + ...
        sStrut([12,11],[62,11],1) + W;

sdf.export('SDF.png');
```

The mesh is then utilized to construct the finite element model, *i.e.*, `fem = Fem(msh)`. The timestep for the implicit solver is set at $\Delta t = 0.33$ ms, which is set using `fem.setTimeStep(dt)`. Instead of modeling the inclined surface, which would also require rotating the mesh, the direction of the gravitational acceleration vector is modified as follows: $g := \mathrm{Rot}_y(\varphi) a_g$ with $\varphi = -\frac{\pi}{6}$ (rad). The gravitational acceleration is then added using the class function `fem.addGravity`. The inclined surface is modeled as a horizontal line SDF, which is used as a contact environment for the FEM model through the function `fem.addContact`. It should be noted that in Figure 6.17, the soft walker is held in place by two fingers, which results in initial deformations of the soft body and nonzero initial conditions for

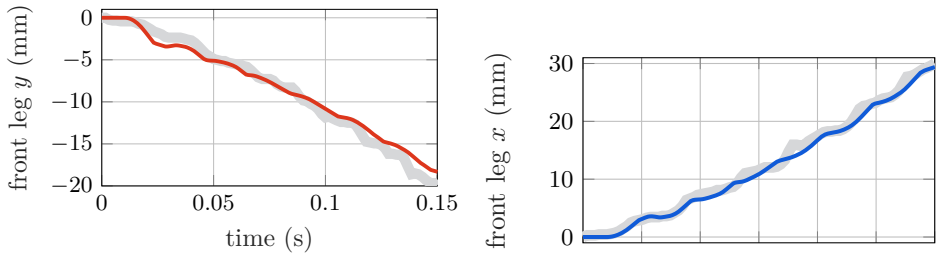


Figure 6.19. Comparison of the front leg rectilinear displacement between the experimental data obtained from Suzumori et al. [**?**] and the numerical data produced by *Sorotoki* shows that, although there are discrepancies, the step-like behavior and the traveled distance in the horizontal and vertical directions appear to be truthfully captured in comparison to the original experimental data.

the dynamic locomotion. To account for this, the mesh is pinned at the grasp locations, and the initial quasi-static deformations are solved for using `fem.solve`. Finally, the forward dynamics are solved implicitly using a Newmark-β solver by calling the routine `fem.simulate`.

Figure 6.18 shows snapshots of the dynamic simulation of the soft passive walker at times corresponding to those depicted in Figure 6.17. Although slight deviations are noticeable, the overall dynamic characteristics of the locomotion are captured closely by the dynamic FEM model produced using *Sorotoki*. To further demonstrate the validity of the model, a comparison of the rectilinear displacement of the front leg between the experimental data and the simulated model is presented in Figure 6.19. The experimental data is obtained from [**?**] and is shown in Figure 6.19 (in gray). As demonstrated in the figure, the step-like behavior is accurately captured by the numerical model, and the horizontal and vertical distances traveled by the numerical model closely match the original experimental data.

To examine the gait cycle, we introduce state variables $\theta_1, \theta_2, \theta_3$ to represent the joint angles between the V-shaped legs, as depicted in Figure 6.20. The trajectory of these angles over a small time window of 200 ms is shown in Figure 6.21. The angular movements exhibit a clear and consistent "stable" gait cycle, indicating that synchronization indeed occurs between the deformable soft legs due to their interaction with the deformable soft body. An analysis of the stable gait cycle reveals a gait period of approximately $T_{\text{gait}} \approx 47.5$ ms or $f_{\text{gait}} \approx 21.1$ Hz.

The numerical simulations presented in this study have effectively demonstrated the capabilities of the *Sorotoki* framework in accurately capturing the

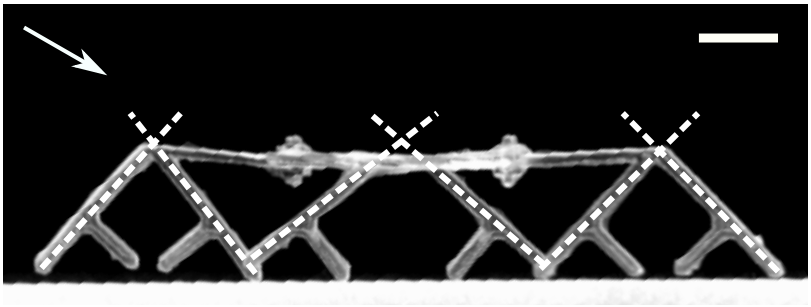


Figure 6.20. Definition of the angular deflection of the three pairs of soft legs, denoted by θ_1, θ_2, and θ_3, respectively. It should be noted that each pair has an intrinsic V-shaped structure, thus their stable equilibrium position during rest is approximately $\theta_i^\star \approx 45\ ^\circ$. Raw image obtained from [**?**].

Figure 6.21. Angular deflection of the V-shaped structure of the soft passive walker, simulated with Sorotoki. A clear gait cycle is observed in these deflections, indicating synchronization between the deformable structures due to the coupling of the soft body. By analyzing the stable gait cycles, a gait period of approximately $T_{\text{gait}} \approx 47.5$ ms or $f_{\text{gait}} \approx 21.1$ Hz is found.

complexities commonly encountered in dynamic analysis of soft robots. Furthermore, it has been demonstrated that the methodology proposed by Suzumori et al. [**?**] can be efficiently replicated using a minimal amount of code, specifically, approximately 30 lines within the *Sorotoki* framework.

6.6.2 Computational design of bending PneuNet actuator

In this section, we demonstrate the use of finite element models to aid in the design of PneuNet actuators, a popular type of soft robot actuator. PneuNet actuators, which have been in use since the 1980s, utilize a rectangular-shaped actuator with a stiffness differential to achieve a bending motion. Recent developments in the field, such as the work of Mosadegh et al. [152] and Ilievski et al. [102], have proposed modern variations of PneuNet actuators that incorporate an inextensible but flexible bottom layer to further enhance the bending motion. The motion of a soft actuator depends on the interaction between the soft material, structural geometry, and the locations where external loads are applied. In their work, Mosadegh et al. [152] demonstrated the importance of geometry in the performance of PneuNet actuators by proposing a new design, called the fast PneuNet (fPN), that improved upon the earlier slower PneuNet (sPN) designs presented in [102]. The fPN design requires less gas for inflation and thus signifi-

Figure 6.22. Evolution of the topology-based optimization routine in Sorotoki. At $k = 0$, we see the initial guess for the PneuNet actuator, and at $k = 100$ we see a converged solution of the optimizer. Observe that the algorithm proposes a solution very similar to the PneuNet, but instead, it has a teardrop shape rather than the classical rectangular shape. It is worth mentioning that the optimizer accounts for the hyper-elastic material properties - in this case, Elastic 80A resin by Formlabs.

cantly increases the actuator's performance. Design optimization for PneuNet soft actuators remains an active area of research, as evidenced by recent studies [196**?**].This demonstrates the continued interest in desing optimizing in soft PneuNet actuators, even decades after its initial development.

The purpose of this example is to demonstrate the use of Sorotoki's design capabilities to optimize and create a PneuNet actuator. We will apply an inverse design method to find the optimal configuration of a soft material that undergoes pure bending when pressurized. This approach is based on our previous research [**?**]. To extend of our prior study, we aim to show that the optimized designs produced through this computational design method can effectively overcome the Sim2Real hurdles. To find the optimal material arrangement, we will use a nonlinear topology optimization technique, specifically designed for compliant mechanisms.

The objective in the nonlinear topology optimization approach is to find the optimal material distribution $\boldsymbol{\rho}^\star = \text{argmin}_{\boldsymbol{\rho}} - \boldsymbol{L}^\top \boldsymbol{x}(\boldsymbol{\rho})$, where $\boldsymbol{L}$ is a sparse unit vector that selects the nodal displacements that promote bending motion. Once an optimum is found, the material distribution $\boldsymbol{\rho}^\star$ can be transformed into a 3D model and manufactured using a commercial 3D printing platform, such as Elastic 80A resin (Formlabs). The optimization algorithm can take into account the specific mechanical properties of the selected printing material, allowing for an optimal design that is tailored to these material properties.

To simplify the problem, we consider a single pressure chamber of the PneuNet actuator. To do this, a rectangular design domain with a size of 15×30 mm is

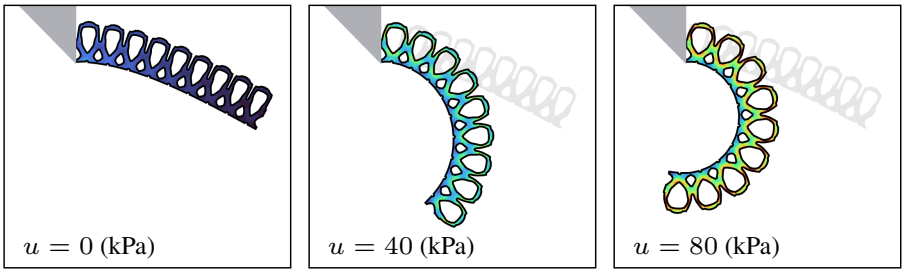


Figure 6.23. Nonlinear finite element simulation of the optimized PneuNet actuator using *Sorotoki*. The system is subjected to a linear ramp upto 80 kPa, and we observe the classical bending behavior of PneuNet actuators.

defined using a `Sdf` library within Sorotoki. The `Mesh` class is then utilized to generate a mesh, which is used to construct a finite element model (FEM) using the `Fem` class. The `Fem` class takes several arguments to set up the optimization solver conditions, including the volume infill, penalty value, filter radius, time steps, and the maximum number of iterations for the method of moving asymptotes (MMA).

The initial material distribution is set using `fem.initialTopology(sdf)` with `sdf = sCircle(5,[7.5,15])`, which creates a hole in the center of the actuator. The center element of the mesh is designated as an invariant pressure input and influences neighboring elements that satisfy the void conditions (i.e., $\rho_i \leq \varepsilon$ with $\varepsilon = 0.1$) using an efficient flood-fill algorithm. The influenced void elements undergo synchronous volumetric expansion to simulate a positive pressure load. Given its similarities to muscular contraction, the syntax for this function is added as `fem.addMyocyte`. The material properties of the Elastic 80A resin from Formlabs are then specified using `fem.Material = Elastic80A`. Boundary conditions are added to the FEM model using the syntax `fem.addSupport`. Finally, the optimization routine is started using the `fem.optimize` command. A code snippet is shown below:

```
% ------------------------------------------------
fem = Fem(msh,'TimeSteps',1/60);

fem.Material = Elastic80A;
fem = fem.initialTopology(sCircle(5,[7.5,15]));

fem = fem.addSupport('Left',[1,1]);
fem = fem.addSpring('Right',[0,1e-3]);
fem = fem.addOutput('Right',[0,1]);
fem = fem.addMyocyte('Center', 10 * kpa);

fem = fem.optimize('Compliant',...
```

```
                    'MaxIteration', 100, ...
                    'VolumeInfill', 0.3);
% ------------------------------------------------
```

The evolution of the material distribution during the first 100 optimization steps is depicted in Figure 6.22. These interpolated isosurfaces are taken from the discrete FEM mesh and show the intermediate design solutions. Surprisingly, the optimization algorithm generates a design that is reminiscent of the fast PneuNet design presented by Mosadegh et al. [152], but with a bellows-shaped pressure chamber in the form of an upside-down teardrop shape.

Next, the focus shifts to validating the optimization results. The aim is two-fold: (*i*) to validate that the optimization algorithm indeed produces the desired bending motion, and (*ii*) to verify if the design suggestion can be successfully transferred to reality (Sim2Real). To do this, the results of the optimization from `fem.optimize` are converted into a triangular mesh using `msh = fem.exportMesh`. Then, boundary conditions are assigned, such as a clamped boundary, gravitational loads, and internal pressure loads for each embedded PneuNet chamber. The same material model for `Elastic80A` resin is chosen. The quasi-static FEM simulation results of the optimized PneuNet actuator for linearly increasing pressure loads of $u = 80$ kPa are shown in Figure 6.23. As can be seen, the desired bending behavior is achieved in the simulation.

Next, the optimized isosurface shown in Figure 6.22 can be transformed into a 3D CAD model and printed as a physical soft actuator using a Form3+ SLA printer (FormLabs) with Elastic 80A resin. To validate its performance, the soft actuator is subjected to a linearly increasing pressure load of 80 kPa. As seen in Figure 6.24, the optimized soft actuator successfully performs the desired bending motion, indicating the feasibility of crossing the Sim2Real barrier.

To quantify the discrepancies between the FEM predictions and the actual system, an optical marker is placed at the tip of the soft actuator. The spatial coordinates of the optical marker are obtained using the `Vision` class of *Sorotoki*, which uses the color-filtered Hough-space circle transformation to return the pixel coordinates of the marker. These measurements are collected using a RealSense D435 RGB-depth camera (Intel). To retrieve the spatial location of the color marker, we use the command `cam = Vision('realsense')` together with the function `cam.getMarker(R,rgb)`, where `R` is an estimate of the color marker radius in pixels, and `rgb` is the RGB color value of the marker.

The comparison between the FEM predictions and experimental results is presented in Figure 6.25. The deformation patterns of the FEM model and the physical system show close agreement, with an average error of ± 2 mm. However, there is a noticeable difference in the initial conditions, as shown in Figure 6.25. For

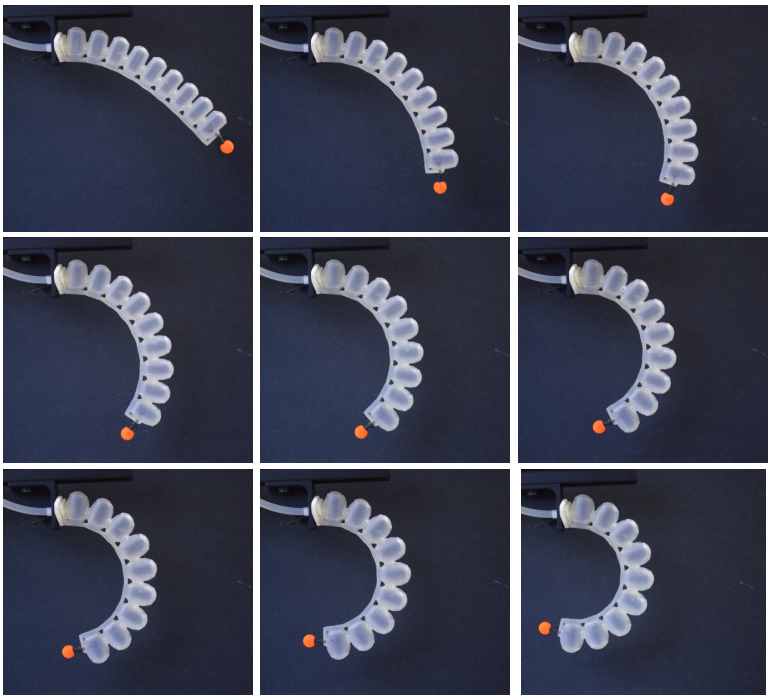


Figure 6.24. Validation study of a 3D-printed *PneuNet* actuator optimized using *Sorotoki*. The soft actuator is printed using SLA on a Form3+ printer using Elastic 50A UV-resin. Similar to the numerical simulations, we vary the pressure from 0 to 80 kPa with a linear ramp. To measure the planar displacement, an orange marker is placed such that *Vision.m* can be employed.

$u = 0$ kPa, under pure gravitational loads, the deformations deviate significantly. The cause of this discrepancy is believed to be related to post-curing and internal stress relaxation of the photopolymerization process. This suggests that the stress-free configurations of the FEM model and the physical system may differ, but accounting for this stress-relaxation phenomenon in photopolymer printing is outside the scope of this study and the Sorotoki toolkit.

Despite the presence of some differences, the numerical and experimental examples presented in this study highlight the ability of the computational design framework within Sorotoki to generate purposeful and useful material distributions with limited prior knowledge of conventional soft robotic design practices. This not only speeds up the design process, but it also opens up the possibility of creating new and innovative soft robot forms.

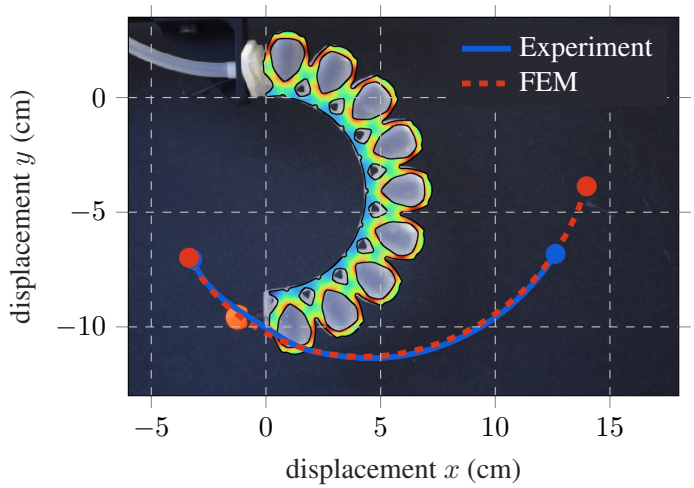


Figure 6.25. Comparison between the numerical simulation in Sorotoki and the experimental results where the orange marker is tracked using the Vision.m class is shown. The results indicate a close overall trend between the simulation and experiment. However, a discrepancy in the initial deformation ($u - 0$ kPa) is observed. It is hypothesized that this discrepancy is attributed to the inherent creep of SLA resin materials, which leads to a predeformed continuum due to the slow relaxation of the material upon actuation.

6.6.3 Dynamic manipulation of high dexterity soft gripper

In this section, we will examine the use of reduced-order models for soft beams within the context of the *Sorotoki* software. These models are designed for efficient simulation by exploring minimal state representations of the dynamics of continuum systems. To demonstrate the capabilities of the soft beam modeling framework within Sorotoki, we will consider a specific example of a soft robotic system proposed by Suzumori et al. in their works [203, 204]. Despite being published in the early 1990s, the work by Suzumori et al. is still recognized as a seminal contribution to the field of soft robotics technology and remains relevant to this day.

In their research, Suzumori et al. developed a highly dexterous soft gripper consisting of four microfluidic soft actuators driven by an electro-pneumatic control system. Each finger has three internal pressure chambers, which together provide three controllable degrees of freedom at the fingertip, including pitch, yaw, and linear stretch. Unlike classic rigid grippers, the soft body of the grip-

per conforms to external forces, enabling intrinsic adaptation during grasping or manipulation tasks. As an illustration of the static grasping capabilities of this soft gripper, Figure 6.26 depicts two grasping configurations: a pinch grasp for a 40 mm diameter glass beaker (left) and a two-finger pair-pinch grasp for a 5 mm thick metal wrench (right). Suzumori et al. then employed inverse kinematic and compliance control to relate the tip position and compliance to input pressure values. As shown in Figure 6.26, they were able to successfully hold and turn a 10 mm hexagonal bolt, with an average speed of 0.25 revolutions per second. Due to the wide range of dexterous actions performed by the gripper, the soft gripper proposed by [203, 204] remains a seminal contribution to the field of soft robotics, demonstrating the potential of the technology.

The aim of this study is to reproduce the static grasping and dynamic bolt-screwing experiments performed by Suzumori et al. in their works [203, 204] using the `Sdf`, `Shapes`, and `Model` classes in the *Sorotoki* software. The design specifications of the soft gripper are based on the parameters provided by Suzumori et al., which include a radius of $R_0 = 6$ (mm) and an assumed length of $L_0 = 80$ (mm) for each finger. However, the material properties of the gripper are not specified in the original works. As a solution, we propose a Neo-Hookean material model with a Young's modulus of $E_0 = 1.0$ (MPa), Poisson ratio of $\nu_0 = 0.3$, and density of $\rho_0 = 2000 \cdot 10^{-12}$ ($\mathrm{kg\,mm^{-3}}$). The reduced-order model for each soft finger in interaction with a rigid object, denoted by $\Sigma_{\mathrm{SR},i}$, is described by the following equation:

$$\boldsymbol{M}(\boldsymbol{q}_i)\ddot{\boldsymbol{q}}_i + \boldsymbol{h}(\boldsymbol{q}_i, \dot{\boldsymbol{q}}_i) = \boldsymbol{G}(\boldsymbol{q}_i)\boldsymbol{p}_i + \sum_{j \in \mathcal{S}_{\Omega_{\mathrm{env}}}} \boldsymbol{J}_{v,j}^{\top}(\boldsymbol{q}_i)\Big[\boldsymbol{\mathcal{F}}_{n,j}(\boldsymbol{q}_i) + \boldsymbol{\mathcal{F}}_{t,j}(\boldsymbol{q}_i, \dot{\boldsymbol{q}}_i)\Big], \tag{6.25}$$

where $\boldsymbol{h}(\boldsymbol{q}_i, \dot{\boldsymbol{q}}_i)$ represents the collection of nonlinear internal forces, $\boldsymbol{p}_i$ is the prescribed pressure input, $\boldsymbol{J}_v(\boldsymbol{q}) := \lfloor \boldsymbol{J}(L, \boldsymbol{q}) \rfloor_3$ is the linear velocity part of the generalized Jacobian matrix of the tip, $\mathcal{S}_{\Omega_{\mathrm{env}}}$ represents the nodal indices that penetrates the object, and

$$\boldsymbol{\mathcal{F}}_n = -\mu_e d \cdot \boldsymbol{e}_n, \quad \boldsymbol{\mathcal{F}}_t = -\mu_v |\boldsymbol{\mathcal{F}}_n| \operatorname{sign}(\dot{d}) \cdot \boldsymbol{e}_t$$

denote the normal and tangent contact forces between the i-th finger and the rigid object, respectively. The distance between the finger and the object is given by $d = \mathtt{sdf}_{\Omega_{\mathrm{env}}}(\boldsymbol{\gamma}_L(\boldsymbol{q}))$, where $\mathtt{sdf}_{\Omega\mathrm{env}}(\cdot)$ represents the signed distance function of the rigid object and $\boldsymbol{\gamma}_L(\boldsymbol{q})$ the finger's tip position. The parameters $\mu_e, \mu_v > 0$ represent the contact coefficients.

In this study, we utilize a third-order Chebyshev polynomial basis to model the deformation of the pneumatic soft robot's finger. The basis is sampled over

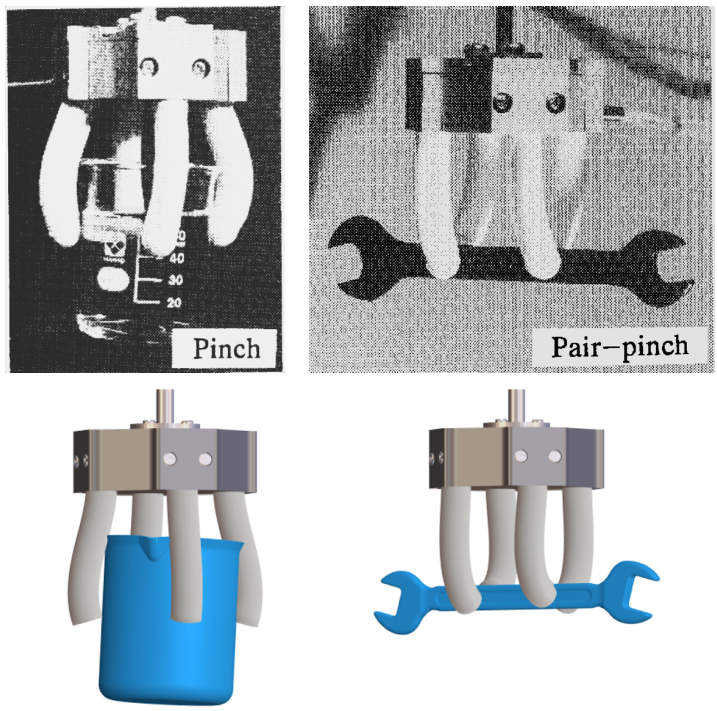


Figure 6.26. (top) Pinch and pair-pinch grasping configurations of the soft gripper proposed by Suzumori et al. [203, 204]. As can be seen, the soft gripper is capable of securely holding a glass beaker of approximately 40 mm in diameter, and it can grasp a wrench with a thickness of approximately 5 mm. The high compliance of each soft finger allows for an adaptive, stable grasp that conforms to the shape of the rigid object. (bottom) A reconstructed soft gripper using the Sorotoki framework. To model each soft finger, we utilized the `Shapes` class and composed the entire gripper using the `Model` class. The rigid objects were modeled using the `Sdf` class. We observed a close resemblance between our simulation model and the original experiments performed by Suzumori et al. in [203, 204].

$N_p = 100$ uniform nodes and is assembled into a matrix using the command `pod = Basis(100,3,'chebyshev')`. It is assumed that only free strains occur in the bending, while elongation and torsion are constrained. As a result, each strain mobility vector is characterized by three modes of the Chebyshev basis, specified as `dof = [0,3,3,0,0,0]`, leaving the κ_x and κ_y curvatures free. The dynamic model for each finger is constructed using the command `shp = Shapes(pod,dof)`. The material properties of the finger are assigned using `shp.Material = NeoHookean(1.0,`

`0.3)`. To set the geometry of each finger, we call `shp.setLength(80)` to set its length, and `shp.setGeometry(sCircle(6))` to set a circular cross-section of radius 6 (mm). Each finger of the soft gripper model is equipped with three fluidic chambers that are radially distributed along its circumference. As a result, the input map $\boldsymbol{G}$ for each finger becomes a nonlinear, non-square matrix. We assume that these distributed forces can be represented by a tangent bundle of linear forces that are positioned 3 (mm) away from the center axis. To assign these forces, the `shp.addFluidic(@p)` command can be utilized, which requires an anonymous function `@p` that describes the path of the soft actuator. A sample code for assembling one soft finger model is given below.

