UNIVERSITY OF TWENTE.

MSc Programme: Systems & Control

Model-Free Reinforcement Learning Control of a Pneumatic-Driven Soft Continuum Robot

Konstantinos Karytsas

M.Sc. Thesis

November 2023

Supervisors:
prof. dr. ir. Bojana Rosic
Minke Berghuis, MSc
Vasos Arnaoutis, MSc

Applied Mechanics & Data Analysis Research Group
Faculty of Engineering Technology
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

Preface

The utilization of artificial intelligence for robot control has been an intriguing area of research for me since the onset of my post-graduate studies in Systems & Control. Therefore, conducting my Thesis research on the application of Reinforcement Learning for the challenging task of controlling a soft continuum robot was a completely natural choice for me.

The research presented in this Master of Science Thesis was conducted in the Applied Mechanics & Data Analysis (AMDA) research group of the University of Twente, under the supervision of Professor Bojana Rosic, who I would like to thank for the opportunity she gave me to focus on a subject that I find truly fascinating. I would also like to thankfully acknowledge the encouragement and assistance of PhD candidates, M.W.Berghuis and V.Arnaoutis. Last but not least, I would like to thank my family for their continuous support.

Summary

Soft continuum robots constitute a category of inherently compliant robots which can interact safely with their surroundings. Therefore, they are considered a promising technology in the biomedical field with possible applications in laparoscopy and endoscopy, among others. Unlike rigid-link robots which have definite kinematic mapping, the accurate analytical modeling of soft continuum robots is difficult, especially when considering their non-linear deformation when actuated, the non-linear elasticity of their materials, and their susceptibility to interactions with the environment. Hence, soft continuum robots face modeling errors which lower the performance of model-based control. The scope of the present thesis is the development of a model-free reinforcement learning control scheme in order to control a pneumatic-driven soft continuum robot in 3D space. A physical simulation environment which is based on Cosserat rod theory and discrete differential geometry is used for policy training data generation while Proximal Policy Optimization is utilized for policy optimization (i.e. controller optimization).

Contents

Chapter 1

Introduction

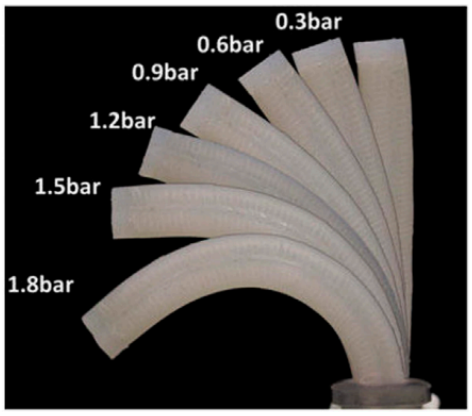
In the present research, the 3D position control of a single-segment, multi-backbone and pneumatic-driven soft continuum robot is investigated. The inherent compliance of the examined soft continuum robot allows it to safely interact with its environment, rendering it a promising technology for laparoscopy or endoscopy. Due to the fact that such robots lack definite kinematic mapping, their accurate control constitutes a challenging task. The scope of the present MSc thesis is the development of a Model-Free Reinforcement Learning control scheme that could be utilized in order to efficiently control the provided soft robotic device. In this chapter, soft continuum robotic systems are analyzed as well as their state-of-the-art control strategies. Finally, the motivation for the control strategy proposed in the present thesis is discussed.

1.1 Soft Continuum Robots

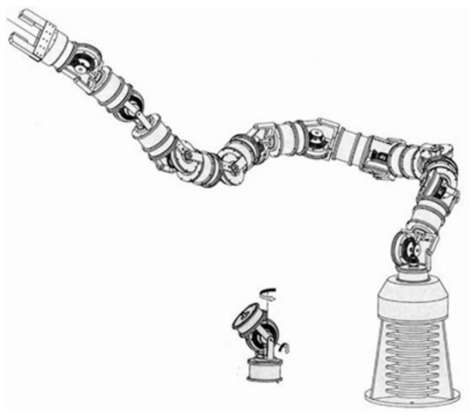
Soft continuum robotics constitute a relatively recent sub-field of robotics, since it has been pioneered in the mid 1980s by James Wilson at Duke University [1–3]. As its name suggests, it is concerned with the design, fabrication and control of continuum robotic systems which are built from highly compliant and flexible materials [4, 5]. Since soft continuum robots are a sub-category of continuum robots, continuum robots are discussed first.

Continuum Robots

Continuum robots are robotic systems which are characterized by (potentially) infinite degrees of freedom/number of joints [6] and are designed to mimic the behavior of biological systems such as tentacles, snakes or elephant trunks. Hence, continuum robots constitute a special category of inherently compliant robots, characterized by their dexterity as well as their ability to adapt [7]. According to J. Burgner-



(a) Continuum robot with curvilinear elastic structure. Image from [10]



(b) Hyper-redundant serial robot with 10 rigid joints. Image from [11]

Figure 1.1: Continuum Robots Vs Hyper-Redundant Robots

Kahrs et al. (2015), "A continuum robot is an actuatable structure, whose constitutive material forms curves with continuous tangent vectors" (p.1262) [8]. More specifically, in contrast to rigid-link robots which are constructed with discrete joints, continuum robots have a curvilinear elastic structure which renders the adjustment and modification of their shape at any point along their bodies possible (Figure 1.1a). This special characteristic makes continuum robots suitable for deployment in complex environments or space-limited environments (e.g. inside the human body), where the use of standard rigid-body robots is impossible [9].

Based on the above, the continuum robots' core structures which continuously curve is what differentiates them, not only from standard rigid-link robots, but from hyper-redundant manipulators as well [8]. In particular, the possible shapes of the hyper-redundant manipulators, only approximate curves with continuous tangent vectors, through the use of multiple, discrete, rigid links and joints (Figure 1.1b).

Classification of Continuum Robots based on Structure, Number of Segments and Actuation Mechanisms

The core structure of a continuum robot is called backbone and it deforms compliantly when external forces/torques are applied [12]. Based on the design of the core structure, continuum robots are classified in the following three categories:

- Single-Backbone Continuum Robots: The core structure of the continuum robots of this category consists of one central elastic backbone. Thus, the transmission or actuation elements run through it [12].
- Multi-Backbone Continuum Robots: In this case, the core structure of the continuum robot consists of two or more elastic elements which can be either

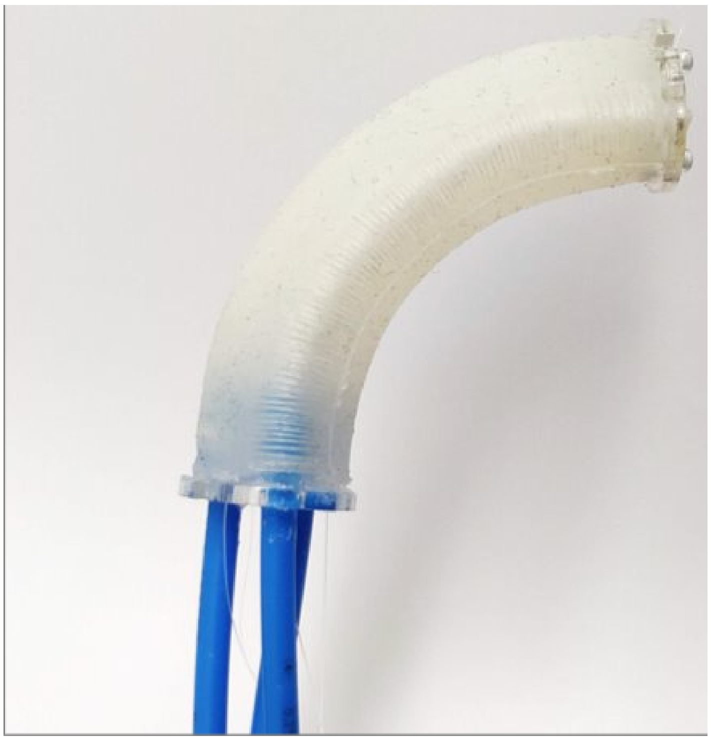


Figure 1.2: The investigated soft continuum robot. It consists of a single segment which is intrinsically actuated via three internal elastic pneumatic chambers. Therefore, its core structure is composed by four elastic elements i.e. the three soft actuators (blue rods) plus the main backbone. Hence, the investigated device constitutes a pneumatic-driven, single-segment, multi-backbone continuum robot.

tubes or rods. They are placed in parallel to each other while being connected to each other through some type of fixture [13].

- Concentric-Tube Continuum Robots: In this category, the robot's backbone consists of multiple concentric tubes.

The body of a continuum robot may consist of one or more segments. Hence, based on the number of segments that compose their body, continuum robots are classified in two categories. Single-segment continuum robots and multi-segment continuum robots [14].

The actuation mechanisms of continuum robots, based on their location, are divided in two categories [15]: the intrinsic actuation mechanisms and the extrinsic actuation mechanisms. The intrinsic mechanisms have actuators which are located inside the robot's mechanism (Integrated Actuators) [16]. More specifically, the actuators of intrinsic mechanisms, are located within the continuum backbone. The backbones of such robots, are usually composed by their intrinsic actuation mechanisms. Typical examples of intrinsically actuated continuum robots are the following [17]:

- Pneumatic-actuated continuum robots which deform when internal elastic chambers are inflated.
- Fluidic elastomer actuated continuum robots. In this case, the actuation mechanism consists of a pneumatic network of channels in an elastomer. In order to make the robot's body bend, these channels are filled with a pressurized fluid so that they expand. This expansion results in the robot's bending motion.
- Electroactive Polymers actuated Continuum Robots. Here, the robot's actuation is achieved through the utilization of Electroactive polymers (EAP) which are smart soft materials whose size and/or shape change when run through by electric current.
- Shape Memory Alloy actuated Continuum Robots. In this category of continuum robots, the contraction of Shape Memory Alloy material when its temperature rises, is exploited for actuation purposes. As soon as, it is cooled down, it returns to its initial state.

On the other hand, in the case of extrinsic mechanisms, external components are utilized in order to manipulate the body of the robot, i.e. the actuators are external to the robot. Since the actuators are located outside the continuum backbone (usually at the base of the robot), the size and weight of continuum robots with extrinsic actuation mechanisms are reduced [17]. Typical examples of extrinsically actuated continuum robots are the following:

- Tendon/Wire actuated continuum robots. In this case, tendons or wires are placed along the backbone and are fixed at fixtures in groups. The actuator, usually a motor at the base of the robot, applies forces to the tendons/wires in the robot base, resulting in the application of torques at the termination points. These torques cause the bending of the backbone.
- Concentric-Tube Continuum Robots. Continuum robots of this structure can be actuated only extrinsically. As stated before,these robots have a backbone that consists of multiple concentric compliant tubes (or rods). The actuation at the base of the continuum robot allows rotational and translational movement between the concentric tubes. [12].

Based on the analysis above, the investigated continuum robot is defined as a pneumatic-driven (i.e. intrinsically actuated), single-segment, multi-backbone continuum robot (Figure 1.2).

Applications of Continuum Robots and their Structural Limitations

In their early stage, continuum robots were characterized by a large structure since they were found suitable for certain industrial applications such where high robot flexibility and dexterity are required (e.g. grasping) [18]. Furthermore, the applicability of continuum robots for human-rescuing operations in space-limited environments has been verified as well [19]. However the high flexibility of continuum robots come at the cost of low payload which imposes considerable structural limitation to these robots [15].

From Continuum Robots to Soft Continuum Robots

As the scale of continuum robots reduced, and the focus shifted to delicate manipulation and safe human-robot interaction, the category of soft continuum robots emerged [15]. Soft continuum robots share all the characteristics of continuum robots while being composed of highly compliant and flexible materials (i.e soft elastic/ malleable materials). Therefore, the bodies of soft continuum robots are built mainly from soft materials (e.g. polymers). In case of soft continuum robots with intrinsic actuators (as in the investigated case), soft actuators are also utilized. In general, the integrated sensors of soft continuum robots should be soft or at least flexible [20]. Hence, soft continuum robots constitute a category of inherently compliant robots which can interact safely with their surroundings [4]. The inherent compliance and flexibility of soft continuum robots render them significantly safer for human-robot interaction than their rigid-link counterparts, thereby giving them a significant advantage [5]. Therefore, soft continuum robots are considered a promising technology in the biomedical field with possible applications in minimally-invasive surgery. In particular, the inherent compliance of soft continuum robots coupled with their shape changing properties, renders them capable for navigating inside the human body while making the operation less invasive [8, 21]. Apart from minimally invasive surgery, soft continuum robots proved to be a promising technology in the fields of exo-suits and deep-sea exploration [22, 23].

Considering the above, soft continuum robots are highly adaptable in unstructured environments and characterized by increased safety in terms of physical interaction with humans [24]. These characteristics render them a promising technology in surgical assistance, rehabilitation and environmental exploration among others.

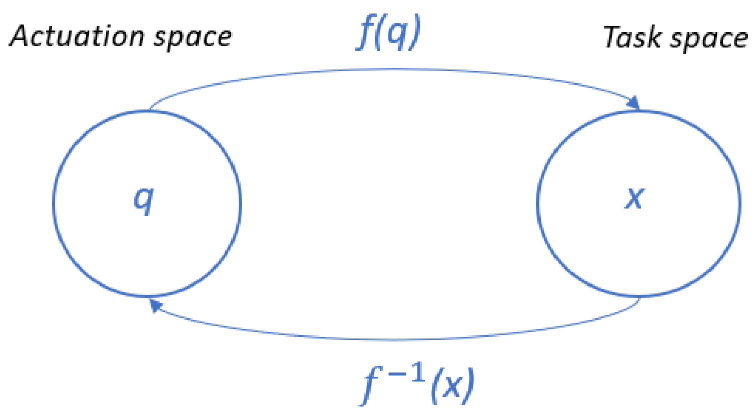


Figure 1.3: Kinematic mapping in rigid link robots. $f(q)$ represents the robot's forward kinematics while $f^{-1}(x)$ represents the inverse kinematics. Actuation space is essentially the joint space (actuation through prismatic/revolute joints).

1.2 Challenges in Soft Continuum Robot Control

A large portion of soft continuum robot related research is focused heavily on the design of control schemes that render the accurate control of these robotic devices, possible. However, soft continuum robot control is considered a highly challenging control problem due to the softness of the robot's material as well as the need for robot-specific controllers and integrated sensors. In particular, soft continuum robot control is highly dependent on integrated sensors which measure the robot's state as well as the environment's state [25]. However, sensor integration is particularly challenging in soft continuum robots since these sensors have to stretch, bend and deform along with the robot, while not restricting the robot's movement [20]. Furthermore, in soft robot control, the control system design and the sensor selection, depends on the utilized actuation mechanism. Thus, each actuation mechanism (fluidic, electric, magnetic etc.) requires different controllers as well as different sensors. Furthermore, the soft materials that compose soft continuum robots, present non-linear and time-dependant properties [25]. Hence, unlike conventional rigid-link robots, which are controlled directly by the actuation produced by their joint motors and have definite kinematic mapping (Figure 1.3), soft continuum robots lack definite mapping between actuation space and task space [15]. In other words, in rigid-link robots the relationship between the configuration space and the task space is linear whereas in soft continuum robots the aforementioned relationship is non-linear due to material elasticity.

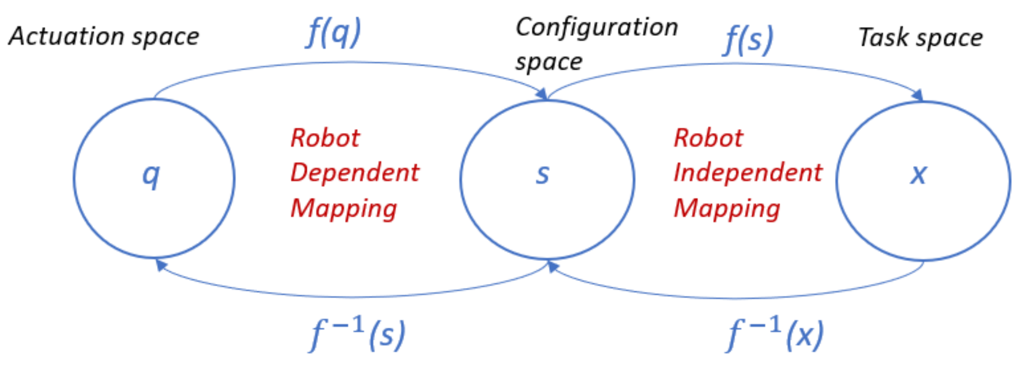


Figure 1.4: Kinematic mapping in soft continuum robots.

Naturally, the accurate analytical modelling of soft continuum robots is considered extremely difficult due to the elasticity of the robot's material, the non-linear deformation of the robot when actuated, and potential collisions with objects in their environment. Due to the fact that soft continuum robots are highly compliant, their interaction with an obstacle would cause deformation to their bodies, which could result in undesired configurations regardless of the actuation commands [15]. Sensing the deformation effects due to a collision and feeding them back to the model is challenging and stability issues can emerge. It is also noteworthy that considerable modelling uncertainties are often present in the case of soft continuum robots, since these robots are susceptible to wear-out effects which occur over time and can alter their characteristics. Additionally, even though analytical models of soft continuum robots exist [25] and they could be utilized for closed-loop position control in the examined case, their dependence on integrated sensors and limited universality constitute deterring factors. In particular, measurement systems for actuator deformation (i.e. measurements of actuator length/strain/curvature) and end-effector displacement would be needed as well as a mathematical model of the utilized pneumatic actuators.

Finally, the objective of the present thesis is end-effector position control in task space. Thus, in a closed-loop scenario, the inverse kinematics of the soft continuum robot would play a significant role (i.e. start from the desired position in task space and determine the needed actuation). Nonetheless, even when using analytical models, the calculation of inverse kinematics is challenging as there is no definite number of solutions and a closed-form solution might not even exist [7]. This can be understood when examining the kinematic mapping of soft continuum robots' analytical models (Figure 1.4). In these models, the modelling of continuum kinematics is based on an evolving frame along a continuous backbone which is parameterized by its arc length. These arc parameters are essentially the coordinates of the configuration space thereby fully describing the robot's shape. The mapping from the actuation space to the configuration space (and vice versa) is non-linear and depends on the soft continuum robot's design and actuation mechanism (robot dependent mapping). On the other hand, the mapping from configuration space to task space is rendered possible via the arc parameters and it is independent of the soft continuum robot's design. Thus, the calculation of inverse kinematics when using (mapping from task space to actuation space) leads to non-linear equations with often no analytical solution.

1.3 Data-Driven Soft Continuum Robot Control

Due to the aforementioned challenges in soft continuum robot control, data-driven control techniques were extensively researched in the recent years. In fact, data-driven control proved to be an effective way for controlling such robotic devices due to the fact that no prior physics-based analytical modelling is required [15]. This is the case, because in data-driven control, the inverse kinematics are estimated based on experimental data obtained by sensors, thereby circumventing the need for parameterizing the soft continuum robot with differential equations. Naturally, since physics-based analytical modelling can be avoided in data-driven control, data-driven methods are less computationally complex. The main groups of control strategies that have been widely utilized for soft continuum robot control, and are based on data-driven control, are the following [15]:

- Iteration-based Kinematic Model-Free Control
- Control based on Supervised Learning
- Hybrid Control
- Reinforcement Learning-based Control

Although physics-based analytical modelling is limited in data-driven control, modelling and simulation of soft continuum robots is required for the application of hybrid control or Reinforcement Learning based control since they are dependent either on prior knowledge of the robot's forward dynamics/kinematics or on simulation data. Therefore it is important to mention some common simulators used for modelling and simulation of soft continuum robots:

- Simulators based on Finite Elements Modelling (FEM) [26]. Since Finite Element Modeling (FEM) is considered an effective method to model soft robotic systems, FEM simulators have been widely used for soft continuum robot design and modelling [26]. FEM constitutes a general numerical method, which can be used to solve the partial differential equations describing the dynamics of soft continuum robots based on a continuum mechanics model, thereby addressing the non-linearities that occur due to large mechanical deformations as well as material non-linearities and contact non-linearities. Two popular continuum mechanics models for soft continuum robot modelling are Kirchoff-Love models and Cosserat models. More specifically, in Kirchoff-Love rod theory, the soft continuum robot is modelled as 1D slender rod which can only bend and twist. Thus, it is assumed that the length of the rod is much larger than the radius at any cross-section and therefore the rod's (robot's) dynamical behavior can be approximated by averaging balance equations at every cross-section [27]. In Cosserat-Rod theory the bodies of soft continuum robots are Cosserat rods, which constitute an extension of Kirchhoff rods. In particular, Cosserat rods share all the characteristics of Kirchhoff rods but apart from bending and twisting, they can also account for stretching and shearing at every cross-section [27]. Consequently, FEM simulators are used to solve the partial differential equations that emerge, by dividing the soft continuum robot into smaller, simpler parts called finite elements [28]. Popular soft robot simulators based on FEM are ”SOFA” and ”SAMCEF (MECANO)” but of course ”ANSYS”, ”COMSOL” and ”ABAQUS” can also be utilized for soft robot simulation.

- Simulators based on Discrete Differential Geometry [29], which are essentially numerical simulation tools for soft continuum robots based on a discrete differential geometry computational framework [29]. Discrete Differential Geometry is the analysis of fundamental geometric concepts (e.g. curvature) through computational and mathematical tools.

- Simulators based on Cosserat-Rod theory and Discrete Differential Geometry [30]. In these simulators, the Cosserat model is consistently discretized based on discrete differential geometry. The open-source soft robot simulation software ”Elastica” belongs in this category [30].

- Simulators based on Ritz-Galerkin method and Cosserat theory [31]. The basis of these simulators is the parameterization of the Cosserat rod's configuration by vector fields such as the strain vector field. In that way, the curvature of the soft robot's body is modelled, and the soft robot's dynamics reduce to a set of Lagrange Ordinary Differential Equations in time [31]. A well-known soft continuum robot simulator of this category, is MATLAB's ”SoroSim”.

It is important to clarify that the feasibility of each of the aforementioned soft continuum robot simulators depends on the user's specific needs and goals. In general, simulators based on continuum mechanics theories (e.g. ”SoroSim”, ”Elastica”, ”SOFA”) are consistent in modelling the geometry, dynamics and kinematics of soft continuum robots. Therefore, in the context of the present thesis, the focus is concentrated on these simulators. In particular, FEM-based soft robot simulators (such as ”SOFA”) achieve the highest modelling accuracy and in contrast to the rest, they are also able to model 3D complex systems in a multi-physics context [31]. Hence, from a designer's perspective, FEM-based simulators are the most suitable choice since they can be utilized for soft robot actuation design as well. On the other hand, from a control engineering perspective, such simulators are deemed unsuitable, since they are particularly computationally expensive and their adaptation to soft continuum robot control is difficult because of the extremely large number of DoF of the produced models. Simulators based on continuum mechanics and Ritz method (such as ”SoroSim”) are most efficient for 1D problems that involve modelling of systems which can be represented as beams with small cross-sectional inflation [31]. The reduced complexity of the produced models in this case, render their coupling with control possible. Similarly, simulators based on continuum mechanics and discrete differential geometry (Elastica) are suitable for 1D problems. ”Elastica” offers a fast real-time interactive soft continuum robot simulation which can be adapted in order to be coupled with control algorithms [30].

In the present thesis, the objective is soft continuum robot control based on Model-Free Reinforcement Learning. On one hand, computational expenses are particularly important since this type of control is based on a large number of interactive simulations and thus simulators that offer fast interactive simulations are desired. On the other hand, a continuum mechanics simulator is also desired in order to ensure consistent modelling at all levels. Therefore, both SoroSim and Elastica are deemed suitable choices. However, Elastica is chosen as the physical simulation environment in the present thesis, since it is an open-source software and it has already been deemed suitable for RL control [30].

1.3.1 Iteration-based Kinematic Model-Free Control

As already mentioned, in rigid body robots there is a linear relation between actuation space and configuration space and thus a definite kinematic mapping between the actuation space and the task space exists. This is not the case with soft continuum robots due to the non-linear elasticity of their material. Simultaneously, the analytical models of soft continuum robots, only approximate the soft continuum robot's kinematic mapping using certain assumptions (e.g. constant curvature assumption). To address this problem, model-free approaches which are based on iterative optimization have been developed in [32–35]. These approaches are based on the estimation of the Jacobian matrix (at each time-step) in order to obtain the robot-dependent mapping. Despite the fact that the Jacobian matrix changes non-linearly while the soft continuum robot is moving, in the works presented in [32–35] it is stated that within a control interval such changes can be considered linear if the investigated continuum robot moves slowly.

Optimization-Based Jacobian Matrix Estimation

In the works presented in [32, 34, 35], optimization-based Jacobian matrix estimation is utilized for the development of closed-loop model-free control schemes for tendon-driven continuum robots. Therefore, the goal in the aforementioned approaches is the update of Jacobian matrix at every time-step in order to obtain the continuum robot's inverse kinematics. Thus, the idea is the following: Assuming that the robot has n-DoF design and each one is independent from the other, the Jacobian matrix is initialized by actuating the n-DoFs successively with a small amount while keeping track of the resulted displacements. Hence, the Jacobian matrix can be reconstructed and updated at every time-step by solving an optimization problem via quadratic programming. The optimization problem is essentially the calculation of the smallest change of the Jacobian matrix which ensures that:

- the Jacobian matrix maps the small change in actuation to the small change in End-Effector task position linearly, during a control interval.
- the Jacobian matrix of the current time step is equal to the Jacobian matrix of the previous time step plus the change in the Jacobian matrix.

It is important to mention that in the discussed cases [32, 34, 35], tendon-driven continuum robots are examined, in which the tension of each tendon can be measured with force sensors in order to calculate the minimum change in actuation command needed. Hence, even though the strategies followed in [32, 34, 35] may be valuable for tendon-driven soft continuum robots, they are deemed unsuitable for the pneumatic-driven soft continuum robot of the present thesis.

Methods Utilizing Adaptive Kalman Filter

Non-linear Kalman Filters have also been proposed for effective soft continuum robot control. More precisely, an approach that does not involve kinematic modelling has been introduced in [33]. In particular, in this case, the change in the elements of the Jacobian matrix is addressed as the stochastic system's process noise. Therefore, an adaptive Kalman Filter is utilized for the estimation of the Jacobian matrix entries. The calculation procedure is based on the utilization of an adaptive covariance matrix for the process noise $\eta(k)$ which results in a Kalman Filter with improved convergence as well as tracking performance against system uncertainties. Finally, the desired configuration that corresponds to the minimal robot deformation (thus minimal actuation change) is defined based on the updated Jacobian matrix.

1.3.2 Soft Continuum Robot Control based on Supervised Machine Learning

Supervised Learning algorithms are also utilized for soft continuum robot control since they are able to learn the mapping from the actuation space to the task space [15]. More precisely, this group of control algorithms use Supervised Machine Learning techniques in order to estimate the inverse Kinematics/Statics of soft continuum robots. Similarly to the iteration-based model-free control schemes, supervised learning control requires only actuator information and a sensor that can provide information regarding the position of the end-effector in task space. Alternatively, information for other control objectives, like the robot's configuration in the task space, can be used. Hence, it can be concluded that in supervised learning soft continuum robot control, dependence on sensors is low.

For context, assuming that all forces acting on the robot are at rest at equilibrium, the configuration of the robot is described by a statics model. On the other hand, a kinematics model does not take into consideration the applied forces and the robot movement is described explicitly geometrically, in terms of its functional dimensions, degrees of freedom, its workspace and its positional constraints/capabilities. Essentially forward kinematics are used to map the actuator velocity to end-effector velocity. If cable-driven soft robots are examined, the tension of the cables are mapped to the position of the tip via forward statics. Therefore, in this case, the inverse statics are used to map the tip position to cable tensions. Thus, the inverse statics calculate the needed cable tensions so that the tip of the soft robot reaches the desired position [36, 37]. It is worth mentioning that open-loop control schemes are based on the calculation of inverse statics, whereas feedback closed-loop control is also dependent on the calculation of the inverse kinematics.

In the corresponding scientific literature, a variety of machine learning-based control techniques has been proven promising or even validated for soft continuum robot control [15]. An analysis on them is presented in the following pages.

Regression-Based Methods for Mapping Approximation

Neural Networks constitute a regression method that is often utilized for inverse

Kinematics/Statics estimation [15]. Researchers in the field focused mainly on regression methods to achieve accurate mapping approximation. Apart from the typical Feed-Forward Neural Networks, three regression methods were examined:

- Extreme Learning Machines (ELMs) [38]. Extreme learning machines are feed-forward neural networks, commonly used for regression, classification, clustering, feature learning etc. An extreme learning machine may consist of a single hidden layer or multiple hidden layers. What differentiates ELMs from standard neural networks is the utilization of the Moore-Penrose generalized inverse for setting the weight matrix. In particular, the weights between the input layer and the hidden layer do not require modification during the phases of training and predicting [38]. The algorithm is the following: Firstly, the weight matrix between the input and the hidden layer is randomly assigned by the ELM. The same is done for the weight matrix between the hidden layer and output layer. Subsequently, the ELM randomly assigns values for the bias of the hidden layer. Then, an activation function in the hidden layer is selected. Finally, the utilization of the least square optimization method, calculates the weight matrix between the hidden layer and output layer that produces the minimum error.

- Locally weighted projection regression (LWPR) [39]. This supervised learning algorithm is able to produce non-linear regression models, i.e the approximation of non-linear relationship between input and output data is possible. Therefore, high dimensional data spaces are not limiting the regression abilities of such models. The basis of LWPR is the utilization of locally linear models, while univariate regressions are spanning them in the input space [39].