```
dof = [0,3,3,0,0,0];
pod = Basis(100,5,'chebyshev');

shp = Shapes(pod, dof);
shp = shp.setLength(80);
shp = shp.setGeometry(sCircle(6));
shp.Material = NeoHookeanMaterial(1.0, 0.3, ...
                'Rho', 2000 * 1e-12);

% for loop to assemble fluidic chambers
for ii = 1:3
    phi  = (ii - 1) * pi / 3
    path = @(s)  3.0 * [cos(phi); sin(phi); s];
    shp  = shp.addFluidic(path);
end

shp = shp.rebuild(); % pre-assemble
```

The full soft gripper model, composed of four identical finger *Shapes* classes, can be assembled using a for-loop routine. In this process, a class representing each finger is copied and assigned a unique SE(3) base frame to each instance. Each finger is placed in a circular array with a radius of 37.5 (mm) relative to the center axis of the gripper body. The contact domain for each finger is specified using the method `shp.addContact(sdf)`, where `sdf` denotes the signed distance field of the contact object. For the beaker example, a cylindrical SDF with dimensions $40 \times 40 \times 60$ (mm) is considered, while for the wrench, a rectangular SDF with dimensions $5 \times 10 \times 100$ (mm) is used. Subsequently, each instance of the `Shapes` class is appended to the `Model` class constructor using the `mdl.addSystem(shp)`.

Once the global model of the soft gripper has been assembled, it can be controlled using an open-loop controller. In the case of a static grasping scenario with a glass beaker and wrench, we apply pressure ramps to each pressure chamber of the soft gripper through an auxiliary anonymous function, `@(t) Controller(t)`.

The function takes in a time variable and outputs a column vector of pressure signals, represented by $\boldsymbol{u} = (\boldsymbol{p}_1, \boldsymbol{p}_2, \boldsymbol{p}_3)^\top$. The controller is then assigned to the Model class using the command `mdl.addControl(@Controller)`, which is executed at each simulation step. The forward dynamics of the soft gripper's interaction with the object are implicitly solved through the `mdl.simulate()` routine. The simulated grasping configurations are shown in Figure 6.26. It is evident that there is close agreement with the experiments in [203, 204].

Subsequently, we aim to reproduce the hexagonal bolt screwing experiment of [203, 204], which involves a more complex simulation than the previous scenario due to the dynamic interaction between the environment and the soft robot. To accurately depict this interaction, we must also incorporate dynamics into the signed distance field describing the hexagonal bolt screw. We assume a rotational mass-damper system, represented by the following equation:

$$I_\Omega \, \ddot{\theta} = -\mu_\Omega \dot{\theta} - \sum_{i=1}^{N_{\text{finger}}} \sum_{j \in \mathcal{S}_\Omega} \boldsymbol{r}_j(\boldsymbol{q}_i) \times \boldsymbol{\mathcal{F}}_{t,j}(\boldsymbol{q}_i, \dot{\boldsymbol{q}}_i) \cdot \boldsymbol{e}_3, \tag{6.26}$$

where I_Ω = is the inertia of the hex-bolt, $\mu_\Omega = 2.5$ its friction coefficient, and $\boldsymbol{r}$ the relative position vector from the point of contact and the central turning axis of the screw. Note that we only include the tangential force components $\boldsymbol{\mathcal{F}}_t$ that are responsible for motion, as the normal forces are assumed to have a net zero-torque contribution. The model described in equation (6.26) is incorporated into the simulation by using the command `mdl.addSystem(@f)`, where `@f(x,u,t)` is an anonymous function that represents the state space. The required input `u` for equation (6.26) is connected to the soft gripper by utilizing the `mdl.addController(@Controller)` command, which inputs tangential reaction forces into the screw model. The controller also includes the prescribed pressure profile $\boldsymbol{p}_{1,2,3,4}$ for each of the four soft fingers.

Remark 6.3 The inertia of the steel bolt I_Ω has been derived using standard ISO metric profiles of M10 screws. The screwing friction, however, has be determined using an ad-hoc approached, where the parameter has been tuned until desired behavior appears.

The comparison between the experiments conducted by Suzumori et al. [203, 204] and our numerical results obtained using *Sorotoki* is depicted in Figure 6.27. Similar to the simulation of the static object grasping scenario, the simulation of the Suzumori soft gripper's screwing experiment accurately reflects the real-world experiment. Not only do we observe similar deformation characteristics in the soft

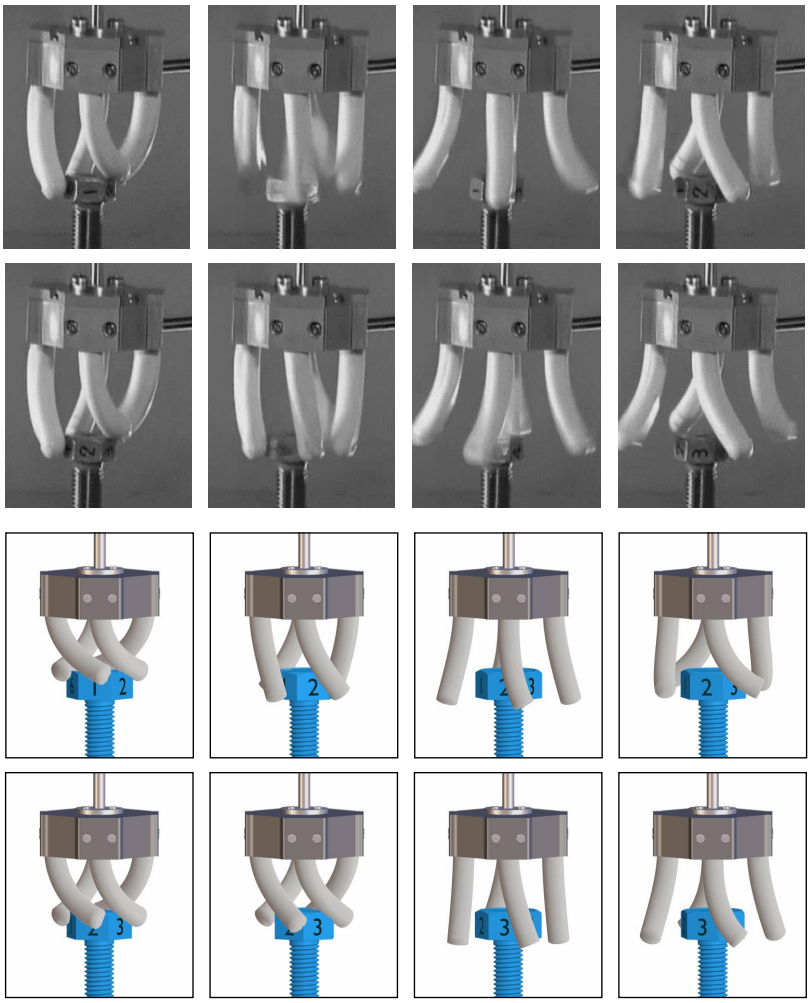


Figure 6.27. (top) Snapshots of the bolt screwing experiment with the soft gripper, as presented in the work of Suzumori et al. [203, 204], are displayed. The soft gripper periodically switches through a predefined set of configurations, enabling the holding and manipulation of a hexagonal bolt screw. In the experiment, a bolt turning rate of 0.25 rps was achieved. (bottom) The bolt screwing experiment was reproduced using *Sorotoki* . In the simulation, each finger was modeled utilizing the class `Shapes` and assembled together into `Model`. The contact interaction with the bolt was modeled using signed distance functions (SDFs), and a rotational mass-damper model was used to describe the dynamics of the bolt. By utilizing solely the frictional interaction between the fingers and the screw, the experiment of Suzumori et al. was successfully reproduced.

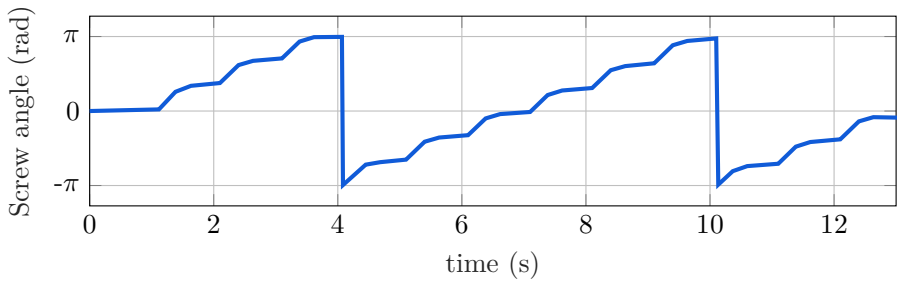


Figure 6.28. The evolution of the hexagonal bolt angle $\theta(t)$ is depicted, where the stair-like trajectory of the screwing motion, as observed in Suzumori's experiment, is apparent. Through careful parameter and input shaping, a bolt-screwing motion of 0.16 rps was achieved. This is slightly slower than the reported rate of 0.25 rps, however, it is believed that the underlying morphological characteristics are accurately captured.

finger, but we also observe the step-like turning of the bolt screw. To highlight its rotational motion, we present the rotation angle $\theta(t)$ in Figure 6.28, which shows that a bolt-screwing motion of 0.16 rps was achieved. Although this rate is slightly slower than the reported rate of 0.25 rps, it is believed that the underlying morphological characteristics are accurately captured. Note that although the system is of highly complex nature, the simulation program contains less than 100 lines of code (including visualization).

6.6.4 Impedance control of soft manipulator with static environmental interaction

In the next section, we will concentrate on designing controllers using the `Model` and `shapes` classes. The simulations of the Suzumori gripper performed previously were controlled in an open-loop manner, with the primary complexity arising from the interaction between the model and the dynamic object. We will now explore the potential of model-based controllers in *Sorotoki*, where we take inspiration from the work of Della Santina et al. [**?**] as an potential study-case.

The study by Della Santina et al. [**?**] presents a design for model-based controllers for soft robot manipulators, highlighting two control architectures: dynamic tracking and surface tracking. The authors proposed a Cartesian impedance controller for the latter architecture, which actively regulates the desired compliance behavior of the soft robot's end-effector in a static environment. Additionally, the work presented a contact path planning algorithm that initially brings

the robot close to a desired setpoint on the surface (Phase 1: Approach), and then adjusts the setpoint to maintain contact with the surface (Phase 2: Explore). The proposed controller was validated both numerically and experimentally on a six-link soft robot manipulator, demonstrating that model-based controllers can lead to higher-level performances in soft robotic systems.

To maintain computational performance, the impedance controllers used in Della Santina et al.'s study [**?**] incorporate an augmented rigid body model. This model employs Constant-Curvature kinematics to approximate the center of mass of the continuously deformable robot by projecting its mass distribution into a lumped mass description. This leads to an Euler-Lagrangian representation similar to the commonly used Denavit-Hartenberg (DH) parametrization models in rigid robotics. Moreover, this approach maintains classical properties, such as positive semi-definiteness of the inertia matrix and passivity properties, which are valuable for stability analysis.

Our aim in this section is to replicate the results of the closed-loop controlled multi-link soft robot during dynamic contact that were presented in the study by Della Santina et al. [**?**]. Instead of employing the augmented rigid body model used in their work, we explore a reduced-order beam model, in which each link is represented as an inextensible, continuous PCC segment. This approach extends their work to a distributed mass robotic system. As a template for the soft manipulator model, we again utilize the `Shapes` class. According to [**?**], each CC segment of the soft manipulator has an intrinsic length of $\delta L_0 = 60$ (mm) and a mass of $m_0 = 334$ (g). The robot has a rounded rectangular cross-section of 60×20 (mm), which is described using `sdf = sSquircle`. The density of the soft robot manipulator, given its length and geometry, is approximated to be $\rho_0 = 1200 \cdot 10^{-12}$ (kg mm^{-3}).

A crucial aspect of the simulation is the dynamic interaction with the environment. Therefore, a static environment must be assigned to the dynamic model. While [**?**] presents multiple examples of dynamic contact, this study focuses on the experiment with a 40° slanted surface with wave indentations, as shown in Figure 6.32. The surface features were extracted from the image data presented in [**?**] and the `env = sPolyLine(V)` function was used to generate the SDF environment, where `V` is a polyline vector. The enviroment is then added using `Shapes.addContact(env)`. Once all settings are assigned to the class `Shapes`, the model is constructed by `mdl = Model(Shapes)`. The code for assembling the dynamic model is given below:

```
dof = [0,6,0,0,0,0];
pod = Basis(100,6,'pcc');

```

```
shp = Shapes(pod, dof);
shp = shp.setLength(60 * 6);
shp = shp.setGeometry(sSquircle(3, 3, 0.25));
shp = shp.addContact(sPolyLine(V));
shp.Material = NeoHookeanMaterial(1.0, 0.3, ...
                'Rho', 1200 * 1e-12);

% build model (calls rebuild automatically)
mdl = Model(shp);
```

Given the `Model` class, we can now start deriving the control law. For conciseness, let $\boldsymbol{J}(\boldsymbol{q}) := \boldsymbol{J}(L, \boldsymbol{q})$ be the geometric Jacobian of the end-effector. We also introduce the Cartesian inertia matrix as $\boldsymbol{\Lambda} := (\boldsymbol{J}^\top \boldsymbol{M}^{-1} \boldsymbol{J}^\top)^{-1}$. Then, the proposed Cartesian stiffness controller given in [**?**] can be written as:

$$\boldsymbol{\tau} = \boldsymbol{J}^\top \Big[\boldsymbol{J}_M^{+\top} \boldsymbol{f}_e + \boldsymbol{f}_g + \boldsymbol{J}^\top \boldsymbol{\eta}_{\boldsymbol{C}} (\boldsymbol{I} - \boldsymbol{J}_M^{+\top} \boldsymbol{J}) \dot{\boldsymbol{q}} + \dots \\ \dots + \boldsymbol{J}^\top (\boldsymbol{K}_c (\boldsymbol{\gamma}_d - \boldsymbol{\gamma}_L) - \boldsymbol{D}_c \boldsymbol{J} \dot{\boldsymbol{q}}) \Big], \quad (6.27)$$

where $\boldsymbol{\gamma}_d$ and $\boldsymbol{\gamma}_L$ represent the desired and true end-effector positions, respectively; and $\boldsymbol{K}_c$ and $\boldsymbol{D}_c$ are the desired stiffness and damping of the end-effector. The closed-loop controller employs the dynamically consistent pseudo-inverse of the Jacobian, denoted as $\boldsymbol{J}_M^+$, which is defined as $\boldsymbol{J}_M^+ := \boldsymbol{M}^{-1} \boldsymbol{J}^\top \boldsymbol{\Lambda}$. The controller also utilizes the Cartesian Coriolis terms, denoted as $\boldsymbol{\eta}_{\boldsymbol{C}}(\boldsymbol{q}, \dot{\boldsymbol{q}})$, which are expressed in the Cartesian frame and given by:

$$\boldsymbol{\eta}_{\boldsymbol{C}} = \boldsymbol{\Lambda} (\boldsymbol{J} \boldsymbol{M}^{-1} \boldsymbol{C} - \dot{\boldsymbol{J}}) \quad (6.28)$$

The closed-loop controller, implemented as an anonymous function, is derived from four system matrices: the inertia matrix $\boldsymbol{M}(\boldsymbol{q})$, the Coriolis matrix $\boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})$, the Jacobian $\boldsymbol{J}$, and its time derivative $\dot{\boldsymbol{J}}$. In *Sorotoki*, these matrices are automatically computed at each solver step and stored in a data structure named `shp.log`. The closed-loop controller can access this data structure. A sample code of the controller is provided below:

```
% ------------------------------------------------
function tau = Control(mdl)
    % retrieve current log
    log = mdl.System{1}.log;

    J  = log.FK.J(:,:,end);
    dJ = log.FK.dJdt(:,:,end);

    Mi  = inv(log.EL.M);
```

```
    Lam = inv(J * Mi * J.');
    Eta = Lam * (J * Mi * log.EL.C - dJ);
    Jmi = Mi * J.' * Lam;

    if mdl.System{1}.isContact()
        xd = [0.3637; 0.1406];
    else
        xd = [0.2307, 0.2576];
    end

    % impedance controller
    tau = J.' * (Jmi.' * fe + fg + ...
        Eta * (log.I - Jmi.' * J) * log.dq + ...
        (Kc * (xd - log.FK.p) - Dc * J * log.dq));
end
% ------------------------------------------------
```

It is noteworthy that the controller above uses the command `shp.isContact` to detect if the robotic system is in contact. Similar to path planning in [**?**], upon contact, a new desired equilibrium position is adopted. These desired equilibrium positions for the end-effector are in line with the approach presented in [**?**]. Subsequently, the implicit solver is invoked by calling `mdl.simulate` to solve the closed-loop dynamics. Snapshots of the dynamics have been presented in Figure 6.32 which are produced by calling the function `shp.render`. Figure 6.29 shows the trajectory of the end-effector (dashed lines are the desired setpoints), Figure 6.30 shows the evolution of the states, and Figure 6.31 shows the control action $\boldsymbol{\tau}$ from (6.27).

When comparing the experimental results reported by Della Santina et al. [**?**] and the numerical simulations produced by *Sorotoki*, we observe similar deformation characteristics in both systems. Most notably, the numerical implementation of the impedance Cartesian stiffness controller also follows the inclined surface until the setpoint is reached. These similarities highlight the reliability of Sorotoki in accurately reflecting true soft robotic systems, even in closed-loop scenarios.

6.6.5 Contact robust shape sensing of elastomer soft actuator (PneuNet) using a FEM-based modal basis

In the next section, our focus shifts from simulation to the experimental domain. Our primary focus will be on the `Vision` and `Control` classes, and we aim to provide experimental validation for the Data-driven Variable Strain (DVS) basis approach detailed in Section 6.5.4. The objectives of this study case are (*i*) to derive a finite-dimensional Cosserat beam model of the PneuNet actuator and (*ii*)

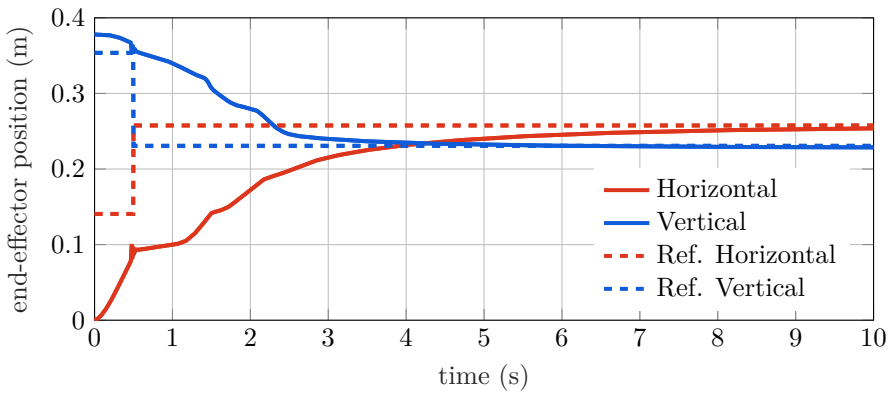


Figure 6.29. The end-effector trajectories of the six-link soft manipulator, where {—, —} denote the horizontal and vertical position, respectively. The setpoints assigned by the controller are indicated by the dashed lines. Note that at $t \approx 0.88$ (s), the point of contact, the controller switches setpoint.

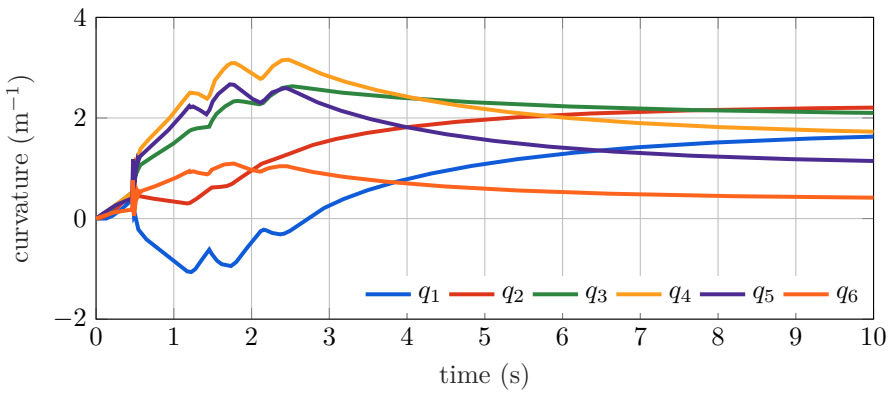


Figure 6.30. State evolutions of the six-link soft manipulator, where the states represent the planar curvature of each individual soft link.

to implement a real-time shape sensing algorithm that is robust against external forces through the exploration of model information.

We begin our investigation by conducting a nonlinear dynamics analysis of a soft PneuNet actuator, the geometry of which has been selected to match that described in the work of Mosadegh et al. [152]. The soft actuator is suspended vertically in order to produce minimal deformation under zero-input conditions. In pursuit of high-accuracy simulation, we first perform a Finite Element simulation. The generation of the mesh is performed using the function `msh =`

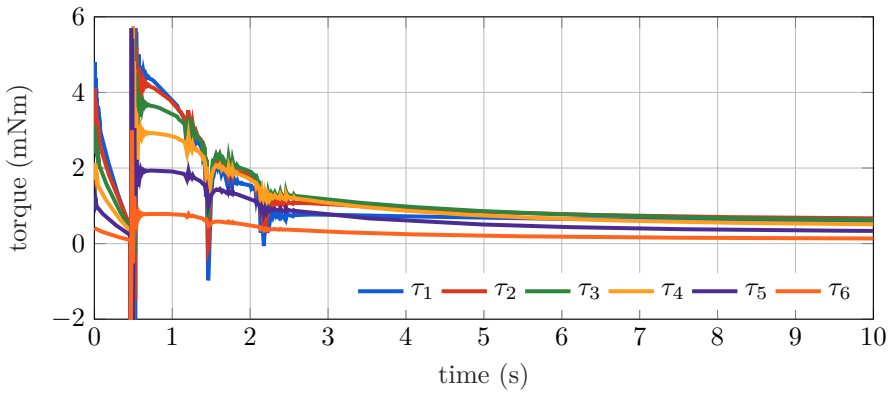


Figure 6.31. The control inputs $\boldsymbol{\tau}$ in (mN m) produced by the control law in (6.27) exhibit a significant peak at the point of contact. This is due to two factors: (*i*) the sudden change of setpoint and (*ii*) the switch in control strategy to accommodate for compliance.

`Mesh('PneuNet.png')`, which utilizes an image of a PneuNet cross-section as input. The finite element model is then formed using `fem = Fem(msh)`, and the appropriate material properties (*i.e.*, Dragonskin10) and boundary conditions are assigned. The system is subjected to a linearly increasing and decreasing pressure ramp of 40 (kPa), and the dynamics are solved using `fem.simulate`. In accordance with the procedure outlined in Section 6.5.4, a third-order DVS basis for pure bending is then constructed, which are used to construct the `Shapes` class. The curvature-bending modes are shown in Figure 6.33.

Given that the `Shapes` class has been constructed, we can now tailor the real-time estimation algorithm. This is achieved through the utilization of an inverse kinematics algorithm, as described in Section 22. The algorithm can be invoked using the function `Shapes.IK(pos, q0)`, where `pos` represents the desired position (such as a camera measurement) and `q0` is an initial estimate. The inverse kinematics solver is then repeatedly called in the real-time control loop. To ensure that the inverse kinematics solution aligns with the true system behavior, we also consider the null-space subtask projection. In this case, the gradient of the subtask is assumed to be $\nabla\Phi_{\text{sub}}(\boldsymbol{q}) = \boldsymbol{K}\boldsymbol{q} + \boldsymbol{f}_g(\boldsymbol{q}) - \boldsymbol{G}(\boldsymbol{q})u$, where $u = 30 \cdot \text{sat}\,[\sin(t)]$ (kPa) is the prescribed pressure input assigned by the open-loop controller. This subtask serves to minimize the internal residual forces – and thus is nothing more than a quasi-static deformation solver that is guided by the camera measurements. Note that model parameter have been pre-tuned to align with the experimental system presented in Figure 6.35. However, certain initial estimates of the system

parameters have been used that are produced by the Finite Element model.

To instantiate the camera class, we call `cam = Vision('realsense')` and establish a Secure Shell (SSH) connection with the control platform through `brd = Control('ip','pwd')`. Once the connection with the control platform is established, a real-time while-loop containing the necessary shape estimation algorithms to be executed. The code for this process is provided below:

```
while brd.loop(20)   % control-loop for 20 s

    % set pressure signal
    brd.setInput([Pressure(brd.t), 0]);

    % real-time visualization
    cam = cam.getFrame();
    if isempty(h)
        h = imshow(cam.lastFrame,....
                   cam.WorldLimits);
    else
        shp.FK(q,'Plot',true);
        h.CData = cam.lastFrame;
    end

    % get color marker
    cam = cam.getMarker(9);