- Gaussian process regression (GPR) [40]. This algorithm also constitutes a non-parametric approach to regression, but it is based on Bayes' rule. What differentiates GPR from other regression methods, is the prior specification of a probability distribution over all the possible values of the fitted function's parameters. During the training stage, probabilities are changed based on the observed data, using Bayes' Rule [40].

The last two proved promising for the soft continuum robot control since they were deemed satisfying in terms of performance [10, 41, 42]. Feed-Forward Neural Networks have been found capable for estimating forward and inverse statics and can be used for open loop soft continuum robot control where online sensors are unavailable [15]. Nevertheless, forward/inverse kinematics constitute relative mappings, the calculation of which is usually difficult for Feed-Forward Neural Networks. Thus, regression methods such as LWPR and GPR are utilized for Kinematics approximation since these algorithms typically converge more stably. Furthermore, when coupled with feedback from sensors, closed-loop control can be implemented, leading to more accurate control [10, 37, 41, 42]. An advantage of regression models is the fact that they render online update of data possible. However, this comes with higher computational expenses as well. For instance, calculating the inverse of a high-dimension matrix for every model refinement time is necessary when utilizing Gaussian Process Regression. The utilization of the local Gaussian Process Regression however, can lead to high computational speed [42]. In such a case, the workspace is divided into several sections (each consisting of less than 300 samples for example) for the purpose of training and updating the model independently.

1.3.3 Hybrid Soft Continuum Robot Control

Hybrid controllers which combine analytical modelling and learning-based methods, are considered effective for soft continuum robot control. In [43] a control strategy based on fuzzy-neural control is utilized in order to control a flexible manipulator consisting of multiple links. More specifically, a fuzzy controller constituted the primary loop and determined the control actions while a neural network forming a secondary loop, was used for compensating the impact of coupling. Furthermore, a control scheme that combines model predictive control and model-free iterative learning has been developed in order to make a soft robotic glove to track a trajectory [44, 45]. In the studies of [46, 47] it is validated that the learning-based component of the hybrid control schemes are able to compensate for errors in analytical modelling.

1.3.4 Reinforcement Learning-Based Control of Soft Continuum Robots

Due to the recent developments in Artificial Intelligence, Reinforcement Learning-based robot control has drawn the attention of the researchers of the field. Reinforcement Learning (RL) constitutes a framework in which the robot learns autonomously to perform new tasks while interacting with its environment. Hence, by utilizing RL the robot can execute a complicated task in a complex and dynamic environment thereby surpassing limitations that may be present in conventional control strategies [15]. Reinforcement Learning robot control based on whether there is prior knowledge of the state transition dynamics, can be divided in two main categories: Model-Based Reinforcement Learning and Model-Free Reinforcement Learning.

Model-Based RL for Soft Continuum Robot Control

Model-based RL for soft continuum robot control depends on the prior knowledge of the soft continuum robot's forward dynamics and kinematics, since model-based RL is applicable on problems where the transition model (i.e transition probability matrix) and reward function are known a-priori or they are approximated [15, 48]. Therefore, in model-based RL soft continuum robot control, the robot's forward model, i.e. transition model, is either fit to real robot-environment interaction data or analytical soft continuum robot models based on geometric assumptions are used to derive the forward model. Once the forward model is available, it is utilized for policy training data generation. The main advantage of model-based RL is that the policy trained on a forward model which is derived from real data, can perform stably when transferred to the physical system. This is to be expected since data from physical interactions are more realistic compared to data derived from simulation, where a simplified virtual model is utilized. Nevertheless, it is important to clarify that model-based RL soft robot control presents significant differences compared to Reinforcement Learning-based control for conventional joint-linked robots since it is associated with certain challenges which are not present in the case of conventional joint-linked robots. In particular, apart from the difficulties which are associated with the quality (presence of noise) and the collection (retrieving sensor data) of real robot-environment interaction data. On top of that, frequent use of the soft continuum robot for data collection purposes, can cause damages to the robot [15, 49].

Model-based RL control for a simulated tendon-driven soft robotic arm has been proposed in [50]. The goal was to teach the manipulator to reach dynamic targets. The forward model was derived using experiment data and utilizing a non-linear autoregressive network with exogenous inputs (NARX). The optimal action was given directly by the policy trained on sampled trajectories. In [15], Deep Q-Learning was utilized in order to control the position of a cable driven soft robotic arm. The robotic arm was modelled using experimental data.

Model-Free RL for Soft Continuum Robot Control

The significant challenges associated with model-based RL soft robot control shifted the focus of the research on model-free RL control techniques. In contrast to the model-based RL approach, model-free RL does not require prior knowledge of a transition model (i.e. transition probability matrix) and policy optimization is achieved through the robot's interaction with an unknown environment. In other words, the optimal behavior is learned through interactions with an unknown environment. More precisely, in model-free RL soft continuum robot control, the robot and its environment are modelled using a suitable soft robot simulator, and virtual robot-environment data coming from the simulation are used for policy training. Therefore, the utilization of virtual data instead of real data for policy training, bypasses the problems associated with real soft robot-environment interaction.

An example of model-free RL control scheme, which is implemented for controlling a soft robotic arm with multiple segments, has been reported in [51]. The objective was controlling the robot so that it is able to reach the desired target in 2D space. Q-learning along with data derived from a simulation were utilized for policy training. The developed control scheme was deemed effective and robust [51]. In the work presented in [52], the same soft manipulator was used but with a more complex objective. More specifically, a control algorithm was developed in order to teach the robot to open a drawer and rotate a wheel. The designed control system consisted of the RL controller trained by Q-learning algorithm for motion control purposes and two more controllers. It was demonstrated that the proposed control scheme can be effective for tasks involving robot-environment interaction within a dynamic environment. Problems with higher dimensionality have also been examined however, since most real world soft robot applications require spatial control. For instance, a 3D open-loop position control scheme for a soft manipulator, has been proposed in [53]. The soft robot was modelled using Cosserat Rod theory and policy training data were obtained from the corresponding simulation. In this case, the RL algorithm that was utilized was Deep-Q Network (DQN) and the developed strategy presented high accuracy according to the authors. Moreover, Dueling DQN (DDQN) has also been examined and was utilized in order to teach a soft catheter to steer inside a model of the human heart [54]. Multi-agent Temporal Difference control has also been utilized by researchers in order to teach a soft manipulator to provide bathing assistance [55]. In this case, each robot's actuator is considered an agent while all of them interacting with the same environment and policy evaluation is done using the state-action-reward-state-action (SARSA) algorithm. Furthermore, in [56], a closed-loop position control scheme for a soft continuum manipulator was developed. The basic improvement of this study compared to [53], is the inclusion of feedback from vision sensors as well as the utilization of the deep deterministic policy gradient (DDPG) model-free RL algorithm. The introduced improvements enabled the examined soft robot to follow sophisticated trajectories with positional error decrease due to the closed-loop formulation. Additionally, a control scheme based on proximal policy optimization (PPO) and a central pattern generator (CPG), was used for 2D position control of soft robot snakes in [57]. In [30], the model-free RL control of a soft continuum robot with internal continuum actuation is examined. The robot is modelled as a discretized Cosserat rod with the use of a suitable simulator and the produced simulation is coupled with five deep RL algorithms. The objective is to use model-free RL control in order to teach the robot to perform a series of control tasks such as tracking a moving target in 3D space, reaching a target with a specific orientation or reaching a target while moving through a series of physical obstacles. Finally, the soft continuum robot in this case is continuously actuated through a continuous action functions which produces internal torques along the robot's body. The proposed actuation can make the robot bend and twist but not elongate.

The existing literature on model-free RL soft continuum robot control, proves the feasibility of such approaches for the problem examined in the present thesis.

1.4 Motivation for the proposed control strategy & Novelty

In the present research, the effective 3D position control of a single segment, multi-backbone, pneumatic-actuated soft continuum robot is investigated. Due to its design, the investigated robot is associated with modelling errors due to intrinsic and extrinsic factors, i.e. no definite relationship between actuation space and task space, undesired configurations due to interactions with the environment and wear-out effects, which seriously limit the performance of model-based control strategies. The aforementioned limitations coupled with the recent developments in Reinforcement Learning-based control constituted the motive for the proposed approach in the present thesis. More precisely, Reinforcement Learning-based soft robot control is promising for the following reasons:

- Prior knowledge of the soft continuum robot's configuration can be avoided [15].
- In contrast to Supervised Learning-based soft robot control, where modelling is limited in a specific workspace, Reinforcement Learning-based soft robot control can expand manipulation to a complex or/and dynamic environment [49].

Reinforcement Learning enables the soft continuum robot to learn from experience while interacting with the environment. Thus, problems regarding the collection and the quality of the obtained data can occur. Additionally, damages from the frequent movement of the soft robot can occur due to its pneumatic actuation. To avoid the aforementioned issues, Model-Free Reinforcement Learning control is chosen in the present research, where virtual data obtained from a suitable physical simulation environment are used for policy training. However, in contrast to the works in [56] and [53], where static, semi-analytical models are used to train the RL agents, in this research, a dynamic Cosserat-Rod model is utilized. Therefore, the robot's soft-body physics as well as its dynamics, degrees of freedom and environmental contacts are modelled, while being combined with RL [30]. Furthermore, different from the works presented in [51, 52, 54], in this research a multi-discrete actuation space that allows elongation as well as bending in all possible directions in 3D space, is designed. The design of observation space differentiates also, since it is continuous in the present case. Similar to [57], Proximal Policy Optimization (PPO) is utilized but this time for solving a 3D position control problem. Finally, similarly to [30], Elastica software is utilized for virtual policy training data generation, however in the present thesis a considerably softer, smaller and more under-actuated soft robot is investigated. Additionally, in contrast to the implementation in [30], the actuation space in the present thesis is multi-discrete and the reward design is designed specifically in order to stabilize the robot's end-effector on the desired task space position. It is also worth noting that in the case of the present research the soft continuum robot is also able to elongate due to its actuation.

In the following chapter, i.e. Chapter 2, the applicability of model-free RL control for the investigated case is explained while the related fundamental theoretical concepts are introduced. Furthermore, the feasibility of a physical simulation environment based on Cosserat-Rod theory, for modelling and simulating the investigated soft continuum robot is discussed while the mathematical description of the utilized Cosserat-rod model is presented. In Chapter 3, the physical simulation's set-up as well as its adaptation for model-free RL control with Proximal Policy Optimization (PPO) are discussed in order to present the implementation details of the proposed approach. In Chapter 4, the obtained results are presented and analyzed while examining the impact of reward function design and certain PPO hyperparameters on them. Finally, in Chapter 5, the advantages and the potential of the developed control scheme are discussed, its limitations are pointed out and recommendations regarding future extension of the present research are given.

Chapter 2

Theoretical Background

The subject of the present thesis is the development of a model-free RL control scheme based on Proximal Policy Optimization in order to control a pneumatic-driven soft continuum robot control in 3D space. The investigated soft continuum robot is actuated by three soft pneumatic chambers which render bending and elongation of its soft body in 3D space possible (Figure 2.1). The robot's soft body is one single segment and its core is composed by multiple elastic elements (3 soft actuators + 1 central elastic element). Therefore, the examined robot can be characterized as a single-segment multi-backbone soft continuum robot with one of its ends fixed and the other one free (Figure 2.1). The goal is to develop a control scheme that moves the soft continuum robot's free end-point (i.e tip/end-effector) from point A to point B, in order to reach a target (and stay on the target) in 3D space (Figure 2.2).

In this section, the theoretical aspects of the tools that are utilized in the context of the present thesis, are analyzed. Firstly, the applicability of model-free RL control for the investigated case is explained while the related fundamental theoretical concepts are introduced. Secondly, the feasibility of a physical simulation environment based on Cosserat-Rod theory, for modelling and simulating the investigated soft continuum robot is discussed while the mathematical description of the utilized Cosserat-rod model is presented.

2.1 Control via Model-Free Reinforcement Learning

Reinforcement Learning (RL) constitutes a sub-field of Machine Learning where learning is based on a trial and error approach. In particular, the learning process is based on the feedback the agent receives when interacting with the environment. Of course in the present thesis the application of RL is focused on robot control.

In reinforcement learning robot control, the robot learns to take actions in an en-

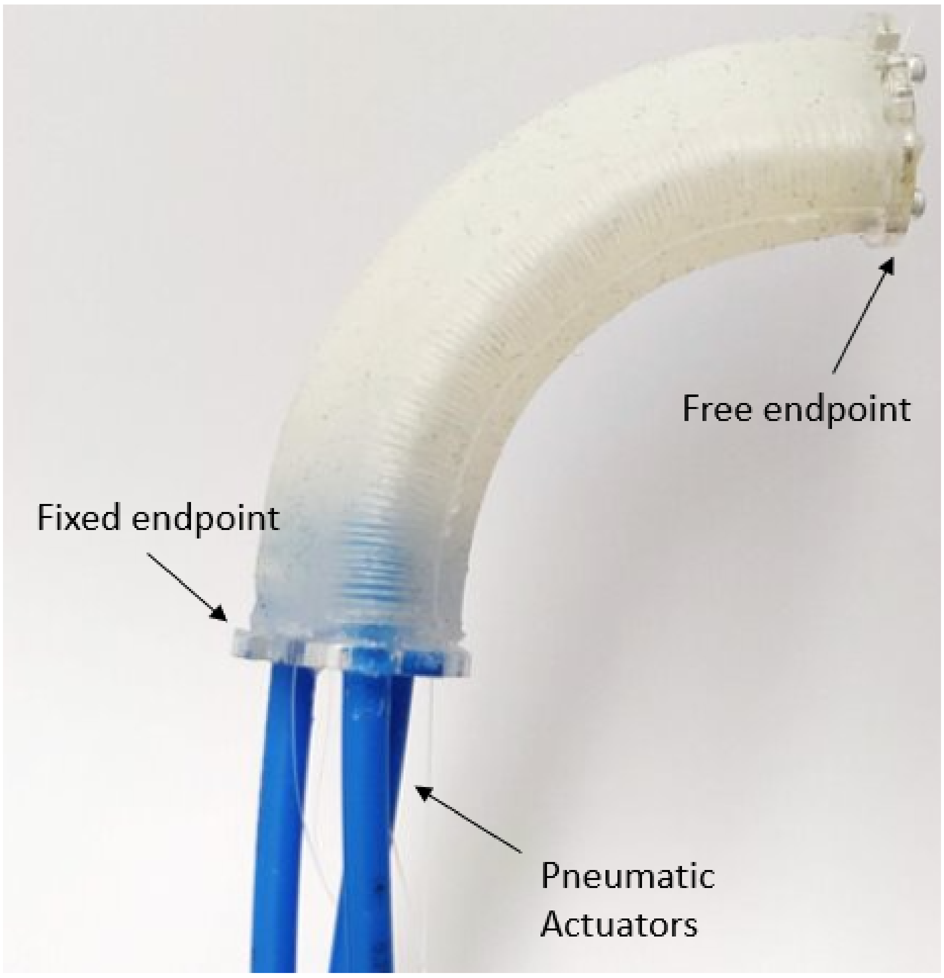


Figure 2.1: The investigated soft continuum robot.It consists of a single segment which is intrinsically actuated via three internal elastic pneumatic chambers. Therefore, its core structure is composed by four elastic elements i.e. the three soft actuators (blue rods) plus the main backbone. Hence, the investigated device constitutes a pneumatic-driven, single-segment, multi-backbone continuum robot.