    % inverse kinematics solver
    q = shp.IK(cam.output, q, ...
               'SubTask', @(s) gradPhi(s,p));
end
% -------------------------------------------------
function y = gradPhi(Shapes,p)
  y = Shapes.Log.EL.fe + Shapes.Log.EL.fg ...
      - Shapes.Log.EL.G * p(1);
end
% -------------------------------------------------
```

Figure 6.35 shows the experiment and overlayed on top are the Real-time shape estimations from the inverse kinematics algorithm that uses a combination with the Data-driven Variable Strain (DVS) basis. Figure 6.35 shows the state trajectories and the time instance of contact. Despite the significant impact of contact forces on the soft actuator, the shape estimation algorithm effectively reflects the behavior of the real soft robotic system. This is due to its relatively low state dimension of $\dim(\mathbf{q}) = 3$, enabling it to achieve a bandwidth of +60 Hz with ease.

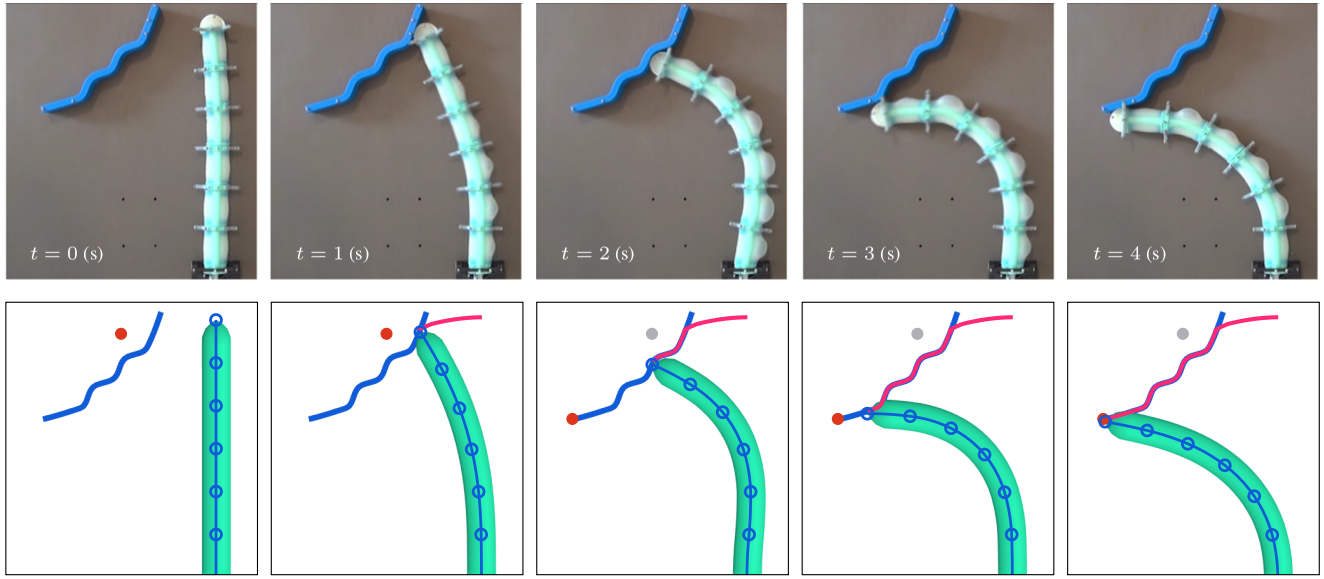


Figure 6.32. (top) The experimental results presented in the work of Della Santina et al. [**?**] show a six-link soft manipulator controlled by the control law described in (6.27). The manipulator is in contact with a static environment. The controller first ensures that the soft robot approaches the environment (Phase 1: Approach) and then follows the rigid wavy surface (Phase 2: Explore). (bottom) The numerical simulations performed using the *Sorotoki* toolkit are inspired by the work of Della Santina et al. [**?**]. In the simulation, the active setpoints are denoted by (•) and the de-activated setpoints are denoted by (•). The end-effector trajectory is shown in (—) and the environment is shown in (—). The results of the simulation truthfully reflect the approaching and surface tracking behavior as seen in the experiments.

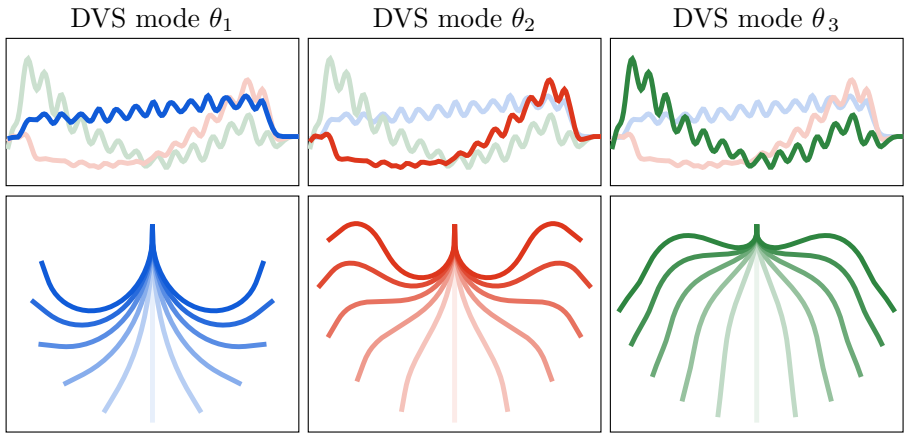


Figure 6.33. The first three DVS modes of the soft PneuNet actuator in vertical hanging condition, where the ordering is {—,—,—}. Note that these differ from the DVS basis in Section 6.5.4.

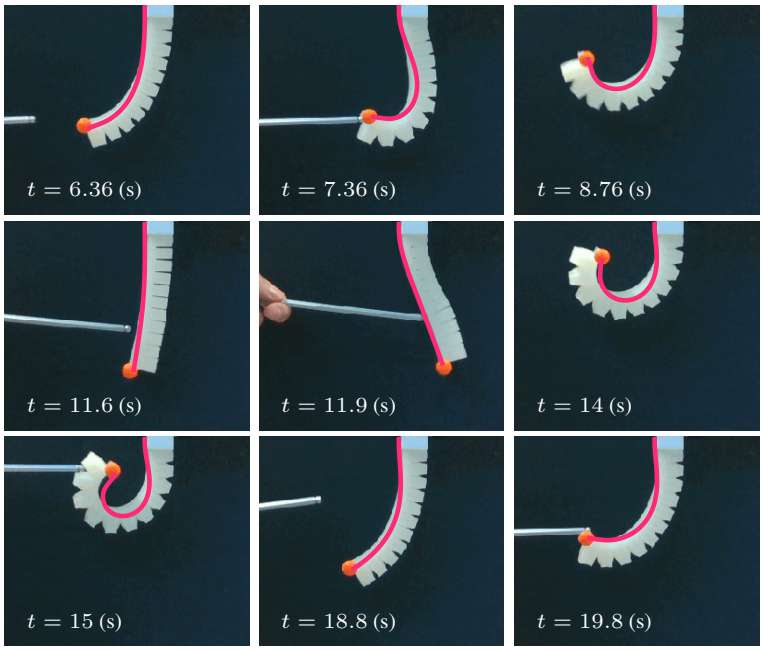


Figure 6.34. Real-time shape estimation using the `Shapes` inverse kinematics algorithm in combination with the Data-driven Variable Strain (DVS) basis. The reconstructed backbone curve is depicted in (—). Despite the presence of substantial contact forces on the soft actuator, the shape estimation algorithm accurately reflects the behavior of the real soft robotic system. Due to the relatively small state dimension of $\dim(\boldsymbol{q}) = 3$, the algorithm achieves a bandwidth of +60 Hz.

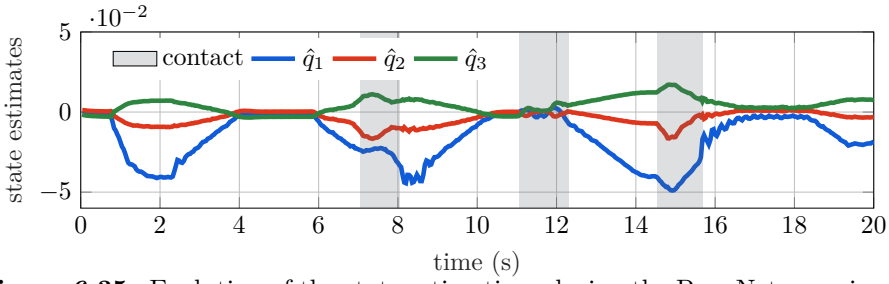


Figure 6.35. Evolution of the state estimations during the PneuNet experiment, where the ordering is given by {—, —, —}. Note that the joint variables represent the modal coefficients of the DVS basis related to curvature-bending.

6.7 Conclusion and future work

This paper has introduced the *Sorotoki* is an open-source toolkit in *MATLAB* that provides a comprehensive and modular programming environment to address the complex interdependencies associated with the design and control of soft robots. The toolkit consists of seven Object-Oriented classes that are designed to work together to solve a wide range of soft robotic problems. We believe the versatility and flexibility of *Sorotoki* make it a valuable resource for many researchers and practitioners in the field of soft robotics. The toolkit has been demonstrated to be effective through a range of case studies encompassing a broad range of issues within the field of soft robotics, including inverse design of soft actuators, passive and active soft locomotion, object manipulation with soft grippers, meta-materials, model reduction, model-based control of soft robots, and shape estimation. A unique aspect of the software package is that it does not follow the traditional linear relationship between the complexity of soft robotics systems and the length of code needed to represent them. Instead, complex system behavior can be effectively modeled using a minimal number of lines of code. The *Sorotoki* software package is particularly notable for its ability to succinctly represent complex soft robotics systems. Despite the intricacy of soft robotics, the accompanying software package is highly effective in modeling complex system behavior with minimal code, making it accessible to individuals with limited programming knowledge. Furthermore, the toolkit provides access to four open-hardware soft robotic systems that can be easily fabricated using commercially available 3D printers.

Nevertheless, the framework presents opportunities for improvement and can be expanded to make it more comprehensive, extensive, and to achieve faster simulation times. This is particularly significant in the field of control, where computational performance is a recurring challenge. The software package *Sorotoki* addresses this challenge by converting *MATLAB* code to its equivalent in `c++` using the `mex` compilation method. Real-time computation is attainable, yet the produced `.mex` functions frequently face limitations due to inadequate memory allocation and a lack of parallel processing. This can result to subpar performance. An alternative solution could be to explore languages with both a strong community and better computational performance, such as `Julia`. An early `Julia` implementation of Sorotoki is found at `github.com/BJCaasenbrood/Sorotoki.jl`, which has already shown performance improvements.

On a concluding note, we would like to emphasize that any form of contribution is greatly appreciated. Our framework is envisioned as a collaborative effort between and for the soft robotics community. Researchers who are interested in contributing are welcome to reach out to the authors. Different forms of contri-

butions are possible and more information can be found on the online repository. Additionally, while we strive to maintain the framework to be as error-free as possible, in case of exceptions during execution or any concerns regarding accuracy, please do not hesitate to contact us or through the issue tracker on the repository.

V

Closing

7

Conclusions and recommendations

7.1 Conclusions

7.2 Recommendations

In the present thesis, several research challenges have been addressed. However, as research is an ongoing endeavor, unexplored areas of research persistently emerge. In this section, we propose recommendations for further research based on the trends identified within the thesis, while taking into account recent advancements in the field of soft robotics. The author's recommendations for future research are summarized in Table **??**, categorized as short-term, medium-term, and long-term work.

VI

Appendices

A

Appendices to Chapter 4

A.1 Adjoint actions on SE(3) and se(3)

Given the position vector $\boldsymbol{\gamma} \in \mathbb{R}^3$ and the homogeneous rotation matrix $\boldsymbol{\Phi} \in \mathrm{SO}(3)$, the adjoint action of the homogeneous transformation $\boldsymbol{g} = (\boldsymbol{\gamma}, \boldsymbol{\Phi}) \in \mathrm{SE}(3)$ is then defined as

$$\mathbf{Ad}_{\boldsymbol{g}(\sigma,\boldsymbol{q})} := \begin{pmatrix} \boldsymbol{\Phi}(\sigma,\boldsymbol{q}) & \mathbf{0}_{3\times 3} \\ \boldsymbol{\gamma}^{\times}(\sigma,\boldsymbol{q})\,\boldsymbol{\Phi}(\boldsymbol{q},\sigma) & \boldsymbol{\Phi}(\sigma,\boldsymbol{q}) \end{pmatrix}. \tag{A.1}$$

Note that the operator $(\cdot)^{\times}$ denotes the isomorphism from $\mathbb{R}^3 \to \mathrm{SO}(3)$ see Murray et al. [154]. In continuation, given the velocity twist $\boldsymbol{\eta}(\sigma,\boldsymbol{q},\dot{\boldsymbol{q}}) = (\boldsymbol{\omega}^{\top}, \boldsymbol{v}^{\top})^{\top} \in \mathbb{R}^6 \cong \mathrm{se}(3)$, being the aggregate of $\boldsymbol{\omega}$ and $\boldsymbol{v}$, the angular and linear velocities, respectively. Then, the adjoint action on the algebra se(3) is defined as

$$\mathbf{ad}_{\boldsymbol{\eta}(\sigma,\boldsymbol{q},\dot{\boldsymbol{q}})} := \begin{pmatrix} \boldsymbol{v}^{\times}(\sigma,\boldsymbol{q},\dot{\boldsymbol{q}}) & \mathbf{0}_{3\times 3} \\ \boldsymbol{\omega}^{\times}(\sigma,\boldsymbol{q},\dot{\boldsymbol{q}}) & \boldsymbol{v}^{\times}(\sigma,\boldsymbol{q},\dot{\boldsymbol{q}}) \end{pmatrix}. \tag{A.2}$$

These adjoint representation on the group SE(3) and its algebra se(3) are analogous to the conventional notations in modern robotics mathematics, such as the work of Murray et al. [154]).

A.2 Passivity properties of the inertia matrix

To show $\dot{\boldsymbol{M}} - 2\boldsymbol{C}$ is skew-symmetric in the chosen coordinates, we start by computing the time-derivative of the inertia matrix. For sake of clarity, lets abbreviate $\boldsymbol{J}(\sigma, \boldsymbol{q}) = \boldsymbol{J}$ and $\dot{\boldsymbol{J}}(\sigma, \boldsymbol{q}, \dot{\boldsymbol{q}}) = \dot{\boldsymbol{J}}$. Through chain differentiation of the inertia matrix, we find

$$\dot{\boldsymbol{M}} = \int_{\mathbb{X}} \dot{\boldsymbol{J}}^{\top} \boldsymbol{\mathcal{M}} \boldsymbol{J} + \boldsymbol{J}^{\top} \boldsymbol{\mathcal{M}} \dot{\boldsymbol{J}} \, d\sigma, \tag{A.3}$$

Then, calculating $\dot{M} - 2C$ leads to

$$\dot{\boldsymbol{M}} - 2\boldsymbol{C} = \int_{\mathbb{X}} \dot{\boldsymbol{J}}^{\top} \boldsymbol{\mathcal{M}} \boldsymbol{J} - \boldsymbol{J}^{\top} \boldsymbol{\mathcal{M}} \dot{\boldsymbol{J}} - 2\boldsymbol{J}^{\top} \boldsymbol{\mathcal{C}}_{\boldsymbol{\eta}} \, \boldsymbol{J} \, d\sigma. \tag{A.4}$$

Since $\boldsymbol{J}^{\top} \boldsymbol{\mathcal{C}} \boldsymbol{J}$ is skew-symmetric, the remainder of the proof consists of showing that the matrix $S = \dot{J}^{\top} \boldsymbol{\mathcal{M}} J - J^{\top} \boldsymbol{\mathcal{M}} \dot{J}$ also satisfies skew-symmetry. Since $\boldsymbol{\mathcal{M}} = \boldsymbol{\mathcal{M}}^{\top}$, we can easily show this holds true:

$$\begin{aligned} \boldsymbol{S} &= \dot{\boldsymbol{J}}^{\top} \boldsymbol{\mathcal{M}}^{\top} \boldsymbol{J} - \boldsymbol{J}^{\top} \boldsymbol{\mathcal{M}}^{\top} \dot{\boldsymbol{J}}, \\ &= -\left(\dot{\boldsymbol{J}}^{\top} \boldsymbol{\mathcal{M}}^{\top} \boldsymbol{J} - \boldsymbol{J}^{\top} \boldsymbol{\mathcal{M}} \dot{\boldsymbol{J}} \right)^{\top} = -\boldsymbol{S}^{\top}. \end{aligned} \tag{A.5}$$

Therefore, the matrix $\dot{\boldsymbol{M}} - 2\boldsymbol{C}$ is skew-symmetric.

A.3 Implicit trapezoidal scheme for the time integration

Here, we detail an numerical approach to efficiently find the solutions to the approximated dynamic model $\boldsymbol{M}(\boldsymbol{q})\ddot{\boldsymbol{q}} + \boldsymbol{C}(\boldsymbol{q}, \dot{\boldsymbol{q}})\dot{\boldsymbol{q}} + \boldsymbol{f}_{\mathrm{e}}(\boldsymbol{q}, \dot{\boldsymbol{q}}) + \boldsymbol{f}_{\mathrm{g}}(\boldsymbol{q}) = \boldsymbol{\tau}$. We emphasize that the control input $\boldsymbol{\tau}(\cdot, t)$ could be state-dependent if closed-loop controllers are considered; for instance, the proposed passivity-based controller in (4.54). First, consider a new state vector defined as $\boldsymbol{z} := (\boldsymbol{q}^{\top}, \dot{\boldsymbol{q}}^{\top})^{\top}$ such that we can rewrite the Lagrangian model in state-space form:

$$\dot{\boldsymbol{z}} = \boldsymbol{f}(\boldsymbol{z}, t), \tag{A.6}$$

where $\boldsymbol{f}(\cdot, \cdot)$ is a nonlinear vector-valued function given by

$$\boldsymbol{f}(\boldsymbol{z}, t) = \begin{pmatrix} \dot{\boldsymbol{q}} \\ \boldsymbol{M}^{-1}[\boldsymbol{\tau} - \boldsymbol{C}\dot{\boldsymbol{q}} - \boldsymbol{f}_{\mathrm{e}} - \boldsymbol{f}_{\mathrm{g}}] \end{pmatrix}. \tag{A.7}$$

The objective here is to compute the solutions to the system above over the finite horizon $\mathbb{T} = [0, T]$ efficiently such that real-time control applications are possible. To do so, we consider an implicit trapezoidal scheme which is given by

$$\boldsymbol{z}_{i+1} = \boldsymbol{z}_i + \frac{\Delta t}{2} \left(\boldsymbol{f}(\boldsymbol{z}_i, t_i) + \boldsymbol{f}(\boldsymbol{z}_{i+1}, t_{i+1}), \right) \tag{A.8}$$

where $\boldsymbol{z}_i$ is the state solution at time instance t_i, and $\Delta t = t_{i+1} - t_i$ the timestep. The advantage of implicit schemes over explicit ones is the improved numerical stability for coarser temporal discretization at the mere cost of numerical precision. Let is be clear that evaluating nonlinear vector function $\boldsymbol{f}(\cdot, \cdot)$ is numerically expensive, as it requires the computation of $\boldsymbol{M}$, $\boldsymbol{C}$ and $\boldsymbol{f}_{\mathrm{g}}$. Therefore, it is advantageous to minimize its calls by using coarser timesteps while retaining stability using an implicit scheme. By fixing $\boldsymbol{z}_i$ and aiming to seek the intermediate state solutions $\boldsymbol{w} := \boldsymbol{z}_{i+1}$, we can define the residual dynamics on the time interval $[t_i, t_{i+1}]$ as

$$\boldsymbol{e}(\boldsymbol{w}) := \boldsymbol{w} - \boldsymbol{z}_i - \frac{\Delta t}{2} \left(\boldsymbol{f}_i + \boldsymbol{f}_{i+1}(\boldsymbol{w}) \right). \tag{A.9}$$

By aiming to find the root of the residual dynamics $e(w) = 0$ and choosing $\boldsymbol{w}_0 = \boldsymbol{z}_i$ as initial guess, we can employ an iterative Newton-Raphson procedure:

$$\boldsymbol{w}_{j+1} = \boldsymbol{z}_i - \alpha^+ \left[\nabla_{\boldsymbol{w}} \, \boldsymbol{e}(\boldsymbol{w}_j) \right]^{-1} \boldsymbol{e}(\boldsymbol{w}_j), \tag{A.10}$$

where j is the iteration index for finding the intermediate state solution $\boldsymbol{w} = \boldsymbol{z}_{i+1}$, and $0 < \alpha^+ \leq 1$ a constant for controlling the update step. Once the residual dynamics converges on the sub-interval, i.e., $||\boldsymbol{e}(\boldsymbol{w})||_2 \ll 1$, we repeat the procedure above until the solutions to $\boldsymbol{z}(t)$ are recovered for the finite time horizon $\mathbb{T}$. Now the key here is that a rough approximation of the hessian $\boldsymbol{H}(\boldsymbol{w}) = \nabla_{\boldsymbol{w}} \, \boldsymbol{e}$ can suffice for numerical convergence. This implies we do not need a close approximation of hessian – which can significantly speed-up simulation speed. This process, however, might require more iterations for the solutions to converge, but it outweighs computing the Hessian directly. Therefore, let us consider the first Taylor approximation of the Hessian:

$$\begin{aligned} \boldsymbol{H}(\boldsymbol{w}) &\simeq \widetilde{\boldsymbol{H}}(\boldsymbol{w}) \\ &:= \boldsymbol{I}_{2n} - \frac{\Delta t}{2} \begin{pmatrix} 0 & \boldsymbol{I}_n, \\ -\boldsymbol{M}^{-1} \widetilde{\boldsymbol{K}} & -\boldsymbol{M}^{-1} \widetilde{\boldsymbol{D}} \end{pmatrix}, \end{aligned} \tag{A.11}$$

where the matrices $\widetilde{\boldsymbol{K}} = \nabla_{\boldsymbol{q}} \, \boldsymbol{f}_{\mathrm{e}} + \nabla_{\boldsymbol{q}} \, \boldsymbol{f}_{\mathrm{g}} + \nabla_{\boldsymbol{q}} \, \boldsymbol{\tau}$ and $\widetilde{\boldsymbol{D}} = \boldsymbol{C} + \nabla_{\dot{\boldsymbol{q}}} \, \boldsymbol{f}_{\mathrm{e}} + \nabla_{\dot{\boldsymbol{q}}} \, \boldsymbol{\tau}$ are a-priori approximations of the Hessians w.r.t. $\boldsymbol{q}$ and $\dot{\boldsymbol{q}}$, respectively. Note that

the Jacobians $\nabla_{\boldsymbol{q}} \boldsymbol{\tau}$ and $\nabla_{\dot{\boldsymbol{q}}} \boldsymbol{\tau}$ can be nonzero, especially in a closed-loop control setting. To approximate these partial derivatives of the control input, we employ a finite-difference scheme. Again, the Hessian does not need to be exact, as such an a-priori computation of the controller Jacobians can be performed before the start of the implicit solver.

B

Appendices to Chapter 5

B.1 Introduction to Lie group and Lie algebra

Here we briefly discuss some notation and basic operations on the Lie groups SO(3) and SE(3), and their respective Lie algebras so(3) and se(3). This appendix serves a compact comprehensive introduction in the context of robotics, and the appendix is based on the comprehensive work of Bullo and Murray (1995, [32]).

B.1.1 Basic definition(s)

Here, we focus first on the (Lie) group of rigid-body transformations about the origin of $\mathbb{R}^3$ – denoted by SE(3). Let $G \subset \mathrm{SE}(3)$ be a matrix Lie group and its respective algebra $\mathfrak{g} \in \mathrm{se}(3)$ its Lie Algebra. Then, the evolution of a general rigid body under motion with pose $\boldsymbol{g} \in G$ can be described using

$$\dot{\boldsymbol{g}} = \boldsymbol{g}\,\hat{\boldsymbol{\eta}}^{\mathrm{b}} \iff \dot{\boldsymbol{g}} = \hat{\boldsymbol{\eta}}^{\mathrm{s}}\,\boldsymbol{g}. \quad \hat{\boldsymbol{\eta}}^{\mathrm{b}}, \hat{\boldsymbol{\eta}}^{\mathrm{s}} \in \mathfrak{g} \tag{B.1}$$

where the velocity twist relative its body frame is given by $\hat{\boldsymbol{\eta}}^{\mathrm{b}}$ or to a spatial frame by $\hat{\boldsymbol{\eta}}^{\mathrm{s}}$. Since $\dot{\boldsymbol{g}} = \boldsymbol{g}\hat{\boldsymbol{\eta}}^{\mathrm{b}}$ is invariant under left multiplication by constant matrices, we call it *left invariant*; and correspondingly $\dot{\boldsymbol{g}} = \hat{\boldsymbol{\eta}}^{\mathrm{s}}\boldsymbol{g}$ is said to be *right invariant*. Let it be clear that the geometric strain $\boldsymbol{\xi}(\sigma, t)$ and velocity twist $\boldsymbol{\eta}(\sigma, t)$ in (5.2) and (5.3) are thus expressed in the body frame. Next, let us discuss the adjoint actions. For all $\boldsymbol{g} \in G$ and any $\boldsymbol{X}, \boldsymbol{Y} \in \mathfrak{g}$, the adjoint action $\mathbf{Ad}_{\boldsymbol{g}}$ and the matrix

commutator or adjoint action on the algebra $\mathbf{ad}_{\boldsymbol{X}}$ are defined as

$$\mathbf{Ad}_{\boldsymbol{g}}\boldsymbol{X} = \boldsymbol{g}\boldsymbol{X}\boldsymbol{g}^{-1} \tag{B.2}$$

$$\mathbf{ad}_{\boldsymbol{X}}\boldsymbol{Y} = [\boldsymbol{X},\boldsymbol{Y}] = \boldsymbol{X}\boldsymbol{Y} - \boldsymbol{Y}\boldsymbol{X}. \tag{B.3}$$

Now, on SE(3) and se(3) we represent a matrix group element $\boldsymbol{g} = (\boldsymbol{\Phi}, \boldsymbol{\gamma}) \in \mathrm{SO}(3) \times \mathbb{R}^3 \cong \mathrm{SE}(3)$ and a (velocity) twist vector field $\hat{\boldsymbol{\eta}} = (\boldsymbol{\Omega}, \boldsymbol{V}) \in \mathrm{se}(3)$ using homogenous coordinates,

$$\boldsymbol{g} := \begin{pmatrix} \boldsymbol{\Phi} & \boldsymbol{\gamma} \\ \mathbf{0}_3^\top & 1 \end{pmatrix}; \quad \hat{\boldsymbol{\eta}} := \begin{pmatrix} \boldsymbol{\Omega}^\times & \boldsymbol{V} \\ \mathbf{0}_3^\top & 1 \end{pmatrix}. \tag{B.4}$$

where the operator $(\cdot)^\times : \mathbb{R}^3 \to \mathrm{so}(3)$ is defined such that $\boldsymbol{x}^\times \boldsymbol{y} = \boldsymbol{x} \times \boldsymbol{y}$ for all $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^3$, and $(\cdot)^\wedge : \mathbb{R}^6 \to \mathrm{se}(3)$. Now, representing the geometric twist as a column vector $\boldsymbol{\eta}^\wedge \to \boldsymbol{\eta}$, it straightforwardly follows from (B.2) and (B.3) that the adjoint actions can be written in the form:

$$\mathbf{Ad}_{\boldsymbol{g}} := \begin{pmatrix} \boldsymbol{\Phi} & \mathbf{0}_{3\times 3} \\ \boldsymbol{\gamma}^\times \boldsymbol{\Phi} & \boldsymbol{\Phi} \end{pmatrix}; \quad \mathbf{ad}_{\boldsymbol{\eta}} := \begin{pmatrix} \boldsymbol{V}^\times & \mathbf{0}_{3\times 3} \\ \boldsymbol{\Omega}^\times & \boldsymbol{V}^\times \end{pmatrix}. \tag{B.5}$$

The notation above are analogous to the notations used in Chapter 3.

B.1.2 Exponential and logarithmic map

An important operation in Lie group theory, is the exponential and logarithmic maps that serve as transformations between the groups and their respective algebras. Lets start with the exponential map. Given $\boldsymbol{\Omega}^\times \in \mathrm{so}(3)$ and $\boldsymbol{\eta} = (\boldsymbol{\Omega}, \boldsymbol{V}) \in \mathrm{se}(3)$, the exponential maps for the orientation group $\exp_{\mathrm{SO}(3)} : \mathrm{so}(3) \to \mathrm{SO}(3)$ and the rigid-body transformation group $\exp_{\mathrm{SO}(3)} : \mathrm{se}(3) \to \mathrm{SE}(3)$ are given respectively by

$$\exp_{\mathrm{SO}(3)}(\boldsymbol{\Omega}) = \boldsymbol{I} + \sin\|\boldsymbol{\Omega}\| \frac{\boldsymbol{\Omega}^\times}{\|\boldsymbol{\Omega}\|} + (1 - \cos\|\boldsymbol{\Omega}\|) \frac{\boldsymbol{\Omega}^\times \boldsymbol{\Omega}^\times}{\|\boldsymbol{\Omega}\|^2}; \tag{B.6}$$

$$\exp_{\mathrm{SE}(3)}(\boldsymbol{\xi}) = \begin{pmatrix} \exp_{\mathrm{SO}(3)}(\boldsymbol{\Omega}) & A(\boldsymbol{\Omega})\boldsymbol{V} \\ \mathbf{0}_3^\top & 1 \end{pmatrix}, \tag{B.7}$$

where $\|\cdot\|$ stands for the Euclidean norm and the operator $A(\boldsymbol{\Omega})$ is given By

$$A(\boldsymbol{\Omega}) = \boldsymbol{I} + \left(\frac{1 - \|\boldsymbol{\Omega}\|}{\|\boldsymbol{\Omega}\|}\right) \frac{\boldsymbol{\Omega}^\times}{\|\boldsymbol{\Omega}\|} + (1 - \sin\|\boldsymbol{\Omega}\|) \frac{\boldsymbol{\Omega}^\times \boldsymbol{\Omega}^\times}{\|\boldsymbol{\Omega}\|^2}. \tag{B.8}$$

Note that we have seen equation (B.6) earlier which is known as the Rogdrigues' formula. Referring to [32], in an open neighborhood of the origin in G, we define

$\boldsymbol{\eta} = \log(\boldsymbol{g}) \in \mathfrak{g}$ to be the *exponential coordinates* of the group element $\boldsymbol{g}$. Then, the logarithmic map can be regarded as the local chart of the manifold G. As such, let $\boldsymbol{g} = (\boldsymbol{\Phi}, \boldsymbol{\gamma}) \in \mathrm{SO}(3) \times \mathbb{R}^3$ be such that $\mathrm{trace}(\boldsymbol{\Phi}) \neq -1$. Then, the logarithmic map $\log_{SO3} : \mathrm{SO}(3) \to \mathrm{so}(3)$ is given by

$$\log_{\mathrm{SO}(3)}(\boldsymbol{\Phi}) = \frac{\theta}{2\sin\theta}(\boldsymbol{\Phi} - \boldsymbol{\Phi}^\top), \tag{B.9}$$

where the angle θ satisfies $\cos\theta = \frac{1}{2}(\mathrm{trace}(\boldsymbol{\Phi}) - 1)$ and is bounded by $|\theta| < \pi$. Following (B.9), and logarithmic map $\log_{SE3} : \mathrm{SE}(3) \to \mathrm{se}(3)$ is then given by

$$\log_{\mathrm{SE}(3)}(\boldsymbol{g}) = \begin{pmatrix} \boldsymbol{\Omega}^\times & A^{-1}(\boldsymbol{\Omega})\boldsymbol{\gamma} \\ \mathbf{0}_3^\top & 1 \end{pmatrix}, \tag{B.10}$$

where $\boldsymbol{\Omega}^\times = \log_{\mathrm{SO}(3)}(\boldsymbol{\Phi})$ and the mapping $A^{-1}(\boldsymbol{\Omega})$ is given By

$$A^{-1}(\boldsymbol{\Omega}) = \boldsymbol{I} - \frac{1}{2}\boldsymbol{\Omega}^\times + \left[1 - \frac{\|\boldsymbol{\Omega}\|}{2}\cot\left(\frac{\|\boldsymbol{\Omega}\|}{2}\right)\right]\frac{\boldsymbol{\Omega}^\times\boldsymbol{\Omega}^\times}{\|\boldsymbol{\Omega}\|^2}, \tag{B.11}$$

where $\cot : \mathbb{R} \to \mathbb{R}$ is the co-tangent function.

B.2 Time-derivate of the geometric Jacobian for continuum robots

The mapping from generalized coordinates $\dot{\boldsymbol{q}} \in \mathbb{R}^n$ to the velocity-twist vector $\hat{\boldsymbol{\eta}} = \boldsymbol{g}^{-1}\dot{\boldsymbol{g}} \in \mathrm{se}(3) \cong \mathbb{R}^6$ for a point σ be given by $\boldsymbol{\eta} = \boldsymbol{J}\dot{\boldsymbol{q}}$ where $\boldsymbol{J}$ is the geometric Jacobian. The k-th order truncations of the exact geometric Jacobian is given by

$$[\boldsymbol{J}]_k = \mathbf{Ad}^{-1}_{[\boldsymbol{g}]_k}\int_0^\sigma \mathbf{Ad}_{[\boldsymbol{g}]_k}\boldsymbol{\Theta}\, ds. \tag{B.12}$$

Unlike its notation in rigid robotics, note that the geometric Jacobian matrix here is time and space-variant. Following the chain rule of differentiation, the partial time-derivative of the geometric Jacobian matrix yields

$$[\dot{\boldsymbol{J}}]_k = \left(\dot{\mathbf{Ad}^{-1}_{[\boldsymbol{g}]_k}}\right)\int_0^\sigma \mathbf{Ad}_{[g]_k}\boldsymbol{\Theta}\, ds + \mathbf{Ad}^{-1}_{[\boldsymbol{g}]_k}\int_0^\sigma \left(\dot{\mathbf{Ad}_{[g]_k}}\right)\boldsymbol{\Theta}\, ds. \tag{B.13}$$

Given the differential relations of the adjoint action mapping on the Lie group, that is, $d/ds\,(\mathbf{Ad}_{\boldsymbol{g}}) = \mathbf{Ad}_{\boldsymbol{g}}\mathbf{ad}_{\boldsymbol{\Upsilon}}$ given a twist $\boldsymbol{\Upsilon} = (\boldsymbol{g}^{-1}d\boldsymbol{g}/ds)^\vee$, we can express the time-derivate of the adjoint action and its inverse as

$$\frac{\partial}{\partial t}\left(\mathbf{Ad}_{\boldsymbol{g}}\right) = \mathbf{Ad}_{\boldsymbol{g}}\mathbf{ad}_{\boldsymbol{\eta}}, \tag{B.14}$$

$$\frac{\partial}{\partial t}\left(\mathbf{Ad}_{\boldsymbol{g}^{-1}}\right) = -\mathbf{ad}_{\boldsymbol{\eta}}\mathbf{Ad}_{\boldsymbol{g}^{-1}}. \tag{B.15}$$

Substituting the truncated variations of (B.14) and (B.15) into (B.13), we find the complete expression of the time-derivate of the geometric Jacobian matrix

$$[\dot{\boldsymbol{J}}]_k = -\mathbf{ad}_{\boldsymbol{\eta}}[\boldsymbol{J}]_k + \mathbf{Ad}^{-1}_{[\boldsymbol{g}]_k} \int_0^{\sigma} \mathbf{Ad}_{[\boldsymbol{g}]_k} \mathbf{ad}_{[\boldsymbol{\eta}]_k} \boldsymbol{\Theta}\, ds. \tag{B.16}$$

Since $\mathbf{ad}_{\boldsymbol{\eta}}(\boldsymbol{J}\dot{\boldsymbol{q}}) = \mathbf{ad}_{\boldsymbol{\eta}}\boldsymbol{\eta} = \mathbf{0}_6$ by definition, the first right-hand term vanishes if (B.16) is post-multiplied with the generalized velocities $\dot{\boldsymbol{q}}$, thus leading to the acceleration twist $[\dot{\boldsymbol{\eta}}]_k$ in (5.18).

C

Appendices to Chapter 6

C.1 Newmark-β solver

The Newmark-β method is an implicit numerical integration scheme extensively used to solve high-dimensional structural dynamic problem [95**?**]. We briefly explain the algorithm implemented in the function `Fem.simulate`. First, let us subdivide the time domain such that $(0, ..., T)$ with uniform timesteps $\Delta t = t_{i+1} - t_i$. Then, given the initial conditions for (6.3), we wish to compute the state evolution $\boldsymbol{x}(t_i)$ and $\dot{\boldsymbol{x}}(t_i)$. For conciseness, let us write the discrete states of the FEM model as $\boldsymbol{x}(t_i) = \boldsymbol{x}^{(i)}$. Through the extended mean value theorem, we can formulate the general Newmark-β scheme as

$$\dot{\boldsymbol{x}}^{(i+1)} = \dot{\boldsymbol{x}}^{(i)} + \Delta t \left[(1 - \beta_1)\ddot{\boldsymbol{x}}^{(i)} + \Delta t \beta_1 \ddot{\boldsymbol{x}}^{(i+1)} \right], \tag{C.1}$$

$$\boldsymbol{x}^{(i+1)} = \boldsymbol{x}^{(i)} + \Delta t \left[\dot{\boldsymbol{x}}^{(i)} + (\tfrac{1}{2} - \beta_2)\ddot{\boldsymbol{x}}^{(i)} + \beta_2 \ddot{\boldsymbol{x}}^{(i+1)} \right], \tag{C.2}$$

where $\beta_1, \beta_2 \geq \frac{1}{2}$. Now, in the expressions above only the forward-time acceleration $\ddot{\boldsymbol{x}}^{(i+1)}$ is the unknown partial solution, hence we conveniently write $\boldsymbol{w} := \ddot{\boldsymbol{x}}^{(i+1)}$. Substituting these into the dynamic flow (6.3), we retrieve a residual expression:

$$\boldsymbol{r}(\boldsymbol{w}) := \boldsymbol{w} + \nabla_{\boldsymbol{x}} \mathcal{H}(\boldsymbol{w}) + \boldsymbol{R} \nabla_{\boldsymbol{\mu}} \mathcal{H}(\boldsymbol{w}) - \boldsymbol{G}\boldsymbol{u}^{(i+1)}. \tag{C.3}$$

Following, the residual vector (C.3) forms an optimization problem in the form $\text{argmin}_{\boldsymbol{w}} \|\boldsymbol{r}(\boldsymbol{w})\|_2$ for unknown accelerations $\boldsymbol{w}$. This implicit relation can be solved

numerically using a recursive Newton Raphson method. Givin the n-th iteration, the recursive solver reads

$$\boldsymbol{w}^{(n+1)} = \boldsymbol{w}^{(n)} + \alpha_+ \left[\boldsymbol{A}(\boldsymbol{w}^{(n)})\right]^{-1} \boldsymbol{r}(\boldsymbol{w}^{(n)}), \tag{C.4}$$

where $\boldsymbol{A} := \left[\boldsymbol{I} + \beta_1 \Delta t \boldsymbol{R} + \beta_2 \Delta t^2 \boldsymbol{K}_T \boldsymbol{M}\right]$ is the hessian matrix, and $0 < \alpha_+ \leq 1$ an update coefficient. The matrix $\boldsymbol{K}_T$ denotes the tangent stiffness related to the local gradient of the elasticity force, given by $\boldsymbol{K}_T := \nabla_{\boldsymbol{x}} \boldsymbol{f}_{\mathrm{e}}$.

Bibliography

[1] PUMA. The Leading Edge in Robotic Technology. [Online; accessed 15. Sep. 2022], url: https://digital.hagley.org/islandora/object/islandora%3A2650820.

[2] The Stanford Arm, Computer History Museum. [Online; accessed 15. Sep. 2022], url: https://www.computerhistory.org/collections/catalog/102723508.

[3] Albu-Schäffer, A., Eiberger, O., Fuchs, M., Grebenstein, M., Haddadin, S., Ott, C., Stemmer, A., Wimböck, T., Wolf, S., Borst, C., and Hirzinger, G. (2011). Anthropomorphic Soft Robotics – From Torque Control to Variable Intrinsic Compliance. In *Robotics Research*, pages 185–207. Springer, Berlin, Germany.

[4] Albu-Schaffer, A., Fischer, M., Schreiber, G., Schoeppe, F., and Hirzinger, G. (2004). Soft robotics: what Cartesian stiffness can obtain with passively compliant, uncoupled joints? In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 4, pages 3295–3301. IEEE.

[5] Alora, J. I., Cenedese, M., Schmerling, E., Haller, G., and Pavone, M. (2022). Data-Driven Spectral Submanifold Reduction for Nonlinear Optimal Control of High-Dimensional Robots. *ArXiv e-prints*.

[6] An, X., Cui, Y., Sun, H., Shao, Q., and Zhao, H. (2023). Active-Cooling-in-the-Loop Controller Design and Implementation for an SMA-Driven Soft Robotic Tentacle. *IEEE Transactions on Robotics*, pages 1–17.

[7] Anderson, R. J. and Spong, M. W. (1988). Hybrid impedance control of robotic manipulators. *IEEE Journal on Robotics and Automation*, 4(5):549–556.

[8] Anderson, V. and Horn, C. (1968). Tensor arm manipulator. US Patent No.3,497,083.

[9] Andorf, P., Franz, D., Lieb, A., Upper, G., Guttropf, W., and K. G., C. F. (1974). Robot finger. US Patent No.3,981,528.

[10] Armanini, C., Boyer, F., Mathew, A. T., Duriez, C., and Renda, F. (2023). Soft Robots Modeling: A Structured Overview. *IEEE Transactions on Robotics*, pages 1–21.

[11] Asbeck, A. T., Schmidt, K., and Walsh, C. J. (2015). Soft exosuit for hip assistance. *Robotics and Autonomous Systems*, 73:102–110.

[12] Astrid, P., Weiland, S., Willcox, K., and Backx, T. (2008). Missing point estimation in models described by proper orthogonal decomposition. *IEEE Transactions on Automatic Control*, 53(10):2237–2251.

[13] Atekson, C. (2022). Big Hero 6: Let's Build Baymax. [Online; accessed 15. Sep. 2022], url: http://www.cs.cmu.edu/~cga/bighero6.

[14] Baaij, T., Holkenborg, M. K., Stölzle, M., van der Tuin, D., Naaktgeboren, J., Babuška, R., and Santina, C. D. (2023). Learning 3D shape proprioception for continuum soft robots with multiple magnetic sensors. *Soft Matter*, 19(1):44–56.

[15] Bartlett, N. W., Tolley, M. T., Overvelde, J. T. B., Weaver, J. C., Mosadegh, B., Bertoldi, K., Whitesides, G. M., and Wood, R. J. (2015a). A 3D-printed, functionally graded soft robot powered by combustion. *Science*, 349(6244):161–165.

[16] Bartlett, N. W., Tolley, M. T., Overvelde, J. T. B., Weaver, J. C., Mosadegh, B., Bertoldi, K., Whitesides, G. M., and Wood, R. J. (2015b). A 3D-printed, functionally graded soft robot powered by combustion. *Science*, 349(6244):161–165.

[17] Bendsøe, M. P. and Sigmund, O. (2003). *Topology optimization: theory, methods, and applications*, volume 2nd Editio.

[18] Benner, P., Sachs, E., and Volkwein, S. (2014). Model order reduction for PDE constrained optimization. *International Series of Numerical Mathematics*, 165:303–326.

[19] Bern, J., Banzet, P., Poranne, R., and Coros, S. (2019). Trajectory Optimization for Cable-Driven Soft Robot Locomotion. *Proceedings of Robotics: Science and Systems*.

[20] Bern, J. M., Chang, K.-H., and Coros, S. (2022). Interactive design of animated plushies. *ACM Transactions on Graphics*, 36(4):1–11.

[21] Bern, J. M. and Rus, D. (2021). Soft IK with Stiffness Control. In *2021 IEEE 4th International Conference on Soft Robotics (RoboSoft)*, pages 465–471. IEEE.

[22] Berthet-Rayne, P., Sadati, S. M. H., Petrou, G., Patel, N., Giannarou, S., Leff, D. R., and Bergeles, C. (2021). MAMMOBOT: A Miniature Steerable Soft Growing Robot for Early Breast Cancer Detection. *IEEE Rob. Autom. Lett.*, 6(3):5056–5063.

[23] Bertoldi, K., Boyce, M. C., Deschanel, S., Prange, S. M., and Mullin, T. (2008). Mechanics of deformation-triggered pattern transformations and superelastic behavior in periodic elastomeric structures. *Journal of the Mechanics and Physics of Solids*, 56(8):2642–2668.

[24] Bhatia, J. S., Jackson, H., Tian, Y., Xu, J., and Matusik, W. (2022). Evolution Gym: A Large-Scale Benchmark for Evolving Soft Robots. *ArXiv e-prints*.

[25] Bishop, R. L. (1975). There is More than One Way to Frame a Curve. *Am. Math. Mon.*, 82(3):246–251.

[26] Borja, P., Dabiri, A., and Santina, C. D. (2022). Energy-based shape regulation of soft robots with unactuated dynamics dominated by elasticity. In *2022 IEEE 5th International Conference on Soft Robotics (RoboSoft)*, pages 396–402. IEEE.

[27] Boyer, F., Lebastard, V., Candelier, F., and Renda, F. (2021). Dynamics of Continuum and Soft Robots: A Strain Parameterization Based Approach. *IEEE Transactions on Robotics*, 37(3):847–863.

[28] Boyer, F., Porez, M., and Leroyer, A. (2010). Poincaré–Cosserat Equations for the Lighthill Three-dimensional Large Amplitude Elongated Body Theory: Application to Robotics. *Journal of Nonlinear Science*, 20(1):47–79.

[29] Boyvat, M., Koh, J.-S., and Wood, R. J. (2017). Addressable wireless actuation for multijoint folding robots and devices. *Science Robotics*, 2(8):eaan1544.

[30] Brock, R. and Stokes, A. (1991). On the synthesis of compliant mechanisms. In *IEEE Conference on Robotics and Automation*, pages 2168–2173.

[31] Bruder, D., Fu, X., Gillespie, R. B., Remy, C. D., and Vasudevan, R. (2021). Data-driven control of soft robots using koopman operator theory. *IEEE Transactions on Robotics*, 37(3):948–961.

[32] Bullo, F. and Murray, R. M. (1995). Proportional Derivative (PD) Control on the Euclidean Group. *California Institute of Technology*.

[33] Caasenbrood, B. (2020). SOROTOKI - an open-source soft robotics toolkit for matlab. GitHub.

[34] Caasenbrood, B., Pogromsky, A., and Nijmeijer, H. (2020). Dynamic modeling of hyper-elastic soft robots using spatial curves. *IFAC Proceedings Volumes (IFAC-PapersOnline)*.

[35] Caasenbrood, B., Pogromsky, A., and Nijmeijer, H. (2022). Control-Oriented Models for Hyperelastic Soft Robots Through Differential Geometry of Curves. *Soft Robotics*, 0(0).