vironment for the purpose of maximizing the receiving positive feedback [58]. This way, the robot can learn to perform the desired task in a way that fundamentally emulates the human way of learning. Consequently, reinforcement learning robot control can be defined as a sequential decision making process based on the reward hypothesis. The reward hypothesis states that all tasks can be described by the maximization of the expected cumulative reward starting from an initial state [59]. Therefore, reinforcement learning robot control is essentially an optimization problem since its goal is finding the optimal sequence of decisions that have to be taken by the robot in order to maximize the cumulative reward, i.e. perform the control task optimally. Taking into consideration the above general definitions, controlling

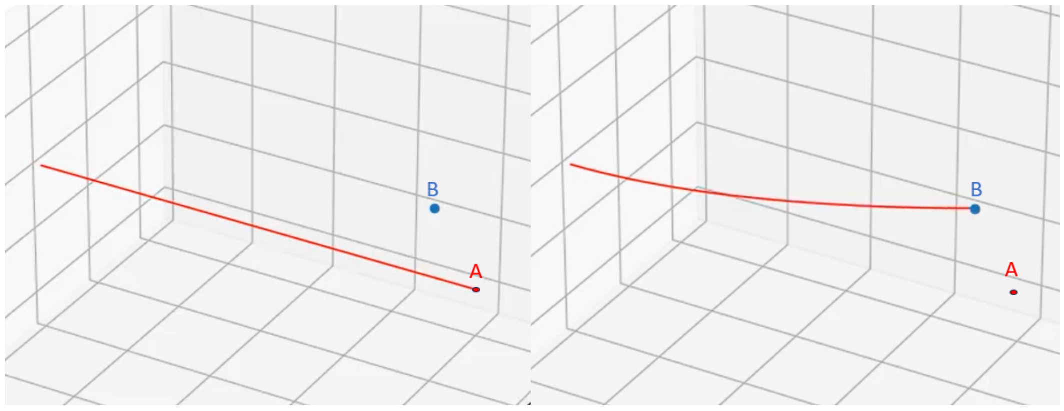


Figure 2.2: Schematic representation of the soft robot (red curve) moving its end-effector from point A (initial position of the end-effector) to point B (target).

the End-Effector position of the investigated pneumatic-driven soft continuum robot in 3D space using RL, essentially means finding the optimal sequence of decisions that minimize the euclidean distance between the robot's End-Effector and the target. However a mathematical representation of this decision making process is necessary. In general, optimization problems based on decision making can be modelled as Markov Decision Processes (MDPs) thereby rendering it possible for almost all RL problems to be formalized as MDPs [48]. The 3D position control

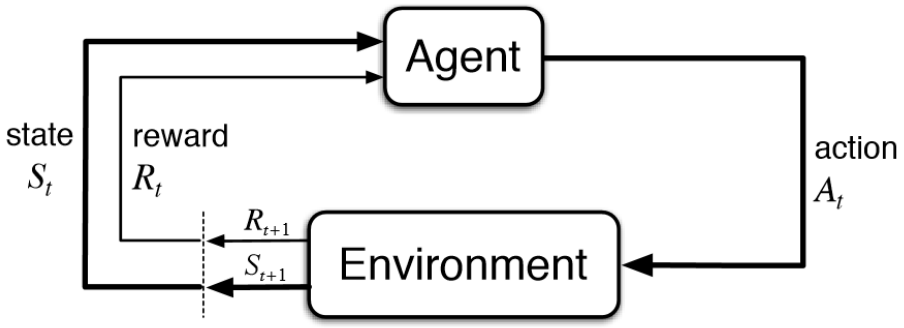


Figure 2.3: Agent-Environment interaction in Reinforcement Learning [60]

problem for the examined pneumatic-driven soft continuum robot is no exception. Hence, the soft robot's actions and reactions within the 3D task space can be formalized by an MDP. The MDP is defined as the tuple $(S, A, P(s'|s,\alpha), R, \gamma)$, where S is the set of states, i.e the state-space, A is the set of possible actions i.e. actuation space, $P(s'|s,\alpha)$ is the transition probability matrix, $R(s,\alpha)$ constitutes the reward function and γ is the discount factor [61]. For the examined problem, the MDP is characterized by a continuous state-space S and each state s can consist of the soft robot's position and velocity as well as the target's position (and velocity in case of a moving target). A finite set of actions A is defined in each state as well. Here, the soft robot can inflate its pneumatic chambers thereby increasing or decreasing or keeping constant the produced actuation forces in order to elongate/compress and/or bend in 3D space. The soft robot's selected action along with the stochastic nature of the environment (e.g. blowing wind, a moving obstacle etc), will determine the new state that the robot will end up. Thus this change of state is a random process. Nevertheless, the current state-action pair determines the distribution of the next possible states for the robot. This random process represents the soft robot's forward Dynamics/Kinematics and in MDP terminology is called transition probability matrix P. Each state transition is associated with a positive or negative reward. This is determined by the reward function R which defines the expected immediate reward the soft robot receives when taking an action α at state s. The policy $\pi(a|s)$ constitutes the mapping from the current state to the suggested action. The objective is the maximization of the expected future reward through the selection of the appropriate actions, while also introducing time-dependent discounting γ to give different weighting to future rewards. Therefore in the investigated RL control problem, the policy plays the role of the controller, and essentially the goal is optimizing the policy so that the state-action pairs with the highest cumulative reward are produced (i.e. robot's end-effector finds the best way to reach the target).

Summarizing the above, the examined RL control problem can be stated mathematically as an MDP which can be represented using by the tuple $(S, A, P(s'|s, \alpha), R, \gamma)$ where the state space S consists of the robot's and target's all possible positions and velocities, the set of possible actions A are described by the inflation of the robot's pneumatic chambers, $P(s'|s, \alpha)$ is the transition probability matrix, $R(s, \alpha)$ the manually designed reward function and γ is the discount factor.

Model-Free Approach

In the context of the present thesis, a model-free RL control scheme is selected for the 3D position control of the investigated system to avoid problems associated with the use of the actual physical system such as damages to the robot because of its interaction with the environment, partially observable state-space, stochastic nature of the environment which affects robot's dynamics and artifacts due to computer discretization [58].

In model-based RL soft continuum robot control, the robot's forward model, i.e. transition model, is either fit to real robot-environment interaction data or analytical soft continuum robot models based on geometric assumptions are used to derive the forward model. Subsequently, the obtained forward model of the robot is utilized in order to produce the sequences of states and actions, i.e. trajectory data:

$$\tau = [(s_0, a_0), .., (s_{T-1}, a_{T-1}), (s_T, .)] \tag{2.1}$$

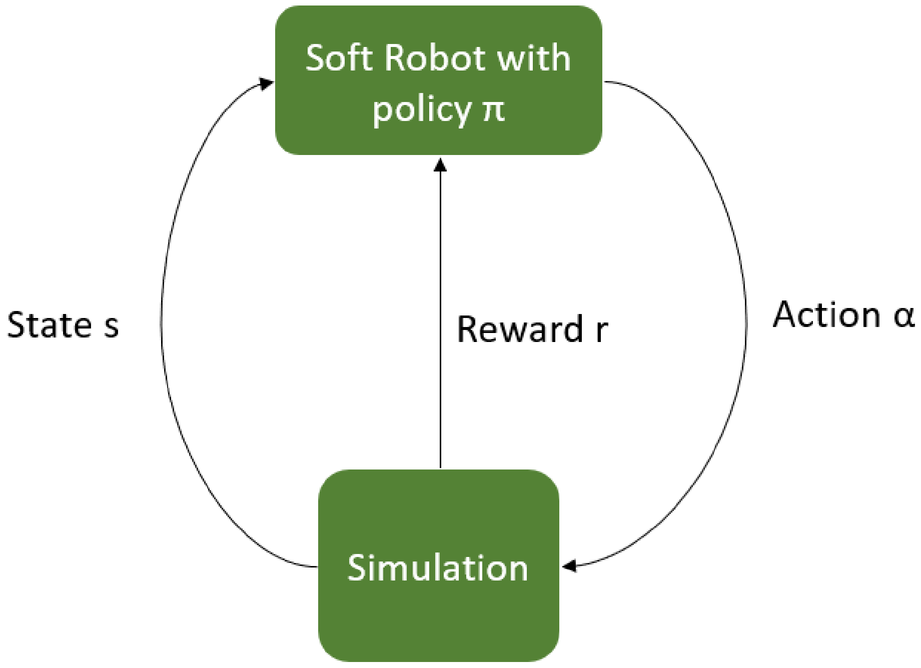


Figure 2.4: Policy Training in Model-Free RL Soft Robot Control

which are used for policy training. On the other hand, model-free RL control does not use an estimation of the robot's forward model. Therefore, in the model-free approach, the transition probability matrix is unknown and as a result the MDP is unknown. In contrast to model-based RL, the followed model-free approach utilizes only a physical simulation environment to produce the policy-training data, i.e trajectory data. In other words, a physical simulation environment is utilized to model the dynamics and kinematics of the investigated soft robot and its environment and simulate their interaction. Finally, the reward function is manually designed to reward the soft robot as it shortens the distance between its end-effector and the target. The policy-training process in the context of the followed model-free approach is visualized in Figure 2.4.

Proximal Policy Optimization

In the present thesis, Proximal Policy Optimization (PPO) is utilized in order to obtain the optimal policy (i.e. the optimal controller) for the examined soft robot RL control problem. PPO belongs to the family of Actor-Critic RL algorithms [48] which estimate both the policy (Actor) and the state value function (Critic) of the Markov Decision Process. In PPO, two neural networks are used as actor and critic function approximators thereby parameterizing the policy $\pi(\alpha|s)$ by $\boldsymbol{\theta} = [\theta_1, \theta_2, ..., \theta_N]$ and the state value function $V(s)$ by $\boldsymbol{\phi} = [\phi_1, \phi_2, ..., \phi_M]$. Hence, the deterministic policy $\pi(s) \rightarrow \alpha$ of value-based methods [48] is replaced by a stochastic policy $\pi_\theta(a|s) = P(\alpha|s, \boldsymbol{\theta})$ which constitutes a probability distribution parameterized by $\boldsymbol{\theta} = [\theta_1, \theta_2, ..., \theta_N]$. Simultaneously, the state value function approximator $V_{\boldsymbol{\phi}}(s)$ i.e Critic, estimates the expected return when being at state s. Hence, in PPO, actions are sampled from a probability distribution while the parameter vector $\boldsymbol{\theta}$ dictates the sampling probability of these actions and these actions are evaluated by the Critic. Therefore, in PPO the policy can be tuned directly as in Policy Gradient Methods, while the state-value function can be used as a baseline function to reduce the variance induced by policy gradient updates [48, 59].

Considering the above, the goal of RL control based on PPO, is the optimization of the parameter vector $\boldsymbol{\theta}$ in order to maximize the amount of reward received from the MDP. Therefore, in a finite horizon discounted MDP the parameter vector $\boldsymbol{\theta}$ that maximizes the probability of following the highest reward trajectory starting from initial state s_0, must be found. Hence, the objective function is:

$$J(\boldsymbol{\theta}) = E_{\pi_{\boldsymbol{\theta}}}[\sum_{t=0}^{T-1} \gamma^t r(s_t)] \tag{2.2}$$

where $r(s_t)$ is the immediate reward, i.e reward when taking action α in state s at time-step t and E is the symbol for expectation.Thus, the corresponding maximization problem is:

$$max_{\boldsymbol{\theta}}(J(\boldsymbol{\theta})) = max_{\boldsymbol{\theta}}(E_{\pi_{\boldsymbol{\theta}}}[\sum_{t=0}^{T-1} \gamma^t r(s_t)]) \tag{2.3}$$

In PPO, as in all policy gradient methods, the policy is differentiable and it is approximated by a neural network whose weights are essentially the parameters $\boldsymbol{\theta}$. In general, to optimize the objective function, the gradient $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$ must be utilized in order to update the parameter vector $\boldsymbol{\theta}$. For policy gradient methods the following equation stands [48, 59, 62]:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = E_{\pi_{\boldsymbol{\theta}}}[(\sum_{t=0}^{T-1} \nabla_{\boldsymbol{\theta}} log\pi_{\boldsymbol{\theta}}(a_t|s_t))(\sum_{t=0}^{T-1} \gamma^t r(s_t))] \tag{2.4}$$

By extending equation (2.4) the following expression can be obtained [62]:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = E_{\pi_{\boldsymbol{\theta}}}[\sum_{t=0}^{T-1} \nabla_{\boldsymbol{\theta}} log\pi_{\boldsymbol{\theta}}(a_t|s_t) Q(s_t, a_t)] \tag{2.5}$$

Where $Q(s_t, a_t)$ is the action-value function. Since the expected policy gradient update is independent from the state-value function approximation $V_{\boldsymbol{\phi}}(s_t)$, the latter can be used as baseline function [48, 62]. Hence, by using $V_{\boldsymbol{\phi}}(s_t)$ and exploiting the Bellman optimality equation, the following expression for the gradient can be obtained:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) \approx \sum_{t=0}^{T-1} \nabla_{\boldsymbol{\theta}} log\pi_{\boldsymbol{\theta}}(a_t|s_t)(r(s_t) + \gamma V_{\boldsymbol{\phi}}(s_{t+1}) - V_{\boldsymbol{\phi}}(s_t)) \tag{2.6}$$

The term $r(s_t) + \gamma V_{\boldsymbol{\phi}}(s_{t+1}) - V_{\boldsymbol{\phi}}(s_t)$ coincides with the Temporal Difference error and it is called the advantage function $A(s_t, \alpha_t)$ (or advantage function approximator in this case) [48]. Furthermore, PPO restricts the range within which the policy can change [63]. Therefore, the difference in expected returns between two consecutive policies ($\pi_{\boldsymbol{\theta}}$,$\pi_{\boldsymbol{\theta}+\delta\boldsymbol{\theta}}$) is expressed by the sum of the expected reward while following $\pi_{\boldsymbol{\theta}}$ and the expected advantage of the next policy $\pi_{\boldsymbol{\theta}+\delta\boldsymbol{\theta}}$. Hence, the expected reward of the new policy can be expressed as the sum of the expected reward of the previous policy and the expected advantage of the new policy [64].

$$J(\pi_{\boldsymbol{\theta}+\delta\boldsymbol{\theta}}) = J(\pi_{\boldsymbol{\theta}}) + E_{\pi_{\boldsymbol{\theta}+\delta\boldsymbol{\theta}}}[\sum_{t=0}^{T-1} \gamma^t A^{\pi_{\boldsymbol{\theta}}}(s_t, \alpha_t)] \tag{2.7}$$