[36] Caasenbrood, B., Pogromsky, A. Y., and Nijmeijer, H. (2021). Energy-based control for soft manipulators using cosserat-beam models. In *International Conference on Informatics in Control, Automation and Robotics - ICINCO*, pages 311–319.

[37] Cao, G., Huo, B., Yang, L., Zhang, F., Liu, Y., and Bian, G. (2021). Model-Based Robust Tracking Control Without Observers for Soft Bending Actuators. *IEEE Robotics and Automation Letters*, 6(3):5175–5182.

[38] Chang, H.-S., Halder, U., Shih, C.-H., Naughton, N., Gazzola, M., and Mehta, P. G. (2022). Energy Shaping Control of a Muscular Octopus Arm Moving in Three Dimensions. *ArXiv e-prints*.

[39] Chang, H.-S., Halder, U., Shih, C.-H., Naughton, N., Gazzola, M., and Mehta, P. G. (2023). Energy-shaping control of a muscular octopus arm moving in three dimensions. *Proceedings of the Royal Society A*.

[40] Cheney, N., MacCurdy, R., Clune, J., and Lipson, H. (2013). Unshackling evolution: evolving soft robots with multiple materials and a powerful generative encoding. In *GECCO '13: Proceedings of the 15th annual conference on Genetic and evolutionary computation*, pages 167–174. Association for Computing Machinery, New York, NY, USA.

[41] Chirikjian, G. and Burdick, J. (1989). A Modal Approach to the Kinematics of Hyper-Redudant Manipulators. California Insitute of Technology, Pasadena.

[42] Chirikjian, G. S. (1992). *Theory and applications of hyper-redundant robotic manipulators*. PhD thesis.

[43] Chirikjian, G. S. and Burdick, J. W. (1991). Kinematics of Hyper-Redundant Manipulators. In *Advances in Robot Kinematics*, pages 392–399. Springer, Vienna, Wien, Austria.

[44] Chirikjian, G. S. and Burdick, J. W. (1992). A Geometric Approach to Hyper-Redundant Manipulator Obstacle Avoidance. *Journal of Mechanical Design*, 114(4):580–585.

[45] Chirikjian, G. S. and Burdick, J. W. (1994a). A modal approach to hyper-redundant manipulator kinematics. *IEEE Transactions on Robotics and Automation*, 10(3):343–354.

[46] Chirikjian, G. S. and Burdick, J. W. (1994b). A hyper-redundant manipulator. *IEEE Robotics Automation Magazine*, 1:22–29.

[47] Choi, W., Whitesides, G. M., Wang, M., Chen, X., Shepherd, R. F., Mazzeo, A. D., Morin, S. A., Stokes, A. A., and Ilievski, F. (2011). Multigait soft robot. *Proceedings of the National Academy of Sciences*, 108(51):20400–20403.

[48] Cianchetti, M., Ranzani, T., Gerboni, G., De Falco, I., Laschi, C., and Menciassi, A. (2013). STIFF-FLOP surgical manipulator: Mechanical design and experimental characterization of the single module. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3576–3581. IEEE.

[49] Cianchetti, M., Ranzani, T., Gerboni, G., Nanayakkara, T., Althoefer, K., Dasgupta, P., and Menciassi, A. (2014). Soft Robotics Technologies to Address Shortcomings in Today's Minimally Invasive Surgery: The STIFF-FLOP Approach. *Soft Robotics*, 1(2):122–131.

[50] Coevoet, E., Escande, A., and Duriez, C. (2017a). Optimization-Based Inverse Model of Soft Robots With Contact Handling. *IEEE Robotics and Automation Letters*, 2(3):1413–1419.

[51] Coevoet, E., Morales-Bieze, T., Largilliere, F., Zhang, Z., Thieffry, M., Sanz-Lopez, M., Carrez, B., Marchal, D., Goury, O., Dequidt, J., and Duriez, C. (2017b). Software toolkit for modeling, simulation, and control of soft robots. *Advanced Robotics*, 31(22):1208–1224.

[52] Corke, P. (2011). *Robotics, Vision and Control*. Springer, Berlin, Germany.

[53] Corke, P. (2020). Why Robots are Important. *Age of Robots*, 2:122–126.

[54] Daerden, F. (1999). *Conception and Realization of Pleated Pneumatic Artificial Muscles and their Use as Compliant Actuation Elements*. PhD thesis.

[55] Daerden, F. and Lefeber, D. (2000). Pneumatic artificial muscles: actuators for robotics and automation. *European journal of Mechanical and Environmental Engineering*, 47:10–21.

[56] De Luca, A. and Book, W. J. (2016). Robots with Flexible Elements. In *Springer Handbook of Robotics*, pages 243–282. Springer, Cham, Switzerland.

[57] Della Santina, C., Catalano, M. G., and Bicchi, A. (2020). Soft Robots. In *Encyclopedia of Robotics*, pages 1–15. Springer, Berlin, Germany.

[58] Della Santina, C., Duriez, C., and Rus, D. (2021). Model Based Control of Soft Robots: A Survey of the State of the Art and Open Challenges. pages 1–69.

[59] Della Santina, C., Katzschmann, R. K., Bicchi, A., and Rus, D. (2020). Model-based dynamic feedback control of a planar soft robot: trajectory tracking and interaction with the environment. *International Journal of Robotics Research*, 39(4):490–513.

[60] Della Santina, C. and Rus, D. (2020). Control oriented modeling of soft robots: The polynomial curvature case. *IEEE Robotics and Automation Letters*, 5(2):290–298.

[61] Drotman, D., Jadhav, S., Karimi, M., de Zonia, P., and Tolley, M. T. (2017). 3D printed soft actuators for a legged robot capable of navigating unstructured terrain. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5532–5538. IEEE.

[62] Duriez, C. (2013). Control of elastic soft robots based on real-time finite element method. pages 3982–3987. Proceedings - IEEE International Conference on Robotics and Automation.

[63] Duriez, C., Coevoet, E., Largilliere, F., Morales-Bieze, T., Zhang, Z., Sanz-Lopez, M., Carrez, B., Marchal, D., Goury, O., and Dequidt, J. (2016). Framework for online simulation of soft robots with optimization-based inverse model. In *2016 IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR)*, pages 111–118. IEEE.

[64] Falkenhahn, V., Hildebrandt, A., Neumann, R., and Sawodny, O. (2015a). Model-based feedforward position control of constant curvature continuum robots using feedback linearization. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 762–767. IEEE.

[65] Falkenhahn, V., Mahl, T., Hildebrandt, A., Neumann, R., and Sawodny, O. (2015b). Dynamic Modeling of Bellows-Actuated Continuum Robots Using the Euler-Lagrange Formalism. *IEEE Transactions on Robotics*, 31(6):1483–1496.

[66] Fang, G., Tian, Y., Yang, Z.-X., Geraedts, J. M. P., and Wang, C. C. L. (2020). Efficient Jacobian-Based Inverse Kinematics of Soft Robots by Learning. *ArXiv e-prints*.

[67] Fischer, O., Toshimitsu, Y., Kazemipour, A., and Katzschmann, R. K. (2022). Dynamic Task Space Control Enables Soft Manipulators to Perform Real-World Tasks. *Advanced Intelligent Systems*.

[68] Franco, E. (2022). Model-Based Eversion Control of Soft Growing Robots With Pneumatic Actuation. *IEEE Control Systems Letters*, 6:2689–2694.

[69] Franco, E. and Astolfi, A. (2022). Energy shaping control of underactuated mechanical systems with fluidic actuation. *International Journal of Robust and Nonlinear Control*, 32(18):10011–10028.

[70] Franco, E. and Garriga-Casanovas, A. (2020). Energy-shaping control of soft continuum manipulators with in-plane disturbances. *International Journal of Robotics Research*.

[71] Franco, E., Tang, J., Casanovas, A. G., Y Baena, F. R., and Astolfi, A. (2020). Position Control of Soft Manipulators with Dynamic and Kinematic Uncertainties. *IFAC-PapersOnLine*, 53(2):9847–9852.

[72] Fras, J. and Althoefer, K. (2018). Soft Biomimetic Prosthetic Hand: Design, Manufacturing and Preliminary Examination. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–6. IEEE.

[73] Galloway, K. C., Becker, K. P., Phillips, B., Kirby, J., Licht, S., Tchernov, D., Wood, R. J., and Gruber, D. F. (2016). Soft Robotic Grippers for Biological Sampling on Deep Reefs. *Soft Robotics*, 3(1):23–33.

[74] Galloway, K. C., Chen, Y., Templeton, E., Rife, B., Godage, I. S., and Barth, E. J. (2019). Fiber Optic Shape Sensing for Soft Robotics. *Soft Robotics*, 6(5):671–684.

[75] Garcia, C. (2019). Robots Are a Few of My Favorite Things. [Online; accessed 15. Sep. 2022], url: https://computerhistory.org/blog/robots-are-a-few-of-my-favorite-things-by-chris-garcia.

[76] Garofalo, G., Ott, C., and Albu-Schaffer, A. (2013). On the closed form computation of the dynamic matrices and their differentiations. *IEEE International Conference on Intelligent Robots and Systems*, pages 2364–2369.

[77] Gazzola, M., Dudte, L. H., McCormick, A. G., and Mahadevan, L. (2018). Forward and inverse problems in the mechanics of soft filaments. *Royal Society Open Science*, 5(6).

[78] Ghorbel, F., Srinivasan, B., and Spong, M. W. (1998). On the uniform boundedness of the inertia matrix of serial robot manipulators. *Journal of Robotic Systems*, 15(1):17–28.

[79] Godage, I. S., Medrano-Cerda, G. A., Branson, D. T., Guglielmino, E., and Caldwell, D. G. (2016). Dynamics for variable length multisection continuum arms. *Int. J. Rob. Res.*, 35(6):695–722.

[80] Godage, I. S. and Walker, I. D. (2015). Dual Quaternion based modal kinematics for multisection continuum arms. *Proceedings - IEEE International Conference on Robotics and Automation*, 2015-June(June):1416–1422.

[81] Gong, D. and Yu, J. (2022). *Design and Control of the McKibben Artificial Muscles Actuated Humanoid Manipulator*. IntechOpen.

[82] Goury, O. and Duriez, C. (2018). Fast, Generic, and Reliable Control and Simulation of Soft Robots Using Model Order Reduction. *IEEE Transactions on Robotics*, 34(6):1565–1576.

[83] Graule, M. A., McCarthy, T. P., Teeple, C. B., Werfel, J., and Wood, R. J. (2022). SoMoGym: A Toolkit for Developing and Evaluating Controllers and Reinforcement Learning Algorithms for Soft Robots. *IEEE Robotics and Automation Letters*, 7(2):4071–4078.

[84] Grazioso, S., Di Gironimo, G., and Siciliano, B. (2019). A Geometrically Exact Model for Soft Continuum Robots: The Finite Element Deformation Space Formulation. *Soft Robotics*, 6(6):790–811.

[85] Grzesiak, A., Becker, R., and Verl, A. (2011). The Bionic Handling Assistant: a success story of additive manufacturing. *Assembly Automation*, 31(4):329–333.

[86] Hairer, E., Lubich, C., and Wanner, G. (2002). *Geometric Numerical Integration: Structure-Preserving Algorithms for Ordinary Differential Equations*. Springer, Berlin, Germany.

[87] Hawkes, E. W., Blumenschein, L. H., Greer, J. D., and Okamura, A. M. (2017). A soft robot that navigates its environment through growth. *Science Robotics*, 2(8):eaan3028.

[88] Hiller, J. and Lipson, H. (2014). Dynamic Simulation of Soft Multimaterial 3D-Printed Objects. *Soft Robotics*, 1(1):88–101.

[89] Hofer, M., Spannagl, L., and D'Andrea, R. (2019). Iterative Learning Control for Fast and Accurate Position Tracking with an Articulated Soft Robotic Arm. *ArXiv e-prints*.

[90] Hogan, N. (1984a). Impedance Control: An Approach to Manipulation. In *1984 American Control Conference*, pages 304–313. IEEE.

[91] Hogan, N. (1984b). Impedance control of industrial robots. *Robotics and Computer-Integrated Manufacturing*, 1(1):97–113.

[92] Hoggett, R. (2012). 1988 - "Shadow" Biped Walker - David Buckley et al (British). [Online; accessed 7. Sep. 2022], url: http://cyberneticzoo.com/walking-machines/1988-shadow-biped-walker-david-buckley-et-al-british.

[93] Hoggett, R. (2014a). 1957 - "Artificial Muscle" - Joseph Laws McKibben (American). [Online; accessed 23. Aug. 2022], url: http://cyberneticzoo.com/bionics/1957-artificial-muscle-joseph-laws-mckibben-american.

[94] Hoggett, R. (2014b). 1981 - Robot Arm with Pneumatic Gripper - Nikolai Teleshev (Russian). [Online; accessed 04. Sep. 2022], url: http://cyberneticzoo.com/bionics/1981-robot-arm-with-pneumatic-gripper-nikolai-teleshev-russian/.

[95] Holzapfel, G. A. (2002). *Nonlinear Solid Mechanics: A Continuum Approach for Engineering Science*, volume 37. Kluwer Academic Publishers, Heidelberg, Germany.

[96] Hu, Y., Anderson, L., Li, T.-M., Sun, Q., Carr, N., Ragan-Kelley, J., and Durand, F. (2019a). DiffTaichi: Differentiable Programming for Physical Simulation. *ArXiv e-prints*.

[97] Hu, Y., Li, T.-M., Anderson, L., Ragan-Kelley, J., and Durand, F. (2019b). Taichi: a language for high-performance computation on spatially sparse data structures. *ACM Transactions on Graphics (TOG)*, 38(6):201.

[98] Hu, Y., Liu, J., Spielberg, A., Tenenbaum, J. B., Freeman, W. T., Wu, J., Rus, D., and Matusik, W. (2019c). ChainQueen: A Real-Time Differentiable Physical Simulator for Soft Robotics. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6265–6271. IEEE.

[99] Hubbard, J. D., Acevedo, R., Edwards, K. M., Alsharhan, A. T., Wen, Z., Landry, J., Wang, K., Schaffer, S., and Sochol, R. D. (2021). Fully 3D-printed soft robots with integrated fluidic circuitry. *Science Advances*, 7(29).

[100] Hughes, J., Culha, U., Giardina, F., Guenther, F., Rosendo, A., and Iida, F. (2016). Soft Manipulators and Grippers: A Review. *Frontiers in Robotics and AI*, 3.

[101] Hyatt, P., Johnson, C. C., and Killpack, M. D. (2020). Model Reference Predictive Adaptive Control for Large-Scale Soft Robots. *Frontiers in Robotics and AI*, 7.

[102] Ilievski, F., Mazzeo, A. D., Shepherd, R. F., Chen, X., and Whitesides, G. M. (2011). Soft Robotics for Chemists. *Angewandte Chemie International Edition*, 50(8):1890–1895.

[103] Immega, G. (1986). Romac muscle powered robots. Technical Report MS86-777.

[104] Jiang, H., Liu, X., Chen, X., Wang, Z., Jin, Y., and Chen, X. (2016). Design and simulation analysis of a soft manipulator based on honeycomb pneumatic networks. In *2016 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pages 350–356. IEEE.

[105] Joachimczak, M., Suzuki, R., and Arita, T. (2014). Fine Grained Artificial Development for Body-Controller Coevolution of Soft-Bodied Animats. *MIT Press*, pages 239–246.

[106] Joachimczak, M., Suzuki, R., and Arita, T. (2015). Improving Evolvability of Morphologies and Controllers of Developmental Soft-Bodied Robots with Novelty Search. *Frontiers in Robotics and AI*, 0.

[107] Jones, B. A. and Walker, I. D. (2006). Kinematics for multisection continuum robots. *IEEE Transaction on Robotics*, 22(1):43–55.

[108] Kamble, V. G., Mersch, J., Tahir, M., Stöckelhuber, K. W., Das, A., and Wießner, S. (2022). Development of Liquid Diene Rubber Based Highly Deformable Interactive Fiber-Elastomer Composites. *Materials*, 15(1):390.

[109] Katzschmann, R. K., DelPreto, J., MacCurdy, R., and Rus, D. (2018). Exploration of underwater life with an acoustically controlled soft robotic fish. *Science Robotics*, 3(16):eaar3449.

[110] Katzschmann, R. K., Santina, C. D., Toshimitsu, Y., Bicchi, A., and Rus, D. (2019a). Dynamic motion control of multi-segment soft robots using piecewise constant curvature matched with an augmented rigid body model. *RoboSoft 2019 - 2019 IEEE International Conference on Soft Robotics*, (February):454–461.

[111] Katzschmann, R. K., Thieffry, M., Goury, O., Kruszewski, A., Guerra, T.-M., Duriez, C., and Rus, D. (2019b). Dynamically Closed-Loop Controlled Soft Robotic Arm using a Reduced Order Finite Element Model with State Observer. In *2019 2nd IEEE International Conference on Soft Robotics (RoboSoft)*, pages 717–724. IEEE.

[112] Kazemipour, A., Fischer, O., Toshimitsu, Y., Wong, K. W., and Katzschmann, R. K. (2022). Adaptive Dynamic Sliding Mode Control of Soft Continuum Manipulators. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 3259–3265. IEEE.

[113] Keplinger, C., Li, T., Baumgartner, R., Suo, Z., and Bauer, S. (2011). Harnessing snap-through instability in soft dielectrics to achieve giant voltage-triggered deformation. *Soft Matter*, 8(2):285–288.

[114] Khatib, O. (1987). A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal on Robotics and Automation*, 3(1):43–53.

[115] Kier, W. M. and Smith, K. K. (1985). Tongues, tentacles and trunks: the biomechanics of movement in muscular-hydrostats. *Zoological Journal of the Linnean Society*, 83(4):307–324.

[116] Kim, K., Agogino, A. K., and Agogino, A. M. (2020). Rolling Locomotion of Cable-Driven Soft Spherical Tensegrity Robots. *Soft Robotics*, 7(3):346–361.

[117] Kim, N. H. (2018). *Introduction Analysis Finite Element to Nonlinear*. Springer.

[118] Kim, S., Laschi, C., and Trimmer, B. (2013). Soft robotics: a bioinspired evolution in robotics. *Trends in Biotechnology*, 31(5):287–294.

[119] Kim, S. Y., Baines, R., Booth, J., Vasios, N., Bertoldi, K., and Kramer-Bottiglio, R. (2019). Reconfigurable soft body trajectories using unidirectionally stretchable composite laminae. *Nature Communications*, 10(3464):1–8.

[120] Kim, Y., Yuk, H., Zhao, R., Chester, S. A., and Zhao, X. (2018). Printing ferromagnetic domains for untethered fast-transforming soft materials. *Nature*, 558:274–279.

[121] Kriegman, S., Blackiston, D., Levin, M., and Bongard, J. (2020a). A scalable pipeline for designing reconfigurable organisms. *Proceedings of the National Academy of Sciences*, 117(4):1853–1859.

[122] Kriegman, S., Blackiston, D., Levin, M., and Bongard, J. (2020b). A scalable pipeline for designing reconfigurable organisms. *Proceedings of the National Academy of Sciences*, 117(4):1853–1859.

[123] Kukolj, M. (1988). Axially contractable actuautor. US Patent No. 4,733,603.