A positive advantage implies that the action taken led to a positively unexpected reward. The actor objective function after adjustments for compatibility with Monte Carlo updates and after the utilization of state samples and importance sampling it takes the following form [64]:

$$J(\pi_{\boldsymbol{\theta}+\delta\boldsymbol{\theta}}) - J(\pi_{\boldsymbol{\theta}}) = E_{\pi_{\boldsymbol{\theta}}}[\frac{\pi_{\boldsymbol{\theta}+\delta\boldsymbol{\theta}}(\alpha|s)}{\pi_{\boldsymbol{\theta}}(\alpha|s)} A^{\pi_{\boldsymbol{\theta}}}(s, \alpha)] \tag{2.8}$$

Finally, PPO uses a clipping range so that advantages achieved by large parameter updates (i.e updates which exceeded the specified range) are rejected. Thus, PPO ensures that the policy after the update stays close to the previous policy. In other words, the goal is to assure that the importance sampling ratio $\rho_t(\pi_{\boldsymbol{\theta}}, \pi_{\boldsymbol{\theta}+\delta\boldsymbol{\theta}}) = \frac{\pi_{\boldsymbol{\theta}+\delta\boldsymbol{\theta}}(\alpha|s)}{\pi_{\boldsymbol{\theta}}(\alpha|s)}$ stays within a range around unity [63]. Stating the above mathematically, the actor objective function in PPO is:

$$L^{clip}_{\pi_{\boldsymbol{\theta}}} = E_{\pi_{\boldsymbol{\theta}}}[\sum_{t=0}^{T-1}[min(\rho_t(\pi_{\boldsymbol{\theta}}, \pi_{\boldsymbol{\theta}+\delta\boldsymbol{\theta}})A^{\pi_{\boldsymbol{\theta}}}_t, clip(\rho_t(\pi_{\boldsymbol{\theta}}, \pi_{\boldsymbol{\theta}+\delta\boldsymbol{\theta}}), 1-\epsilon, 1+\epsilon)A^{\pi_{\boldsymbol{\theta}}}_t)]] \tag{2.9}$$

Since there is no dependence of $(1-\epsilon)A$ and $(1+\epsilon)A$ on $\boldsymbol{\theta}$, their gradient is 0 (Figure 2.5). Consequently, samples outside the defined region are rejected thereby avoiding large updates and the parameters are updated via stochastic gradient ascent [63]. Finally, the parameters of the critic are updated by regression on mean-squared error [63]. Thus, $L_{critic} = \frac{1}{M}\sum_{i=1}^{M}(G_i - V_{\boldsymbol{\phi}}(s_i))^2$, where M are the set of current experiences (i.e. trajectory of $state, action, immediate\ reward$ obtained by following the current policy) that are used in order to update the critic parameter vector $\boldsymbol{\phi}$, G_i is the return that corresponds to experience i defined by a set of $state, action, immediate\ reward$, and $V_{\boldsymbol{\phi}}(s_i)$ the approximation of state-value function that corresponds to the state of i^{th} experience.

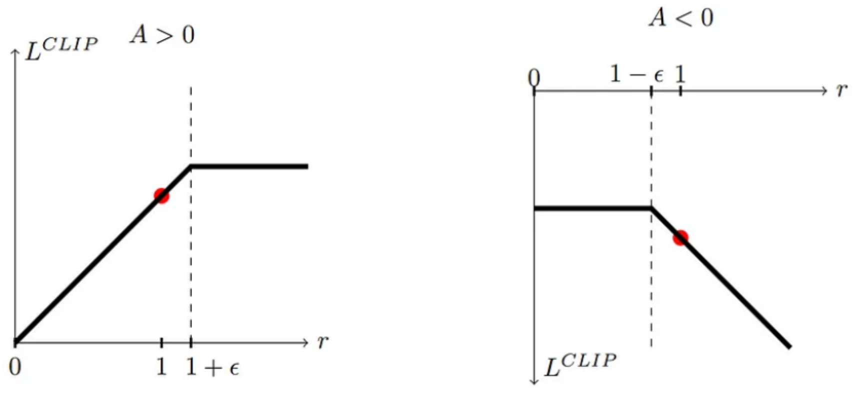


Figure 2.5: If the deviation between the updated policy and the previous one is larger than ϵ, the gradient is zero for the corresponding sample. In this graph, r is the importance sampling ratio ρ. Image from [63].

2.2 Soft Continuum Robot Modelling & Simulation

Since Model-Free RL control is the followed approach in the present research, the policy-training data must be derived from a physical simulation environment. As already stated in the previous chapter, multiple environments for modelling and simulation of soft continuum robots exist, each one based on a different theory or method. Hence, selecting one environment over another entails differences in model accuracy and complexity. Therefore, in the context of Model-Free RL control, it is essential to select a modelling and simulation environment which ensures that the special dynamics and kinematics of soft continuum robots are captured while the produced model's complexity renders its coupling with Model-Free RL algorithms possible.

2.2.1 Physical Simulation Environment Selection

The investigated physical system is conceptually similar to the beams studied in classical mechanics. In particular, the examined soft continuum robot, as the term continuum implies, is a continuous three-dimensional dynamic structure, whose length is much larger compared to the diameter. Hence, due to its geometry, it can be considered as a continuous slender beam. Therefore, it can be assumed that only 6 strains are possible for each infinitesimal material point of the robot's body i.e. 2 shear strains, 1 normal strain, 2 bending strains and 1 twisting strain. Consequently, it can be argued that modelling the examined system based on continuum mechanics is sufficient, since its 3D-dynamics can be analyzed [65]. However, due to the softness of the robot's material, strains are large and thus the deformed configuration of the examined robot differs substantially from its un-deformed configuration.

This essentially means, that in the examined case, linear elasticity does not stand and a continuum mechanics model, based on non-linear elasticity must be utilized. To capture the continuum nature of the investigated robotic device, Cosserat-Rod theory can be utilized, since Cosserat rods present the ability to bend, twist, shear and stretch thereby modelling sufficiently all the possible deformations of the examined soft robotic system. The robot's dynamics are expressed through a system of geometrically exact and decoupled non-linear equations [65]. Furthermore, it has been shown that the Cosserat rod model presents high accuracy when it is used for modelling the shape of elongated rod-like continuum robots, such as the one that is investigated in the present thesis [66–69].

Therefore, in the present thesis, the physical simulation environment is chosen based on whether or not it satisfies the following conditions:

- It utilizes a dynamic Cosserat rod model that captures all possible deformations while accounting for the non-linear elasticity of the robot's material
- Its computational cost is not prohibitive for coupling it with RL

Elastica satisfies both of the aforementioned conditions, since it is able to consistently model the geometry, dynamics and kinematics of the investigated soft continuum robots while offering a fast real-time interactive simulation which can be adapted in order to be coupled with RL control algorithms [30]. Hence, for the scope of the present research, it is chosen as the physical simulation environment which is used for the application of Model-Free RL soft robot control. Elastica has shown sufficient modelling accuracy without constituting an oversimplified approach nor a prohibitively expensive one, computationally [27]. In particular, unlike spring-damper rigid body solvers [70], Elastica is able to simulate the soft robot's elastic behavior while avoiding the complexity of finite elements methods (FEM). The feasibility of Elastica has been demonstrated in multiple engineering problems involving the modelling of complex biophysical systems such as bio-hybrid soft robots and biological systems such as human joints, snakes and wings [27, 71, 72]. In the following sections Cosserat rod theory is explained as well as how Elastica can solve the dynamics/kinematics equations of the Cosserat rod to account for the soft robot's non-linear elasticity, and simulate its behavior.

2.2.2 Cosserat-Rod Theory

Cosserat rod theory constitutes an extension of the Kirchhoff rod theory [27]. In particular, Kirchhoff rod theory is used to model one-dimensional slender rods which can only twist and bend. Cosserat rods extend Kirchhoff theory by taking into consideration the ability of a rod to stretch and shear on top of bending and twisting [27].

The Cosserat rod can be characterized as a mathematical concept [27, 65, 73]. In particular, it can be described as an infinitely thin filament which forms an one-dimensional continuum of material points (also called center-line or space-curve or material curve) $\boldsymbol{r}(s,t)$ in task space, where t represents time and s its arc-length (Figure 2.6).

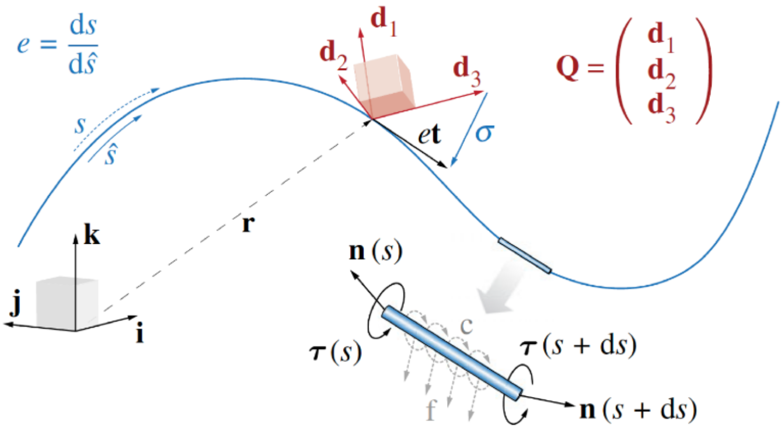


Figure 2.6: Mathematical Description of a Cosserat Rod. Image from [27].

This filament consists of cross-sections which cannot deform and whose thickness is approaching zero. Due to the absence of cross-sectional distortion as well as the elements' homogeneity, Poisson expansion as well as lateral inertia and stochastic internal damping can be omitted. Consequently, the 3D dynamics of the Cosserat rod can be described by a system of deterministic and geometrically exact non-linear equations [65]. To fully describe a Cosserat rod, apart from the definition of the right handed fixed Cartesian base (i.e. global frame defined by $\boldsymbol{i}, \boldsymbol{j}, \boldsymbol{k}$), a set of directors $\boldsymbol{d_1}, \boldsymbol{d_2}, \boldsymbol{d_3}$ (i.e a triad of orthonormal basis vectors) is defined at each material point. The directors also constitute the Cosserat rod's local frame at each material point (Figure 2.6). Thus, the location of each material point at arc-length s and the orientation of vectors $\boldsymbol{d}_1, \boldsymbol{d_2}$ at these material points, uniquely identify the Cosserat rod's position and orientation in task space. It must also be clarified that each arc-length coordinate s which corresponds to the current configuration of the Cosserat rod uniquely identifies a material point. The transformation between the local frame of an arbitrary material point and the global frame is possible using the 3×3 rotation matrix $\boldsymbol{Q}$ whose elements are the directors of this material point expressed in global frame [27]. Therefore, assuming a vector $\boldsymbol{x} = x_1\boldsymbol{i} + x_2\boldsymbol{j} + x_3\boldsymbol{k}$ expressed in global frame, the same vector can be expressed in local frame using the following equation: $\boldsymbol{x_L} = \boldsymbol{Q}\boldsymbol{x_G}$. Similarly, $\boldsymbol{x} = \boldsymbol{Q}^T\boldsymbol{x_L}$.

Since axial stretching or compression is possible in the case of the Cosserat rod, it is important to make a distinction between the arc-length coordinate s of the axially strained configuration and the rest (i.e. axial strain-free) configuration arc-length coordinate $\hat{s}$. The local stretching or compression ratio expresses the Cosserat rod's local deformation i.e how much the material point's arc length changed relative to rest reference configuration due to the presence of axial strain (Figure 2.6). Mathematically, it can be defined as:

$$e(\hat{s},t) = \frac{ds}{d\hat{s}} \tag{2.10}$$

The unit tangent vector $\boldsymbol{t}$ of a material point s expressed in current deformed configuration, is given by:

$$\frac{\partial \boldsymbol{r}(s,t)}{\partial s} = \boldsymbol{t} \tag{2.11}$$

In equation 2.11, t is time and $\boldsymbol{t}$ is the unit tangent vector corresponding to material point s. Therefore by substituting equation 2.10 to equation 2.11, the tangent vector of the same material point s expressed in the rest reference configuration $\hat{s}$ becomes:

$$\frac{\partial \boldsymbol{r}(s,t)}{\partial \hat{s}} = e(\hat{s},t) \cdot \boldsymbol{t} \tag{2.12}$$

If there is local deformation due to axial stretching/compression (i.e. $e \neq 1$) and no shear strains, the tangent vector of the material point s is parallel to the director $\boldsymbol{d_3}$. Hence, the axial strain can be expressed in global frame as:

$$\sigma = e\boldsymbol{d_3} - \boldsymbol{d_3} = (e-1)\boldsymbol{d_3} \tag{2.13}$$

However, in a scenario where shear strains are also present, the director frame is detached from the tangent vector and thus the director $\boldsymbol{d_3}$ shifts away from the tangent vector i.e. $\boldsymbol{t} \neq \boldsymbol{d_3}$. This is visualized clearly in Figure 2.6 and the strain vector expressed, in global frame, takes the following form:

$$\sigma = e\boldsymbol{t} - \boldsymbol{d_3} \tag{2.14}$$

Finally, the equations that describe the Cosserat rod's kinematics and dynamics have to be identified. In particular, in order to track changes in the rod's shape, the rate of change of the material frame $\boldsymbol{Q}(s,t)$ with respect to time is needed, since it is associated with the rod's angular velocity [65]. The relationship between the rate of change of the directors in time and the angular velocity $\boldsymbol{\omega}$ expressed in global frame is shown in the following equation.

$$\frac{\partial \boldsymbol{d_k}}{\partial t} = \boldsymbol{\omega} \times \boldsymbol{d_k} \tag{2.15}$$

where k=1,2,3. The rod's translational velocity is simply [65]:

$$\frac{\partial \boldsymbol{r}}{\partial t} = \boldsymbol{v} \tag{2.16}$$

The Cosserat Rod dynamics are defined by the conservation of linear and angular momentum at each material point's cross-section. Mathematically, they are described by Equations (2.17),(2.18) which are expressed in global frame.

$$Linear : \frac{\partial(\rho A \boldsymbol{v})}{\partial t} = \frac{\partial \boldsymbol{n}}{\partial s} + \boldsymbol{f} \tag{2.17}$$

$$Angular : \frac{\partial \boldsymbol{h}}{\partial t} = \frac{\partial \boldsymbol{\tau}}{\partial s} + \frac{\partial \boldsymbol{r}}{\partial s} \times \boldsymbol{n} + \boldsymbol{c} \tag{2.18}$$

where ρ is the density of the rod's material, A is the cross-sectional area in current state, $\boldsymbol{v}$ is linear velocity and $\boldsymbol{h}$ is the angular momentum line density. The internal force vector, caused by the presence of shear and/or axial strains at the cross-section is represented by $\boldsymbol{n}$ and described in global frame. Simultaneously, internal torques due to internal forces are described by the term $\frac{\partial \boldsymbol{r}}{\partial s} \times \boldsymbol{n}$ and $\boldsymbol{\tau}$ is the internal torque vector as a result of bending and/or twisting strains. Finally, the vectors $\boldsymbol{f}, \boldsymbol{c}$ represent the externally applied force and torque respectively. The aforementioned strains express the local deformations of the rod relative to its reference configuration and therefore they are naturally expressed in the local frame. Of course, this is also the case for the internal forces/torques which are produced because of these strains. Thus, the equations (2.17) and (2.18) need to be closed by defining a constitutive relation between the internal forces/torques and then expressed in local frame.

2.2.3 Constitutive Model

The solution to the system of equations (2.15)-(2.18) provides the dynamics and kinematics of the Cosserat rod. However, in order to close the aforementioned system, it is necessary to derive an explicit description of the internal torques and forces that act on cross-sectional level. Elastica restricts the Cosserat Rod model to a system where local strains are linearly related to local stresses [27]. By integrating the stress/couple density over the area of the cross-section, the corresponding load-strain relationships emerge. As already mentioned, the local bending/twisting strains are naturally expressed in local frame and they constitute the local curvature $\boldsymbol{k_L} = k_1 \boldsymbol{d_1} + k_2 \boldsymbol{d_2} + k_3 \boldsymbol{d_3}$ [65]. Thus, assuming that the Cosserat rod's intrinsic curvatures are zero (i.e. the rod is straight in its stress-free state), the corresponding internal torque vector for a material point is:

$$\boldsymbol{\tau_L} = \boldsymbol{B} \cdot \boldsymbol{k_L} \tag{2.19}$$

where $\boldsymbol{B}$ is the diagonal bend/twist stiffness matrix composed by the bending and twisting stiffnesses.

Similarly, assuming that the rod's intrinsic shear/axial strains are zero, the internal forces due to the local shear/axial strains $\boldsymbol{\sigma}_L = \sigma_1 \boldsymbol{d_1} + \sigma_2 \boldsymbol{d_2} + \sigma_3 \boldsymbol{d_3} = \boldsymbol{Q}(e\boldsymbol{t} - \boldsymbol{d_3})$ take the following form:

$$\boldsymbol{n_L} = \boldsymbol{S} \cdot \boldsymbol{\sigma}_L \tag{2.20}$$

where $\boldsymbol{S}$ is the diagonal shear/stretch stiffness matrix which consists of the shearing and stretching stiffnesses.

Matrices $\boldsymbol{B}, \boldsymbol{S}$ are diagonal since it is assumed that the rod's material is isotropic, i.e. material properties identical in all directions [27]. The aforementioned stiffnesses are defined by the material components (Young's Modulus E and Shear Modulus G) as well as the geometric components (Cross-sectional area A, second moment of area matrix $\boldsymbol{I} \in \mathbb{R}^{3\times 3}$ with diagonal elements (I_1, I_2, I_3)) [74]. More precisely the stiffness matrices take the following forms:

$$\boldsymbol{B} = \begin{bmatrix} EI_1 & 0 & 0 \\ 0 & EI_2 & 0 \\ 0 & 0 & EI_3 \end{bmatrix} \tag{2.21}$$

$$\boldsymbol{S} = \begin{bmatrix} \alpha_c GA & 0 & 0 \\ 0 & \alpha_c GA & 0 \\ 0 & 0 & EA \end{bmatrix} \tag{2.22}$$

where α_c is a constant for circular cross-sections. As it can be seen from equations (2.21) and (2.22), Elastica's constitutive model renders the stiffness matrix S directly dependent on the cross-sectional area A and the stiffness matrix B dependent the second moment of area I. Since stretching/compression can occur in the Cosserat rod, the radius of the cross-section changes upon deformation. Thus, A and I are not constant and the load-strain relation is non-linear. Furthermore, Elastica assumes that the Cosserat rod's material is incompressible i.e. $v = 0.5$ and that the material points' cross-sections always retain their circular shape. Therefore, the relationship between the cross-sectional area in rest configuration ($\hat{A}$) and the cross-sectional area in the deformed configuration A can be defined as:

$$A = \frac{\hat{A}}{e} \tag{2.23}$$

Similarly, the relationship between the second moment of area in rest configuration ($\hat{I}$) and the second moment of area in the deformed configuration I can be defined as:

$$I = \frac{\hat{I}}{e^2} \tag{2.24}$$

Naturally, the rest of the geometric components have to be scaled by the local stretching ratio e when axial extension/compression occurs. More specifically:

$$ds = ed\hat{s},\ \boldsymbol{S} = \frac{\hat{\boldsymbol{S}}}{e},\ \boldsymbol{B} = \frac{\hat{\boldsymbol{B}}}{e^2},\ \boldsymbol{k_L} = \frac{\hat{\boldsymbol{k_L}}}{e} \tag{2.25}$$

Based on the above, Elastica's assumption of linear elasticity rendered the identification of the internal torques and forces possible and thus the system of equations (2.15)-(2.18) can be closed. Simultaneously, even though the stress-strain relationship is assumed linear, the load-strain relationship is non-linear and the equations of the dynamics can be expressed with respect to the rest reference configuration, by re-scaling the corresponding geometric components with the local stretching ratio e (thanks to the incompressibility assumption). Hence, Elastica's constitutive model is based on a non-linear load-strain relationship which is dependent on local stretching ratio. According to [27], Elastica's model is able to approximate the Neo-Hookean and Mooney–Rivlin models when axial extension/compression is below 30%. Thus, the non-linear elasticity of soft continuum robots can be captured by Elastica if the modelled strains are within the acceptable range and assumption of incompressibility is not violated. Consequently, by substituting the constitutive relation, and by expressing local strains and stresses in local frame the general Cosserat rod dynamics equations (2.17) and (2.18) can be transformed and expressed with respect to the rest reference configuration $\hat{s}$ as follows:

$$\rho\hat{A}\frac{\partial(\boldsymbol{v})}{\partial t} = \frac{\partial(\frac{\boldsymbol{Q}^T\hat{\boldsymbol{S}}\boldsymbol{\sigma_L}}{e})}{\partial\hat{s}} + e\boldsymbol{f} \tag{2.26}$$

$$\frac{\rho\hat{\boldsymbol{I}}}{e}\frac{\partial(\boldsymbol{\omega}_L)}{\partial t} = \frac{\partial(\frac{\hat{\boldsymbol{B}}}{e^3}\hat{\boldsymbol{k}_L})}{\partial\hat{s}} + \frac{\hat{\boldsymbol{k}_L}\times\hat{\boldsymbol{B}}\hat{\boldsymbol{k}}_L}{e^3} + (\boldsymbol{Qt}\times\hat{\boldsymbol{S}}\boldsymbol{\sigma}_L) + (\frac{\rho\hat{\boldsymbol{I}}\boldsymbol{\omega}_L}{e})\times\boldsymbol{\omega}_L + (\frac{\rho\hat{\boldsymbol{I}}\boldsymbol{\omega}_L}{e^2})\frac{\partial e}{\partial t} + e\boldsymbol{c}_L \tag{2.27}$$

Finally, the kinematics equations have the following form:

$$\frac{\partial\boldsymbol{d_k}}{\partial t} = \boldsymbol{Q^T}\boldsymbol{\omega}_L\times\boldsymbol{d_k} \tag{2.28}$$

where k=1,2,3.

$$\frac{\partial\boldsymbol{r}}{\partial t} = \boldsymbol{v} \tag{2.29}$$

The equations (2.26)-(2.29) are non-linear partial differential equations whose analytical solution is not always available. Therefore Elastica solves these equations using a numerical method. Therefore, the obtained dynamic Cosserat rod model has to be discretized both spatially and in time.

Spatial Discretization

Elastica follows the Discrete Elastic Rod (DER) approach [75] in order to spatially discretize the Cosserat rod. In DER approach, the configuration of the rod at a specific time instant t_0 is defined by the Cartesian position of each material point (i.e. cross-section) $\boldsymbol{r}(s, t_0)$ as well as by the orientation of the corresponding material frames $\boldsymbol{Q}(s, t_0)$. Thus, the poses of the rod's cross-sections can be parameterized by the space-curve $\boldsymbol{r}(s)$ and the angle $\Theta(s)$ between the Bishop frame and the material frame $\boldsymbol{Q}(s)$, since the Bishop frame expresses the non-twisted material frame of the cross-section [27, 31, 75]. DER exploits the aforementioned parameterization and the fact that the Bishop frame's evolution along the rod can be defined by parallel transport, in order to discretize the rod using vertices r_i and edges e (Figure 2.7) [75].

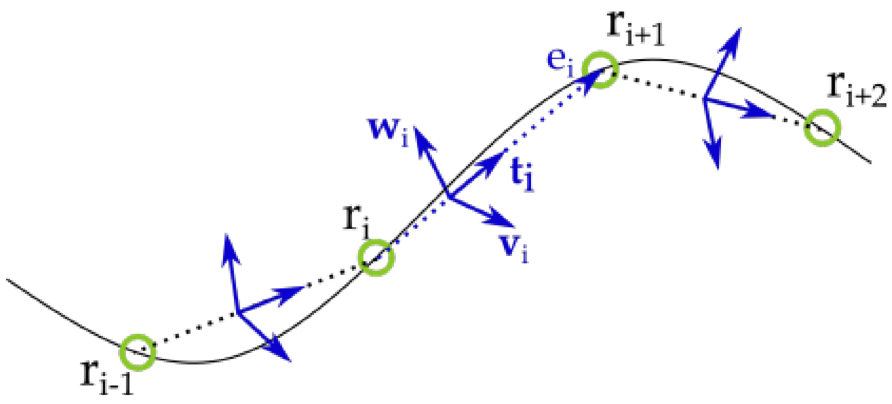


Figure 2.7: Rod spatial discretization based on DER with Bishop frames attached on each edge. Image from [31]

Since Elastica follows the DER approach in order to consistently discretize the Cosserat rod, the deformation of the Cosserat rod in 3D space is captured through the dynamics of a discrete set of vertices $\boldsymbol{r}_i(t) \in R^3, i \in [1, n+1]$ and a discrete set of material frames $\boldsymbol{Q}_i(t) \in R^{3\times3}, i \in [1, n]$ as shown in Figure 2.8.

Thus, the concept is basically similar to the continuous Cosserat rod but instead of infinitesimal material points, cylindrical elements with finite length (edges) formed by sequential vertices are defined. In particular, for each vertex, its velocity $\boldsymbol{v_i}$, the externally applied force $\boldsymbol{f}_i$ on it and its mass m_i are defined. Two consecutive vertices form an edge of length l_i. This edge is basically a cylindrical element of current length $\boldsymbol{l}_i$ which is uniquely identified by its material frame $\boldsymbol{Q}_i(t)$. Thus, each edge is defined by its cross-sectional area $\hat{A}_i$, its mass second moment of inertia $\hat{\boldsymbol{J}}_i$, its angular velocity $\boldsymbol{\omega_L}$, its Bending/Twisting Stiffness $\hat{\boldsymbol{B}}_i$, its Shearing/Stretching stiffness and $\hat{\boldsymbol{S}}_i$ at rest edge length $\hat{\boldsymbol{l}}_i$. The unit tangent vector of edge $\boldsymbol{l}_i$ is simply:

$$\boldsymbol{t}_i = \frac{\boldsymbol{l}_i}{l_i} \tag{2.30}$$

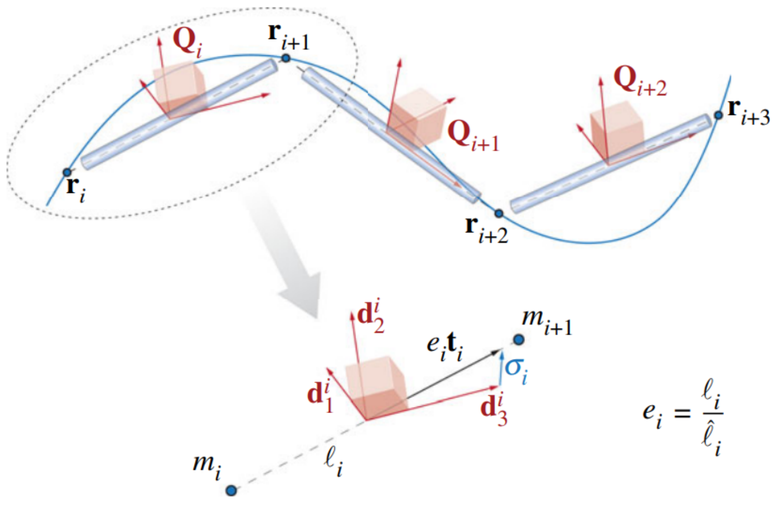


Figure 2.8: Spatial Discretization of Cosserat Rod in Elastica [27]

Hence, the stretching ratio e can be expressed as:

$$e = \frac{l_i}{\hat{l}_i} \tag{2.31}$$

Finally, discrete shear/axial strain vector that corresponds to edge $\boldsymbol{l}_i$ can be mathematically expressed as:

$$\boldsymbol{\sigma_L}^i = \boldsymbol{Q}_i(e\boldsymbol{t}_i - \boldsymbol{d}_3^i) \tag{2.32}$$

In the continuum approach (i.e. before spatial discretization), curvature $\boldsymbol{k_L}$ is considered at each material point along the Cosserat rod. Nevertheless, after DER spatial discetization, the curvature is expressed naturally in an integrated form over a domain D of the rod and not point-wise [75]. The division of the integrated quantity by the length of domain D, produces the point-wise average [75]. Therefore, in the discretized case, the interior vertex $\boldsymbol{r}_i$ (where i=1,..,n-1) defines the domain D which is essentially a Voronoi region D_i with length $\frac{l_i+l_{i+1}}{2}$ and thus curvature $\boldsymbol{k_L}^i$ is defined at each interior vertex. In order to be able to re-scale geometrical components at the defined domain length to model the desired non-linearity, Elastica defines the dilatation ratio for the domain D as:

$$\epsilon_i = \frac{D_i}{\hat{D}_i} \tag{2.33}$$

where $\hat{D}_i$ is the Voronoi domain length at rest stretch-free state.

Since curvature can be described as rotation per unit length about its axis, $D_i\boldsymbol{k_L}^i$ represents the rotation of the material frame Q_{i+1} with respect to material frame Q_i. Hence, the relative rotation between two consecutive edges can also be expressed.

The discrete curvature in Elastica is given by the following equation:

$$\boldsymbol{k_L}^i = \frac{log(\boldsymbol{Q}_{i+1}\boldsymbol{Q}_i^T)}{D_i} \tag{2.34}$$

Based on the above representation, Elastica defines the rest of necessary discrete quantities (e.g. discrete bend/twist stiffness matrix), and through spatial integration, the spatially discretized kinematics and dynamics equations are derived. In [27], the analytical derivation of the spatially discretized cosserat rod dynamics and kinematics equations, is presented.