[124] Largilliere, F., Verona, V., Coevoet, E., Sanz-Lopez, M., Dequidt, J., and Duriez, C. (2015). Real-time control of soft-robots using asynchronous finite element modeling. *Proceedings - IEEE International Conference on Robotics and Automation*, 2015-June(June):2550–2555.

[125] Laschi, C. and Cianchetti, M. (2014). Soft Robotics: New Perspectives for Robot Bodyware and Control. *Frontiers in Bioengineering and Biotechnology*, 0.

[126] Li, D., Fan, D., Zhu, R., Lei, Q., Liao, Y., Yang, X., Pan, Y., Wang, Z., Wu, Y., Liu, S., and Wang, H. (2022). Origami-Inspired Soft Twisting Actuator. *Soft Robotics*.

[127] Li, G., Chen, X., Zhou, F., Liang, Y., Xiao, Y., Cao, X., Zhang, Z., Zhang, M., Wu, B., Yin, S., Xu, Y., Fan, H., Chen, Z., Song, W., Yang, W., Pan, B., Hou, J., Zou, W., He, S., Yang, X., Mao, G., Jia, Z., Zhou, H., Li, T., Qu, S., Xu, Z., Huang, Z., Luo, Y., Xie, T., Gu, J., Zhu, S., and Yang, W. (2021). Self-powered soft robot in the Mariana Trench. *Nature*, 591:66–71.

[128] Li, S., Vogt, D. M., Rus, D., and Wood, R. J. (2017a). Fluid-driven origami-inspired artificial muscles. *Proceedings of the National Academy of Sciences*, 114(50):13132–13137.

[129] Li, Z., Wu, L., Ren, H., and Yu, H. (2017b). Kinematic comparison of surgical tendon-driven manipulators and concentric tube manipulators. *Mechanism and Machine Theory*, 107(September 2016):148–165.

[130] López-Valdeolivas, M., Liu, D., Broer, D. J., and Sánchez-Somolinos, C. (2018). 4D Printed Actuators with Soft-Robotic Functions. *Macromolecular Rapid Communications*, 39(5):1700710.

[131] Lynch, K. M. and Park, F. C. (2017). *Modern Robotics: Mechanics, Planning, and Control*. Cambridge University Press, Cambridge, England, UK.

[132] Maas, S. A., Ellis, B. J., Ateshian, G. A., and Weiss, J. A. (2012). FEBio: finite elements for biomechanics. *Journal of Biomechanical Engineering*, 134(1):011005.

[133] Marchese, A. D., Katzschmann, R. K., and Rus, D. (2015). A Recipe for Soft Fluidic Elastomer Robots. *Soft Robotics*, 2(1):7–25.

[134] Marchese, A. D., Onal, C. D., and Rus, D. (2014). Autonomous Soft Robotic Fish Capable of Escape Maneuvers Using Fluidic Elastomer Actuators. *Soft Robotics*, 1(1):75–87.

[135] Marchese, A. D. and Rus, D. (2016). Design, kinematics, and control of a soft spatial fluidic elastomer manipulator. *The International Journal of Robotics Research*, 35(7):840–869.

[136] Markiewicz, B. R. (1973). Analysis of the computed torque drive method and comparison with conventional position servo for a computer-controlled manipulator. Technical Memorandum 33-601.

[137] Mathew, A. T., Armanini, C., Alshehhi, A. A. S. A., Hmida, I. M. B., and Renda, F. (2022a). Multifunctional Underwater Soft Robots: A Simulation Essay. *IOP Conference Series: Materials Science and Engineering*, 1261(1):012008.

[138] Mathew, A. T., Hmida, I. B., Armanini, C., Boyer, F., and Renda, F. (2021). SoRoSim: a MATLAB Toolbox for Soft Robotics Based on the Geometric Variable-strain Approach. *ArXiv e-prints*.

[139] Mathew, A. T., Hmida, I. M. B., Armanini, C., Boyer, F., and Renda, F. (2022b). SoRoSim: A MATLAB Toolbox for Hybrid Rigid-Soft Robots Based on the Geometric Variable-Strain Approach. *IEEE Robotics & Automation Magazine*, pages 2–18.

[140] McDonald, K., Rendos, A., Woodman, S., Brown, K. A., and Ranzani, T. (2020). Magnetorheological Fluid-Based Flow Control for Soft Robots. *Advanced Intelligent Systems*, 2(11):2000139.

[141] Merriam-Webster (1983). *Websters Ninth New Collegiate Dictionary*. Merriam-webster+ Inc.

[142] Meyers, M. A. and Chawla, K. K. (2009). *Mechanical Behavior of Materials*. Cambridge University Press, Cambridge, England, UK.

[143] Mickle, P. (2008). 1961: A peep into the automated future. [Online; accessed 10. Aug. 2022], url: http://www.capitalcentury.com/1961.html.

[144] Milana, E., Stella, F., Gorissen, B., Reynaerts, D., and Santina, C. D. (2021). Model-Based Control Can Improve the Performance of Artificial Cilia. In *2021 IEEE 4th International Conference on Soft Robotics (RoboSoft)*, pages 527–530. IEEE.

[145] Mochiyama, H. (1992). *Theory and applications of hyper-redundant robotic manipulators*. PhD thesis.

[146] Mochiyama, H., Shimemura, E., and Kobayashi, H. (1998). Direct kinematics of manipulators with hyper degrees of freedom and Frenet-Serret formula. *Proceedings. 1998 IEEE International Conference on Robotics and Automation (Cat. No.98CH36146)*, 2(May):1653–1658.

[147] Mochiyama, H., Shimemura, E., and Kobayashi, H. (1999). Shape Control of Manipulators with Hyper Degrees of Freedom. *The International Journal of Robotics Research*, 18(6):584–600.

[148] Mochiyama, H. and Suzuki, T. (2003). Kinematics and dynamics of a cable-like hyper-flexible manipulator. *IEEE International Conference on Robotics and Automation*, pages 3672–3677.

[149] Moerman, K. M. (2018). GIBBON: The Geometry and Image-Based Bioengineering add-On. *Journal of Open Source Software*, 3(22):506.

[150] Morgan, A. P. and Narendra, K. S. (1977). On the Uniform Asymptotic Stability of Certain Linear Nonautonomous Differential Equations. *SIAM Journal on Control and Optimization*.

[151] Morin, A. (1953). Elastic diaphragm. US Patent No. 2,642,091.

[152] Mosadegh, B., Polygerinos, P., Keplinger, C., Wennstedt, S., Shepherd, R. F., Gupta, U., Shim, J., Bertoldi, K., Walsh, C. J., and Whitesides, G. M. (2014). Pneumatic networks for soft robotics that actuate rapidly. *Advanced Functional Materials*, 24(15):2163–2170.

[153] Mullin, T., Deschanel, S., Bertoldi, K., and Boyce, M. C. (2007). Pattern Transformation Triggered by Deformation. *Physical Review Letters*, 99(8):084301.

[154] Murray, R. M., Li, Z., and Sastry, S. S. (1994). *A Mathematical Introduction to Robotic Manipulation*, volume 29.

[155] Mustaza, S. M., Elsayed, Y., Lekakou, C., Saaj, C., and Fras, J. (2019). Dynamic Modeling of Fiber-Reinforced Soft Manipulator: A Visco-Hyperelastic Material-Based Continuum Mechanics Approach. *Soft Robotics*, 6(3):305–317.

[156] Naccarato, F. and Hughes, P. (1991). Inverse kinematics of variable-geometry truss manipulators. *Journal of Robotic Systems*, 8(2):249–266.

[157] Naccarato, F., Hughes, P., and Hughes, P. (1989). an inverse kinematics algorithm for a highly redundant variable-geometry-truss manipulator. *in JPL, Proceedings of the 3rd Annual Conference on Aerospace Computational Control, Volume 1*.

[158] Naughton, N., Sun, J., Tekinalp, A., Parthasarathy, T., Chowdhary, G., and Gazzola, M. (2021). Elastica: A Compliant Mechanics Environment for Soft Robotic Control. *IEEE Robotics and Automation Letters*, 6(2):3389–3396.

[159] Ortega, R., Perez, J. A. L., Nicklasson, P. J., and Sira-Ramirez, H. (1998). *Passivity-based Control of Euler-Lagrange Systems: Mechanical, Electrical and Electromechanical Applications*. Springer, London, England, UK.

[160] Ortega, R., Spong, M. W., Gómez-Estern, F., and Blankenstein, G. (2002). Stabilization of a Class of Underactuated MechanicalSystems Via Interconnection and Damping Assignment. *IEEE Transactions on Automatic Control*, 47(8):1218–1233.

[161] Park, Y.-L., Chen, B.-R., and Wood, R. J. (2012). Design and Fabrication of Soft Artificial Skin Using Embedded Microchannels and Liquid Conductors. *IEEE Sensors Journal*, 12(8):2711–2718.

[162] Paynter, H. (1974a). Hyperboloid of revolution fluid-driven tension actuators and methods of making. US Patent No. 4721 030.

[163] Paynter, H. (1974b). Tension actuated pressurized gas driven rotary motors. US Patent No. 3,854,383.

[164] Peter Asaro and Selma Šabanović (2010). Oral-History: Victor Scheinman - Engineering and Technology History Wiki. *IEEE History Center*. transcript: https://ethw.org/Oral-History:Victor_Scheinman.

[165] Pilz da Cunha, M., Ambergen, S., Debije, M. G., Homburg, E. F., den Toonder, J. M., and Schenning, A. P. (2020). A Soft Transporter Robot Fueled by Light. *Advanced Science*, 7(5):1–7.

[166] Polygerinos, P., Lyne, S., Wang, Z., Nicolini, L. F., Mosadegh, B., Whitesides, G. M., and Walsh, C. J. (2013). Towards a soft pneumatic glove for hand rehabilitation. *IEEE International Conference on Intelligent Robots and Systems*, pages 1512–1517.

[167] Polygerinos, P., Wang, Z., Galloway, K. C., Wood, R. J., and Walsh, C. J. (2015). Soft robotic glove for combined assistance and at-home rehabilitation. *Robotics and Autonomous Systems*, 73:135–143.

[168] Renaud, C., Cros, J. M., Feng, Z. Q., and Yang, B. (2009). The Yeoh model applied to the modeling of large deformation contact/impact problems. *International Journal of Impact Engineering*, 36(5):659–666.

[169] Renda, F., Armanini, C., Lebastard, V., Candelier, F., and Boyer, F. (2020). A Geometric Variable-Strain Approach for Static Modeling of Soft Manipulators with Tendon and Fluidic Actuation. *IEEE Robotics and Automation Letters*, 5(3):4006–4013.

[170] Renda, F., Boyer, F., Dias, J., and Seneviratne, L. (2018). Discrete Cosserat Approach for Multisection Soft Manipulator Dynamics. *IEEE Transactions on Robotics*, 34(6):1518–1533.

[171] Renda, F., Cianchetti, M., Abidi, H., Dias, J., and Seneviratne, L. (2017). Screw-Based Modeling of Soft Manipulators With Tendon and Fluidic Actuation. *Journal of Mechanisms and Robotics*, 9(4).

[172] Robinson, G. and Davies, J. B. C. (1999). Continuum robots - a state of the art. In *Proceedings 1999 IEEE International Conference on Robotics and Automation (Cat. No.99CH36288C)*, volume 4, pages 2849–2854vol.4. IEEE.

[173] Roels, E., Terryn, S., Brancart, J., Verhelle, R., Van Assche, G., and Vanderborght, B. (2020). Additive Manufacturing for Self-Healing Soft Robots. *Soft Robotics*, 7(6):711–723.

[174] Roh, S., Okello, L. B., Golbasi, N., Hankwitz, J. P., Liu, J. A.-C., Tracy, J. B., and Velev, O. D. (2019). 3D-Printed Silicone Soft Architectures with Programmed Magneto-Capillary Reconfiguration. *Advanced Materials Technologies*, 4(4):1800528.

[175] Röntgen, W. (1880). Ueber die durch Electricität bewirkten Form- und Volumenänderungen von dielectrischen Körpern. *Annalen der Physik*, 247:771–786.

[176] Runge, G., Wiese, M., Gunther, L., and Raatz, A. (2017). A framework for the kinematic modeling of soft material robots combining finite element analysis and piecewise constant curvature kinematics. *2017 3rd International Conference on Control, Automation and Robotics, ICCAR 2017*, (April):7–14.

[177] Rus, D. and Tolley, M. T. (2015). Design, fabrication and control of soft robots. *Nature*, 521:467–475.

[178] Sadati, S. M. H., Naghibi, S. E., Shiva, A., Michael, B., Renson, L., Howard, M., Rucker, C. D., Althoefer, K., Nanayakkara, T., Zschaler, S., Bergeles, C., Hauser, H., and Walker, I. D. (2020). TMTDyn: A Matlab package for modeling and control of hybrid rigid–continuum robots based on discretized lumped systems and reduced-order models. *International Journal on Robotics Research*, 40(1):296–347.

[179] Salerno, R. J., Reinholtz, C. F., Robertshaw, H. H., Reinholtz, C. F., and Robertshaw, H. H. (1989). shape control of high degree-of-freedom variable geometry trusses. *NASA. Langley Research Center, Proceedings of the Workshop on Computational Aspects in the Control of Flexible Systems, Part 2*.

[180] Sanan, S. (2013). *Soft Inflatable Robots for Safe Physical Human Interaction*. PhD thesis.

[181] Satheeshbabu, S., Uppalapati, N. K., Chowdhary, G., and Krishnan, G. (2019). Open Loop Position Control of Soft Continuum Arm Using Deep Reinforcement Learning. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 5133–5139. IEEE.

[182] Schaft, A. J. (2004). Port-Hamiltonian Systems: Network Modeling and Control of Nonlinear Physical Systems. *Advanced Dynamics and Control of Structures and Machines*, 444(1):127–167.

[183] Schegg, P., Ménager, E., Khairallah, E., Marchal, D., Dequidt, J., Preux, P., and Duriez, C. (2022). SofaGym: An Open Platform for Reinforcement Learning Based on Soft Robot Simulations. *Soft Robotics*, 0(0).

[184] Schulte, H. (1961). The characteristics of the mckibben artificial muscle. *The Application of External Power in Prosthetics and Orthotic*, pages 94–115.

[185] Shepherd, R. F., Ilievski, F., Choi, W., Morin, S. A., Stokes, A. A., Mazzeo, A. D., Chen, X., Wang, M., and Whitesides, G. M. (2011). Multigait soft robot. *Proceedings of the National Academy of Sciences*, 108(51):20400–20403.

[186] Shim, J., Shan, S., Košmrlj, A., Kang, S. H., Chen, E. R., Weaver, J. C., and Bertoldi, K. (2013). Harnessing instabilities for design of soft reconfigurable auxetic/chiral materials. *Soft Matter*, 9(34):8198–8202.

[187] Shiriaev, A., Ludvigsen, H., Egeland, O., and Pogromsky, A. (1999). On global properties of passivity based control of the inverted pendulum. In *Proceedings of the 38th IEEE Conference on Decision and Control (Cat. No.99CH36304)*, volume 3, pages 2513–2518vol.3. IEEE.

[188] Simo, J. C. and Vu-Quoc, L. (1986). A three-dimensional finite-strain rod model. part II: Computational aspects. *Computer Methods in Applied Mechanics and Engineering*, 58(1):79–116.

[189] Sinatra, N. R., Teeple, C. B., Vogt, D. M., Parker, K. K., Gruber, D. F., and Wood, R. J. (2019). Ultragentle manipulation of delicate structures using a soft robotic gripper. *Science Robotics*, 4(33):eaax5425.

[190] Skorina, E. H., Luo, M., Tao, W., Chen, F., Fu, J., and Onal, C. D. (2017). Adapting to Flexibility: Model Reference Adaptive Control of Soft Bending Actuators. *IEEE Robotics and Automation Letters*, 2(2):964–970.

[191] Slotine, J.-J. E. and Li, W. (1989). Composite adaptive control of robot manipulators. *Automatica*, 25(4):509–519.

[192] Slotine, J. J. E. and Weiping, L. (1988). Adaptive manipulator control: A case study. *IEEE Transactions on Automatic Control*, 33(11):995–1003.

[193] Slotine J.-J., E. (1987). Theoretical Issues In Adaptive Manipulator Control. *Proc. Fifth Yale Workshop on the Applications of Adaptive Systems Theory*, pages 252–258.

[194] Smith, B., Goes, F. D., and Kim, T. (2018). Stable neo-hookean flesh simulation. *ACM Trans. Graph.*, 37(2).

[195] Smith, L., Haimes, J., and MacCurdy, R. (2022a). Stretching the Boundary: Shell Finite Elements for Pneumatic Soft Actuators. In *2022 IEEE 5th International Conference on Soft Robotics (RoboSoft)*, pages 403–408. IEEE.

[196] Smith, L., Hainsworth, T., Haimes, J., and MacCurdy, R. (2022b). Automated Synthesis of Bending Pneumatic Soft Actuators. In *2022 IEEE 5th International Conference on Soft Robotics (RoboSoft)*, pages 358–363. IEEE.

[197] Smith, L. and MacCurdy, R. (2023). SoRoForge: End-to-End Soft Actuator Design. *IEEE Transactions on Automation Science and Engineering*, pages 1–12.

[198] Sonneville, V., Cardona, A., and Brüls, O. (2014). Geometrically exact beam finite element formulated on the special Euclidean group SE(3). *Computer Methods in Applied Mechanics and Engineering*, 268:451–474.

[199] Spong, M. W. (1996a). Energy Based Control of a Class of Underactuated Mechanical Systems. *IFAC Proceedings Volumes*, 29(1):2828–2832.

[200] Spong, M. W. (1996b). Energy Based Control of a Class of Underactuated Mechanical Systems. *IFAC Proceedings Volumes*, 29(1):2828–2832.

[201] Spong, M. W., Hutchinson, S., and Vidyasagar, M. (2006). *Robot modeling and control*. John Wiley & Sons, New York.

[202] Stölzle, M. and Santina, C. D. (2021). Piston-Driven Pneumatically-Actuated Soft Robots: Modeling and Backstepping Control. *IEEE Control Syst. Lett.*, 6:1837–1842.

[203] Suzumori, K., Iikura, S., and Tanaka, H. (1991). Development of flexible microactuator and its applications to robotic mechanisms. In *Proceedings. 1991 IEEE International Conference on Robotics and Automation*, volume 2, pages 1622–1627.

[204] Suzumori, K., Iikura, S., and Tanaka, H. (1992). Applying a flexible microactuator to robotic mechanisms. *IEEE Control Systems Magazine*, 12(1):21–27.

[205] Takagi, T., Sakaguchi, Y., and Corp, B. (1983). Pneumatic actuator for manipulator. US Patent No. 4,615,260.

[206] Tapia, J., Knoop, E., Mutný, M., Otaduy, M. A., and Bächer, M. (2020). MakeSense: Automated sensor design for proprioceptive soft robots. *Soft Robotics*, 7(3).

[207] Tatlicioglu, E., Walker, I. D., and Dawson, D. M. (2007a). Dynamic Modelling for Planar Extensible Continuum Robot Manipulators. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 1357–1362. IEEE.

[208] Tatlicioglu, E., Walker, I. D., and Dawson, D. M. (2007b). New dynamic models for planar extensible continuum robot manipulators. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1485–1490. IEEE.

[209] Tedrake, R. (2022a). *Underactuated Robotics – Algorithms for Walking, Running, Swimming, Flying, and Manipulation*. Course Notes for MIT 6.832, url: http://underactuated.mit.edu.

[210] Tedrake, R. (2022b). Underactuated robotics – algorithms for walking, running, swimming, flying, and manipulation. Course Notes for MIT 6.832.

[211] Tekinalp, A., Kim, S. H., Parthasarathy, T., and Bhosale, Y. (2022). Pyelastica: A computational framework for cosserat rod assemblies.