Time Discretization

In order to numerically integrate the spatially discretized system of equations, the second order position Verlet Integration scheme is utilized [76]. Thus a full iteration of the algorithm from time-step t to time-step $t + \delta t$ consists of three stages [27]. Firstly, the position of vertices and edges' material frames is updated for half a step. Mathematically it looks like this [27]:

$$\boldsymbol{r}_i(t + \frac{\delta t}{2}) = \boldsymbol{r}_i(t) + \frac{\delta t}{2}\boldsymbol{v}_i(t) \;\; i = 1, .., n+1 \tag{2.35}$$

$$\boldsymbol{Q}_i(t + \frac{\delta t}{2}) = e^{\frac{\boldsymbol{\omega_L}^i(t)\delta t}{2}}\boldsymbol{Q}_i(t) \;\; i = 1, .., n \tag{2.36}$$

Subsequently, the local accelerations are calculated.

$$\boldsymbol{v}_i(t + \delta t) = \boldsymbol{v}_i(t) + \delta t\frac{d\boldsymbol{v}_i}{dt}(t + \frac{\delta t}{2}) \;\; i = 1, .., n+1 \tag{2.37}$$

$$\boldsymbol{\omega_L}^i(t + \delta t) = \boldsymbol{\omega_L}^i(t) + \delta t\frac{d\boldsymbol{\omega_L}^i}{dt}(t + \frac{\delta t}{2}) \;\; i = 1, .., n \tag{2.38}$$

Finally the positions of the full step are obtained through the following expressions:

$$\boldsymbol{r}_i(t + \delta t) = \boldsymbol{r}_i(t + \frac{\delta t}{2}) + \frac{\delta t}{2}\boldsymbol{v}_i(t + \frac{\delta t}{2}) \;\; i = 1, .., n+1 \tag{2.39}$$

$$\boldsymbol{Q}_i(t + \frac{\delta t}{2}) = e^{(\frac{[\boldsymbol{\omega_L}^i(t+\frac{\delta t}{2})]\delta t}{2})}\boldsymbol{Q}_i(t + \frac{\delta t}{2}) \;\; i = 1, .., n \tag{2.40}$$

Hence, during an iteration, the calculation of $\frac{d\boldsymbol{v}_i}{dt}(t+\frac{\delta t}{2})$ and $\frac{d\boldsymbol{\omega_L}^i}{dt}(t+\frac{\delta t}{2})$ is needed only once. This entails significant computational cost reduction since the evaluation of the right-hand side of the dynamics equations (discretized forms of equations 2.26 and 2.27) is minimized. Furthermore, according to [27], a Courant–Friedrichs–Lewy (CFL) condition has not been derived for the Elastica simulator. Thus, the maximum time-step that guarantees numerical stability is not specified. Instead of a CFL, an empirical relationship for the time-step is presented in [27]:

$$\delta t = a \cdot \frac{L}{n} \tag{2.41}$$

where L is the rod's length, n the number of spatial discretization elements (i.e. edges) and $a \approx 0.01 s/m$. The aforementioned empirical relationship entails that the computational cost per time-step is inversely proportional to the chosen number of edges. Furthermore, as it can be observed from the equations (2.36) and (2.40), the exponential term used for expressing material frame rotation in time, can cause stability issues. Finally, the relationship between the duration of the simulation (i.e specified number of Verlet iterations) and the real time needed to produce the simulation varies from linear to quadratical [27].

2.2.4 The physical system's pneumatic actuation in Elastica's discretized dynamic Cosserat rod model

It is concluded that Elastica's discretized dynamic Cosserat rod model is able to capture all six deformations at each cross-section thereby accounting for the elongation, bending, twisting and shearing of the examined soft robot. Consequently, the chosen model is deemed sufficient for modelling the soft robot's body. However, it is important to examine whether the soft robot's actuation system can be modelled sufficiently in Elastica as well, since the actions chosen by the policy should produce realistic deformations to the Elastica model, i.e. in accordance to the actual system. The investigated physical system utilizes three soft actuators which are known as the reverse Pneumatic Artificial Muscles (rPAM) [77]. Each of these actuators constitutes a cylindrical pressure chamber which is made from an elastomer (Figure 2.9). Additionally, each chamber is helically re-inforced by two woven fibres (Figure 2.9). The first fibre is woven clockwise while the second one is woven counter-clockwise, and the robot cannot twist. The helical reinforcement renders the translation of positive pressure inside the artificial muscle i.e. inflation, into elongating force acting along the longitudinal axis of chamber, due to the fact that the fibres are inextensible and circumferential stresses are countered [77]. Therefore, rPAM is essentially similar to the McKibben actuator [78] but it operates in reverse [77]. In particular, inflation of the reverse pneumatic artificial muscle, results in its radial expansion and due to the use of a helical reinforcement with an initial (winding) angle of nearly $90\,\mathrm{deg}$, the rPAM is longitudinally extended [78]. Thus, the inflation of an rPAM entails the axial stretching of its structure and its deflation makes it return back to its initial length. In order to achieve bending in all directions in 3D space, the investigated physical system uses three rPAMs which are radially distributed thereby creating a section (Figure 2.10). If internal pressure is changed in the same way in all chambers simultaneously, the section elongates or contracts (back to natural length). On the other hand, if the applied pressures differ in magnitude, the section bends. Naturally, the inflation/deflation of each chamber determines the bending plane.

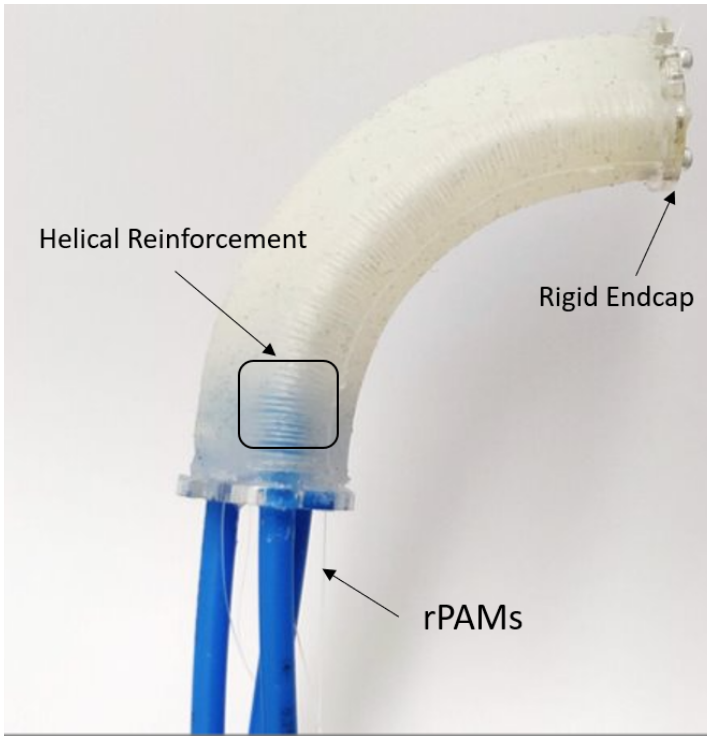


Figure 2.9: The investigated pneumatic-driven soft continuum robot (physical system). The three rPAMs are the blue pneumatic chambers which are helically reinforced. The produced forces act on a rigid end-cap.

Based on the equations (2.26) and (2.27), Elastica is able to simulate the actuation forces produced by the rPMAs as external forces and torques. In particular, the forces can be modelled as external forces acting on the last vertex $\boldsymbol{r_{last}}$ of the rod i.e. robot's free endpoint in Elastica, and along the final edge's tangent vector $\boldsymbol{t}_{last}$, due to the presence of a rigid end-cap at the robot's free endpoint (Figure 2.9). Similarly, the produced torques due to pressure difference among the chambers, can be modelled in Elastica as external torques acting on the rod's last edge of the rod and expressed in its material frame Q_{last}. In chapter 3, the modelling of the actuation is described in detail.

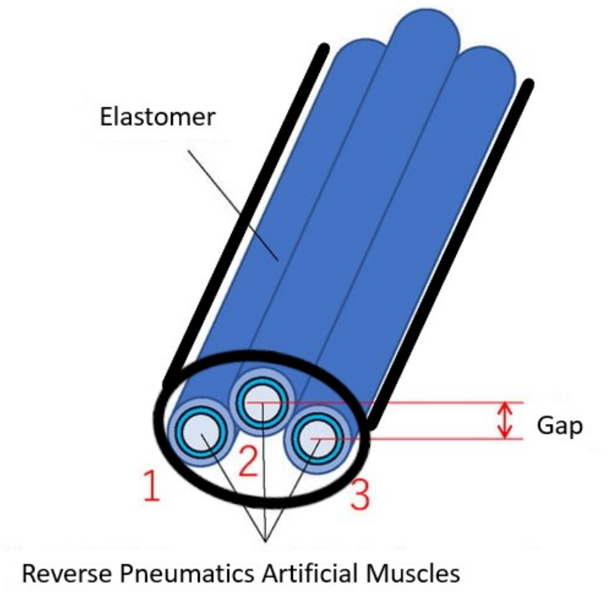


Figure 2.10: Cross-sectional view of the investigated soft robot. The robot's radially distributed rPAMs are the blue pneumatic chambers which are made from elastomer and constitute the robot's multi-backbone core.

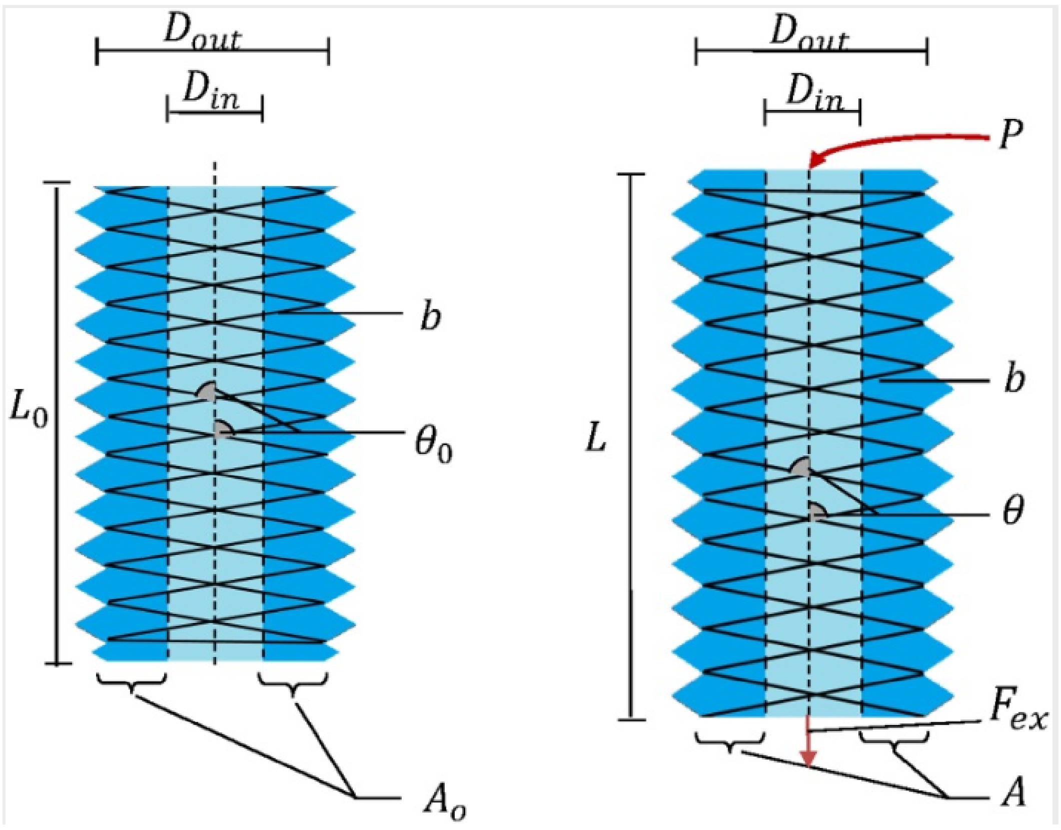


Figure 2.11: Schematic representation of the rPAM actuator utilized by the examined soft robot. Image from [77]. The left scheme depicts the actuator and its geometrical parameters before pressure application while the right one is after pressure application. The total length of the thread is represented by b, while L_0, L are the actuator's initial length and increased length due to deformation respectively and θ_0 is the initial (winding) angle and θ is the one after deformation. Finally, A_0, A are the cross-section areas of the actuator wall before and after deformation correspondingly. Upon inflation of the chamber with pressure P, the utilized helical reinforcement makes the rPAM elongate while a force F_{ext} is produced along the chamber's direction.

Chapter 3

Implementation

3.1 Modelling & Simulation of the Physical system in Elastica

In the context of the present thesis, the application of Model-Free RL control depends upon the utilization of a suitable physical simulation environment which models the soft robot, its environment and their interactions in order to produce the policy-training data. The Elastica simulator is able to sufficiently capture the soft robot's possible deformations, since its discretized dynamic Cosserat rod model accounts for all six deformation modes associated with the 6 DoF at each cross section along the rod. Before coupling the Elastica simulator with a Model-Free RL algorithm to train a policy, it is essential to produce a stable model of the soft robot and its environment and simulate its behavior upon realistic actuation. Deriving a stable discretized dynamic Cosserat model that faithfully represents the examined soft continuum robot in Elastica is not trivial. Firstly, the physical parameters and geometric characteristics of the system must be defined and simultaneously it must be ensured that they comply with the assumptions of Elastica's modelling. Secondly, the spatial and time discretization parameters have to be chosen in such a way so that numerical stability is ensured, robot's flexibility is not compromised and the final model's complexity is not too high that it becomes prohibitive to couple it with RL. Finally, the actuation system of the physical system has to be modelled in Elastica. Therefore, in this section the procedure that was followed for the creation of a suitable simulation environment in Elastica, which can later be coupled with Model-Free RL, is described.

The investigation of the soft robot's physical parameters and geometric characteristics has to be prioritized to ensure that Elastica is able to reproduce the behavior of the actual system faithfully enough. The body of the examined pneumatic-driven soft robot is essentially a cylindrical rod (as shown in Figure 2.9) which is in line with the Elastica's assumption of Cosserat rods consisting of cylindrical elements and having circular cross-sections. Furthermore, the physical system has the following geometric characteristics: Length $L = 0.1\ m$ and Radius $R = 0.009\ m$. This follows Elastica's consideration of Cosserat rods being cylindrical slender structures with length much larger than cross-sectional radius. The investigated soft robot is built from the elastomer Eco-Flex 00-10. Hence, due to the assumptions that the Elastica software is based upon, the soft robot's physical parameters which are utilized in order to model it, are the following [79–81]:

- Density $D = 1040.5979\ kg/m^3$
- Young's Modulus $E = 0.05\ MPa$

Furthermore, the Eco-Flex 00-10 material has a Poisson ratio of approximately 0.5 ,i.e $v \approx 0.5$ [80], which satisfies Elastica's assumption of incompressibility. Thus, the shear modulus can be calculated, $G \approx 0.0166\ MPa$. According to [27], the rods with high Young's modulus make the Elastica's governing equations stiff, and as a result, extremely small steps have to be taken when solving them numerically. The low Young's Modulus of the examined case implies that this problem can be avoided. The process of modelling and simulating the investigated soft robot and its workspace in Elastica consists of 6 steps [82]. The first step is to set up the system simulator combining all the modules required for the needs of the simulation. The necessary modules for the examined case are:

- ”Constraints” in order to be able to use boundary conditions,
- ”Forcing” for actuation purposes,
- ”CallBacks” in order to be able to collect data during simulation and
- ”Damping” to introduce damping into the simulation.