[212] Teng, L., Jeronimo, K., Wei, T., Nemitz, M. P., Lyu, G., and Stokes, A. A. (2018). Integrating soft sensor systems using conductive thread. *Journal of Micromechanics and Microengineering*, 28(5):054001.

[213] Terryn, S., Langenbach, J., Roels, E., Brancart, J., Bakkali-Hassani, C., Poutrel, Q.-A., Georgopoulou, A., George Thuruthel, T., Safaei, A., Ferrentino, P., Sebastian, T., Norvez, S., Iida, F., Bosman, A. W., Tournilhac, F., Clemens, F., Van Assche, G., and Vanderborght, B. (2021). A review on self-healing polymers for soft robotics. *Materials Today*, 47:187–205.

[214] Thieffry, M. (2020). Model-Based Dynamic Control of Soft Robots To cite this version : HAL Id : tel-02363267 Maxime Thieffry Model-Based Dynamic Control of Soft Robots.

[215] Thieffry, M., Kruszewski, A., Goury, O., Thieffry, M., Kruszewski, A., Goury, O., Guerra, T.-m., Duriez, C., Thieffry, M., and Kruszewski, A. (2017). Dynamic Control of Soft Robots.

[216] Thieffry, M., Kruszewski, A., Guerra, T.-M., and Duriez, C. (2019). Trajectory Tracking Control Design for Large-Scale Linear Dynamical Systems With Applications to Soft Robotics. *IEEE Transactions on Control Systems Technology*, 29(2):556–566.

[217] Thuruthel, T. G., Falotico, E., Renda, F., and Laschi, C. (2017). Learning dynamic models for open loop predictive control of soft robotic manipulators. *Bioinspiration & Biomimetics*, 12(6):066003.

[218] Thuruthel, T. G., Falotico, E., Renda, F., and Laschi, C. (2018). Model-Based Reinforcement Learning for Closed-Loop Dynamic Control of Soft Robotic Manipulators. *IEEE Transactions on Robotics*, 35(1):124–134.

[219] Tian, J., Zhao, X., Gu, X. D., and Chen, S. (2020). Designing Ferromagnetic Soft Robots (FerroSoRo) with Level-Set-Based Multiphysics Topology Optimization. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10067–10074. IEEE.

[220] Till, J., Aloi, V., and Rucker, C. (2019). Real-time dynamics of soft and continuum robots based on Cosserat rod models. *International Journal of Robotics Research*, 38(6):723–746.

[221] Tolley, M. T., Shepherd, R. F., Karpelson, M., Bartlett, N. W., Galloway, K. C., Wehner, M., Nunes, R., Whitesides, G. M., and Wood, R. J. (2014a). An untethered jumping soft robot. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 561–566. IEEE.

[222] Tolley, M. T., Shepherd, R. F., Mosadegh, B., Galloway, K. C., Wehner, M., Karpelson, M., Wood, R. J., and Whitesides, G. M. (2014b). A Resilient, Untethered Soft Robot. *Soft Robotics*, 1(3):213–223.

[223] Tonkens, S., Lorenzetti, J., and Pavone, M. (2020). Soft Robot Optimal Control Via Reduced Order Finite Element Models. *ArXiv e-prints*.

[224] Tonkens, S., Lorenzetti, J., and Pavone, M. (2021). Soft Robot Optimal Control Via Reduced Order Finite Element Models. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 12010–12016. IEEE.

[225] Toshimitsu, Y., Wong, K. W., Buchner, T., and Katzschmann, R. (2021). SoPrA: Fabrication & Dynamical Modeling of a Scalable Soft Continuum Robotic Arm with Integrated Proprioceptive Sensing. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 653–660. IEEE.

[226] Trivedi, D., Rahn, C. D., Kier, W. M., and Walker, I. D. (2008). Soft Robotics: Biological Inspiration, State of the Art, and Future Research. *Applied Bionics and Biomechanics*, 5(3):520417–117.

[227] Trumić, M., Santina, C. D., Jovanović, K., and Fagiolini, A. (2020). Adaptive Control of Soft Robots Based on an Enhanced 3D Augmented Rigid Robot Matching. *IEEE Control Systems Letters*, 5(6):1934–1939.

[228] Vantomme, G., Elands, L. C. M., Gelebart, A. H., Meijer, E. W., Pogromsky, A. Y., Nijmeijer, H., and Broer, D. J. (2021). Coupled liquid crystalline oscillators in Huygens' synchrony. *Nature Materials*, 20:1702–1706.

[229] Wallin, T. J., Pikul, J., and Shepherd, R. F. (2018). 3D printing of soft robotic systems. *Nature Reviews Materials*, 3:84–100.

[230] Wang, H., Zhang, R., Chen, W., Liang, X., and Pfeifer, R. (2016). Shape Detection Algorithm for Soft Manipulator Based on Fiber Bragg Gratings. *IEEE/ASME Transactions on Mechatronics*, 21(6):2977–2982.

[231] Wang, R., Zhang, X., Zhu, B., Zhang, H., Chen, B., and Wang, H. (2020). Topology optimization of a cable-driven soft robotic gripper. *Structural and Multidisciplinary Optimization*, 62(5):2749–2763.

[232] Wang, T., Halder, U., Gribkova, E., Gillette, R., Gazzola, M., and Mehta, P. G. (2022a). A Sensory Feedback Control Law for Octopus Arm Movements. In *2022 IEEE 61st Conference on Decision and Control (CDC)*, pages 1059–1066. IEEE.

[233] Wang, T., Zhang, Y., Zhu, Y., and Zhu, S. (2019). A computationally efficient dynamical model of fluidic soft actuators and its experimental verification. *Mechatronics*, 58:1–8.

[234] Wang, Z., Wang, G., Chen, X., and Freris, N. M. (2022b). Dynamical Modeling and Control of Soft Robots with Non-constant Curvature Deformation. *ArXiv e-prints*.

[235] Webster, R. J. and Jones, B. A. (2010). Design and Kinematic Modeling of Constant Curvature Continuum Robots: A Review. *International Journal of Robotics Research*, 29(13):1661–1683.

[236] Weerakoon, L. and Chopra, N. (2021). Swing up Control of a Soft Inverted Pendulum with Revolute Base. In *2021 60th IEEE Conference on Decision and Control (CDC)*, pages 685–690. IEEE Press.

[237] Wehner, M., Truby, R. L., Fitzgerald, D. J., Mosadegh, B., Whitesides, G. M., Lewis, J. A., and Wood, R. J. (2016). An integrated design and fabrication strategy for entirely soft, autonomous robots. *Nature*, 536(7617):451–455.

[238] Weisburd, S. (1988). The muscular machinery of tentacles, trunks and tongues. *Science News*, 113:204–205.

[239] Wilson, J. F. and Inou, N. (2007). Bellows-type springs for robotics. In *Proc. Adv. Spring Technol. JSSE 60th Anniversary Int. Symp*, pages 109–119.

[240] Wilson, J. F. and Mahajan, U. (1989). The Mechanics and Positioning of Highly Flexible Manipulator Limbs. *Journal of Mechanisms, Transmissions, and Automation in Design*, 111(2):232–237.

[241] Wilson, J. F. and Snyder, J. M. (1988). The Elastica With End-Load Flip-Over. *Journal of Applied Mechanics*, 55(4):845–848.

[242] Wu, K. and Zheng, G. (2021a). FEM-Based Gain-Scheduling Control of a Soft. *Robotics and Automation Letters*, 6(2):3081–3088.

[243] Wu, K. and Zheng, G. (2021b). FEM-Based Gain-Scheduling Control of a Soft Trunk Robot. *IEEE Robotics and Automation Letters*, 6(2):3081–3088.

[244] Wu, S., Baker, G. L., Yin, J., and Zhu, Y. (2021a). Fast Thermal Actuators for Soft Robotics. *Soft Robotics*, 0(0).

[245] Wu, S., Ze, Q., Dai, J., Udipi, N., Paulino, G. H., and Zhao, R. (2021b). Stretchable origami robotic arm with omnidirectional bending and twisting. *Proceedings of the National Academy of Sciences*, 118(36):e2110023118.

[246] Xavier, M. S., Tawk, C. D., Zolfagharian, A., Pinskier, J., Howard, D., Young, T., Lai, J., Harrison, S. M., Yong, Y. K., Bodaghi, M., and Fleming, A. J. (2022). Soft Pneumatic Actuators: A Review of Design, Fabrication, Modeling, Sensing, Control and Applications. *IEEE Access*, 10:59442–59485.

[247] Yang, D., Verma, M. S., So, J. H., Mosadegh, B., Keplinger, C., Lee, B., Khashai, F., Lossner, E., Suo, Z., and Whitesides, G. M. (2016). Buckling Pneumatic Linear Actuators Inspired by Muscle. *Advanced Materials Technologies*, 1(3):31–33.

[248] Yap, H. K., Lim, J. H., Nasrallah, F., Goh, J. C. H., and Yeow, R. C. H. (2015). A soft exoskeleton for hand assistive and rehabilitation application using pneumatic actuators with variable stiffness. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4967–4972. IEEE.

[249] Yarlott, J. (1972). Fluid actuator. US Patent No. 3,645,173.

[250] Zanna, A. (1999). Collocation and Relaxed Collocation for the Fer and the Magnus Expansions on JSTOR. *SIAM Journal on Numerical Analysis*, 36(4):1145–1182.

[251] Zhang, X., Chan, F., Parthasarathy, T., and Gazzola, M. (2019). Modeling and simulation of complex dynamic musculoskeletal architectures. *Nature Communications*, 10(1):1–12.

[252] Zhang, Z., Bieze, T. M., Dequidt, J., Kruszewski, A., and Duriez, C. (2017). Visual servoing control of soft robots based on finite element model. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2895–2901.

Acknowledgements

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum. Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris. Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa. Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis

augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula. Fusce mauris. Vestibulum luctus nibh at lectus. Sed bibendum, nulla a faucibus semper, leo velit ultricies tellus, ac venenatis arcu wisi vel nisl. Vestibulum diam. Aliquam pellentesque, augue quis sagittis posuere, turpis lacus congue quam, in hendrerit risus eros eget felis. Maecenas eget erat in sapien mattis porttitor. Vestibulum porttitor. Nulla facilisi. Sed a turpis eu lacus commodo facilisis. Morbi fringilla, wisi in dignissim interdum, justo lectus sagittis dui, et vehicula libero dui cursus dui. Mauris tempor ligula sed lacus. Duis cursus enim ut augue. Cras ac magna. Cras nulla. Nulla egestas. Curabitur a leo. Quisque egestas wisi eget nunc. Nam feugiat lacus vel est. Curabitur consectetuer. Suspendisse vel felis. Ut lorem lorem, interdum eu, tincidunt sit amet, laoreet vitae, arcu. Aenean faucibus pede eu ante. Praesent enim elit, rutrum at, molestie non, nonummy vel, nisl. Ut lectus eros, malesuada sit amet, fermentum eu, sodales cursus, magna. Donec eu purus. Quisque vehicula, urna sed ultricies auctor, pede lorem egestas dui, et convallis elit erat sed nulla. Donec luctus. Curabitur et nunc. Aliquam dolor odio, commodo pretium, ultricies non, pharetra in, velit. Integer arcu est, nonummy in, fermentum faucibus, egestas vel, odio. Sed commodo posuere pede. Mauris ut est. Ut quis purus. Sed ac odio. Sed vehicula hendrerit sem. Duis non odio. Morbi ut dui. Sed accumsan risus eget odio. In hac habitasse platea dictumst. Pellentesque non elit. Fusce sed justo eu urna porta tincidunt. Mauris felis odio, sollicitudin sed, volutpat a, ornare ac, erat. Morbi quis dolor. Donec pellentesque, erat ac sagittis semper, nunc dui lobortis purus, quis congue purus metus ultricies tellus. Proin et quam. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent sapien turpis, fermentum vel, eleifend faucibus, vehicula eu, lacus. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Donec odio elit, dictum in, hendrerit sit amet, egestas sed, leo. Praesent feugiat sapien aliquet odio. Integer vitae justo. Aliquam vestibulum fringilla lorem. Sed neque lectus, consectetuer at, consectetuer sed, eleifend ac, lectus. Nulla facilisi. Pellentesque eget lectus. Proin eu metus. Sed porttitor. In hac habitasse platea dictumst. Suspendisse eu lectus. Ut mi mi, lacinia sit amet, placerat et, mollis vitae, dui. Sed ante tellus, tristique ut, iaculis eu, malesuada ac, dui. Mauris nibh leo, facilisis non, adipiscing quis, ultrices a, dui. Morbi luctus, wisi viverra faucibus pretium, nibh est placerat odio, nec commodo wisi enim eget quam. Quisque libero justo, consectetuer a, feugiat vitae, porttitor eu, libero. Suspendisse sed mauris vitae elit sollicitudin malesuada. Maecenas ultricies eros sit amet ante. Ut venenatis velit. Maecenas sed mi eget dui va-

rius euismod. Phasellus aliquet volutpat odio. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Pellentesque sit amet pede ac sem eleifend consectetuer. Nullam elementum, urna vel imperdiet sodales, elit ipsum pharetra ligula, ac pretium ante justo a nulla. Curabitur tristique arcu eu metus. Vestibulum lectus. Proin mauris. Proin eu nunc eu urna hendrerit faucibus. Aliquam auctor, pede consequat laoreet varius, eros tellus scelerisque quam, pellentesque hendrerit ipsum dolor sed augue. Nulla nec lacus. Suspendisse vitae elit. Aliquam arcu neque, ornare in, ullamcorper quis, commodo eu, libero. Fusce sagittis erat at erat tristique mollis. Maecenas sapien libero, molestie et, lobortis in, sodales eget, dui. Morbi ultrices rutrum lorem. Nam elementum ullamcorper leo. Morbi dui. Aliquam sagittis. Nunc placerat. Pellentesque tristique sodales est. Maecenas imperdiet lacinia velit. Cras non urna. Morbi eros pede, suscipit ac, varius vel, egestas non, eros. Praesent malesuada, diam id pretium elementum, eros sem dictum tortor, vel consectetuer odio sem sed wisi. Sed feugiat. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Ut pellentesque augue sed urna. Vestibulum diam eros, fringilla et, consectetuer eu, nonummy id, sapien. Nullam at lectus. In sagittis ultrices mauris. Curabitur malesuada erat sit amet massa. Fusce blandit. Aliquam erat volutpat. Aliquam euismod. Aenean vel lectus. Nunc imperdiet justo nec dolor. Etiam euismod. Fusce facilisis lacinia dui. Suspendisse potenti. In mi erat, cursus id, nonummy sed, ullamcorper eget, sapien. Praesent pretium, magna in eleifend egestas, pede pede pretium lorem, quis consectetuer tortor sapien facilisis magna. Mauris quis magna varius nulla scelerisque imperdiet. Aliquam non quam. Aliquam porttitor quam a lacus. Praesent vel arcu ut tortor cursus volutpat. In vitae pede quis diam bibendum placerat. Fusce elementum convallis neque. Sed dolor orci, scelerisque ac, dapibus nec, ultricies ut, mi. Duis nec dui quis leo sagittis commodo.

List of publications

Peer-reviewed journal articles

- B. Caasenbrood, A. Pogromsky and H. Nijmeijer, "*Generative Design of Soft Robotic Actuators – a Gradient-based Approach*", Frontiers in Robotics and AI, 2022. (*in preparation for journal submission*);

- B. Caasenbrood, A. Pogromsky and H. Nijmeijer, "*Energy-shaping Controllers for Soft Robot Manipulators through Port-Hamiltonian Cosserat Models*", SN Computer Science Springer, 2022.;

- B. Caasenbrood, A. Pogromsky and H. Nijmeijer, "*Control-oriented Models for Hyper-elastic Soft Robots through Differential Geometry of Curves*", Soft Robotics, 2022.

Peer-reviewed articles in conference proceedings

- H. Khanh Chu, B. Caasenbrood, , H. Nijmeijer and I.A. Kuling, "*A Desktop-sized Platform for Real-time Control Applications of Pneumatic Soft Robots*," IEEE International Conference on Soft Robotics, RoboSoft 2022, pp 217-223.

- B. Caasenbrood, F.E. van Beek, H. Khanh Chu, and I.A. Kuling, "*A Desktop-sized Platform for Real-time Control Applications of Pneumatic Soft Robots*," IEEE International Conference on Soft Robotics, RoboSoft 2022, pp 217-223.

- B. Caasenbrood, F.E. van Beek, H. Khanh Chu, and I.A. Kuling, "*A Desktop-sized Platform for Real-time Control Applications of Pneumatic Soft Robots*," IEEE International Conference on Soft Robotics, RoboSoft 2022, pp 217-223.

- A. Amoozandeh Nobaveh, and B. Caasenbrood, "*Design Feasibility of an Energy-efficient Wrist Exoskeleton using Compliant Beams and Soft Actuators*", Proceedings of the 18th International Consortium for Rehabilitation Robotics, 2022 (accepted).

- B. Caasenbrood, A. Pogromsky and H. Nijmeijer, "*Energy-based control for Soft Robots using Cosserat-beam models*", Proceedings of the 18th International Conference on Informatics in Control, Automation and Robotics, 2021, pp. 311–319.
- B. Caasenbrood, A. Pogromsky and H. Nijmeijer, "*A Computational Design Framework for Pressure-driven Soft Robots through Nonlinear Topology Optimization*," 2020 3rd IEEE International Conference on Soft Robotics, 2020, pp. 633-638.
- B. Caasenbrood, A. Pogromsky and H. Nijmeijer, "*Dynamic modeling of hyper-elastic soft robots using spatial curves*," IFAC World Congress, IFAC-Papers OnLine, 2020, pp. 9238-9243.

Talks, workshops, and non peer-reviewed abstracts

- B. Caasenbrood, talk on "*3D-printed Soft Robotics*," Symposium on Robotic Technologies, Ultimaker, 2022. (invited speaker).
- B. Caasenbrood, `SOROTOKI`: *an open-source MATLAB toolkit for Design, Modelling and Control of Soft Robots*," 4TU Federation's Symposium on Soft Robotics, Delft University, 2022. (invited speaker).
- B. Caasenbrood, "`SOROTOKI`*: an Open-source Toolkit for Soft Robotics written in MATLAB*," IEEE International Conference on Soft Robotics, RoboSoft 2022, Edinbrugh. (poster presentation). `Best Poster Award`
- B. Caasenbrood, C. Della Santina, and A. Pogromsky, "*Workshop on Model-based Control of Soft Robots*," European Control Conference (ECC), 2021. (main organizer).
- B. Caasenbrood, talk on "*Towards Design and Control of Soft Robotics*," 4TU Symposium on Soft Robotics (digital), 2020. (invited speaker).
- B. Caasenbrood, talk on "*3D-printed Soft Robotics*," Symposium on Robotic Technologies, 2019. (invited speaker).
- B. Caasenbrood, A. Pogromsky and H. Nijmeijer, talk on "*Forward Dynamics of Hyper-elastic Soft Robotics*," 39th Benelux Meeting on Systems and Control, 2019. (abstract).
- B. Caasenbrood, A. Pogromsky and H. Nijmeijer, talk on "*Dynamical modeling and control of continuum soft robots*," 37th Benelux Meeting on Systems and Control, 2018. (abstract).

Curriculum Vitae

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum. Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.