The second step is to model the investigated soft robot as a straight, spatially discretized, Cosserat rod (i.e initial state's curvature and shear/axial strains are zero). For the creation of the rod, the soft robot's physical parameters have to be specified as well as the number of cylindrical elements (edges) that should constitute the Cosserat rod. Subsequently, the appropriate boundary conditions of the rod as well as its forcing and damping have to be defined. Then, the data collection functions have to be designed. The next step is to define the dicretization time-step in order to numerically solve the system of partial differential equations. Finally, post-processing functions are designed in order to plot the data obtained from the callback functions and create animation videos of the rod in 3D space. The described process can be visualized in Figure 3.1.

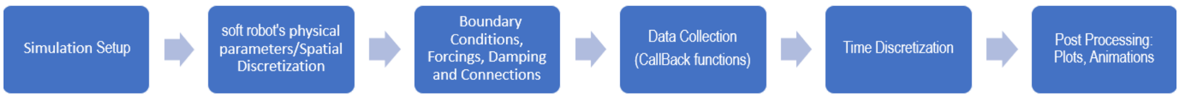

Simulation Setup
soft robot's physical parameters/Spatial Discretization
Boundary Conditions, Forcings, Damping and Connections
Data Collection (CallBack functions)
Time Discretization
Post Processing: Plots, Animations

Figure 3.1: Process of creating an Elastica Simulation

Having defined the necessary simulation modules as well as the soft robot's physical parameters, the next step is to define the number of cylindrical elements of the straight Cosserat rod. The higher the number of cylindrical elements, the higher the flexibility of the simulated rod. Of course it is not possible to accurately identify the physical system's total number degrees of freedom. Therefore a specific number of elements is not suggested by the physical device. The goal is to choose a number of spatial discetization elements that ensures high flexibility, i.e. much higher DoF than typical rigid link manipulators while not raising the time needed to solve the kinematics and dynamics equations to prohibitive levels (e.g. quadratic relationship between physical simulation time and real time needed solve the PDEs [27]). Even though the relationship between the computational cost per time-step and the number of elements is linear [27], the Elastica's empirical relationship [82] $dt = a \cdot dx$ (where a is a configurable constant parameter, dt is the numerical integration time-step and dx is the edge length after spatial discetization), implies that a larger number of cylindrical elements entail smaller time-steps and therefore increased computational cost. Furthermore, large a could produce exponentially large rotations of the material frame in time as equations (2.36) and (2.40) imply and therefore lead to inaccuracy or even instability depending on the chosen number of cylindrical elements. A small number of cylindrical elements allows larger time-steps without causing numerical stability issues, however the modelling accuracy is substantially impaired. Of course the exact impact of each choice is difficult to predict, however since the parameter a can be configured and the relationship between the computational cost per time-step and the number of elements is linear, using a larger number of cylindrical elements is possible without reaching prohibitive computational costs. Elastica's suggestion is to use 30-50 spatial discretization elements for one rod without specifying the rod's length [82]. Nevertheless, the length of the rod has to be taken into consideration as well. In the examined case the length of the Cosserat rod is 10cm therefore the use of 30 spatial discretization elements is deemed sufficient, since 90 DoF are ensured and therefore high flexibility is guaranteed. In the initial state, the rest edge length of each one of the 30 edges is $dx = \frac{0.1\ m}{30} = 0.0033\ m$. Following the procedure shown in Figure 3.1, the Cosserat rod's boundary conditions must be defined next as well as the external forces that are exerted on it and the type of damping that acts on it. One of the physical system's ends is fixed while the other one is free, similar to a Cantilever beam. Therefore in the context of Elastica simulation, assuming a (x, y, z) global frame, it chosen that one end of the rod is fixed at $(0, 0, 0)$ and the rod's free end is located at $(0, 0, 0.1)$. Thus, the rod's normal vector is $(0, 1, 0)$ since the rod is normal to X-Y plane. The rod's direction and constraint can be visualized in Figure 3.2.

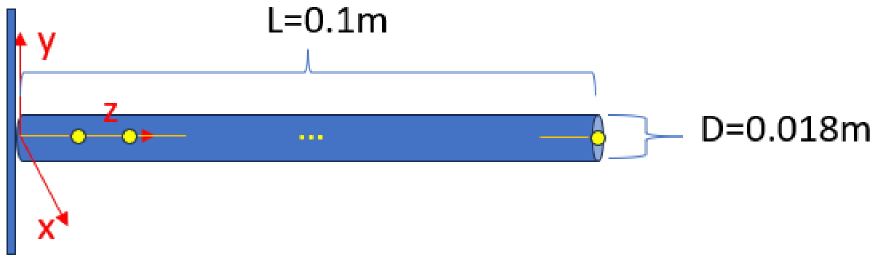


Figure 3.2: Schematic representation of the soft robot as modelled in Elastica. The yellow edges and nodes represent the model's spatial elements and vertices.

The physical system's actuation consists of multiple pneumatic chambers (actuators) which constitute the multi-backbone core of the robot. When these chambers are inflated forces which act along the soft robot's free endpoint tangent vector, are produced. These forces make the robot elongate along the free endpoint's tangent vector and when there is difference in the magnitude of the aforementioned actuation forces, bending of the robot's body in 3D space also occurs. It is important to mention that the actuation system is not able to twist the soft robot. In Figure 3.3 an example of how the physical system's actuation produces a bending motion is presented.

The soft robot body is modelled in Elastica as a single, straight, and spatially discretized Cosserat rod with the same physical parameters, characteristics and constraints. Hence, the soft robot is modelled in Elastica as a single backbone in order to avoid an implementation that includes a complex and computationally expensive system of Cosserat rods. However, Elastica does not contain a force implementation that matches the physical system's actuation system. Consequently, a custom "Forcing" class has to be designed within Elastica in order to model the physical system's actuation. In particular, it is specified that the actuation forces are external forces which act on the free endpoint's vertex (31st vertex). Furthermore, four force sources i.e. actuators are modelled, the locations of which are circumferential of the endpoint's center of mass at a distance $d_f = \frac{R(t)}{2}$. These actuators play the role of the physical system's pneumatic chambers and produce four force vectors $(\boldsymbol{F_1}, \boldsymbol{F_2}, \boldsymbol{F_3}, \boldsymbol{F_4})$,with non-negative magnitudes, which have the direction of the

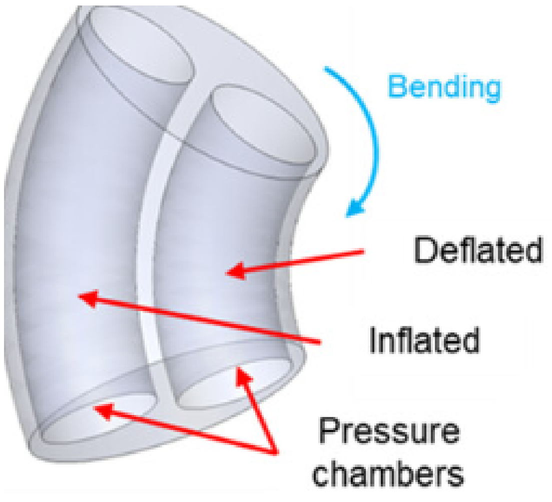


Figure 3.3: Bending due to the inflation of a chamber and the deflation of the other. Image from [83].

endpoint's tangent vector every time. The use of four actuators ensures bending in all directions in the 3D space along with elongation, thereby faithfully emulating the physical system's actuation system (Figure 3.4). An example is presented in Figure 3.5 for intuitiveness. Nevertheless, at the Elastica level, all forces act in reality on the last vertex, therefore there is no torque produced in a physical sense. Hence, the application of external torques as products of the four force vectors, has to be mathematically stated.

The modelled actuation forces can be expressed in global frame as $\boldsymbol{F} = F\frac{\partial \boldsymbol{r}}{\partial s} = F\boldsymbol{t}$ and this is sufficient for simulating elongation. Nonetheless, in order to simulate bending in 3D space, the externally applied torques need to be expressed in local frame. Therefore, the directors of the material frame corresponding to the last edge have to be expressed in local frame. Hence, using the transformation matrix $\boldsymbol{Q}$, the directors in local frame can be obtained through the equation $\boldsymbol{d}_L = \boldsymbol{Qd}$. The Force vector can then be expressed in local frame as: $\boldsymbol{F} = F\boldsymbol{Qt} = (F_1 + F_2 + F_3 + F_4)Q\boldsymbol{t}$. Hence $\boldsymbol{F_i} = F_i Q\boldsymbol{t}$ where i=1,2,3,4. Since bending in 3D space is possible but twisting is not, torques about directors $\boldsymbol{d_1}$ and $\boldsymbol{d_2}$. The equations that describe the modelled applied external torques are the following (Figure 3.4):

$$\boldsymbol{\tau_1} = [(0.5R(t)\boldsymbol{d_2}) \times \boldsymbol{F_1}] + [(-0.5R(t)\boldsymbol{d_2}) \times \boldsymbol{F_2}] \tag{3.1}$$

$$\boldsymbol{\tau_2} = [(0.5R(t)\boldsymbol{d_1}) \times \boldsymbol{F_4}] + [(-0.5R(t)\boldsymbol{d_1}) \times \boldsymbol{F_3}] \tag{3.2}$$

Where R(t) is the radius of the Cosserat rod at time-step t. The total applied torque is:

$$\boldsymbol{\tau_{total}} = \boldsymbol{\tau_1} + \boldsymbol{\tau_2} \tag{3.3}$$

Gravity: In Elastica, gravitational forces can also be modelled. More specifically

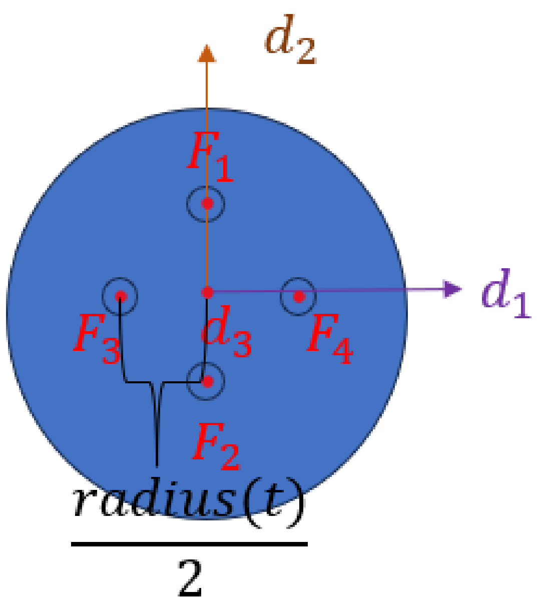


Figure 3.4: Visualization of the Modelled Actuation System in Elastica (Cross-Sectional View)

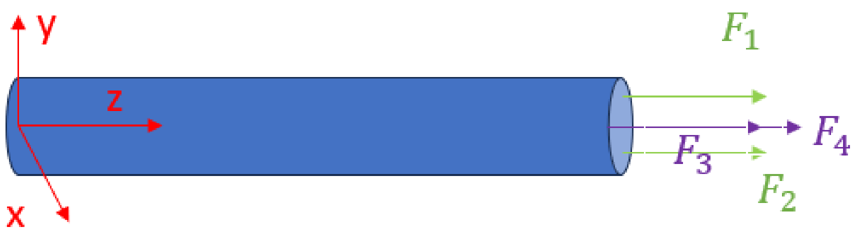


Figure 3.5: Modelled actuation in Elastica. The cosserat rod is in its initial state. If the four actuators produce forces with magnitudes $F_1 = F_2$ and $F_3 < F_4$, the rod will elongate along the endpoint's tangent vector and perform bending in the X-Z plane.

due to Elastica's spatial discretization, gravity is modelled as a uniformly distributed force. Nevertheless, since the actuation forces act only on the endpoint, the modelled actuation system is not able to perform gravity compensation effectively and therefore gravitational forces are omitted. This is also the case for the physical system as well, because the actuation system of the examined soft continuum robot is not an internal continuum one.

Damping: In order to model damping, an analytical linear damper is introduced. The way the velocities of the vertices and the edges are damped, is described by the following equations:

$$\vec{v}(n+1) = \vec{v}(n)e^{-cdt} \tag{3.4}$$

$$\vec{\omega}(n+1) = \vec{\omega}(n)e^{(\frac{-cmdt}{J})} \tag{3.5}$$

Where n is the current discrete time step, c is the damping coefficient, m is the mass of the vertex, dt is the selected simulation time-step, and J is the second moment of inertia of the edge. The analytical linear damper class treats the damping term analytically and not numerically. Consequently, configuring the damping coefficient does not affect time-step size. Hence, the damping coefficient can be set high in order to over-damp the rod and ensure that potential instabilities occur due to the selected simulation time-step dt and not from insufficient damping .

CallBack Functions & Post-Processing: Finally, the callback functions collect the position and the velocity of the Cosserat rod, i.e. modelled soft robot, throughout the simulation as well as the external torques and forces applied on its last edge and vertex respectively, due to actuation. Data are collected every 100 time-steps. Post-processing functions are designed plot the collected data in order to analyze the system's behavior and create 3D animations of the rod for intuitiveness.

Choosing simulation time-step dt: The choice of numerical integration time-step dt is the most critical aspect of creating a numerically stable simulation in Elastica. Additionally, the time-step dt determines the computational cost of the simulation. In order to find a suitable dt, the damping coefficient is set to the extremely high value $1000\ s^{-1}$ so that numerical instability caused by unsuitable time-step can be differentiated by the instability caused by the excitation of an insufficiently damped frequency. The proposed heuristic of Elastica, $dt = a \cdot dx$ is also used. Since $dx = 0.0033\ m$ finding a suitable dt is essentially tuning the parameter a. Elastica suggests the use of $a = 0.01$ [82]. Two things have to be considered nonetheless. On one hand, this relationship is based on the observations of Elastica's creators and it is not supported by a CFL condition [27]. On the other hand, apart from stability, the computational cost is important since it could become prohibitive for RL application. Thus, tests with different values of the parameter a are run in order to obtain a time-step that is appropriate for the examined problem (Appendix A.1). It is concluded that $a = 0.05$ constitutes a reasonable choice. In particular, by applying

$a = 0.05$, time-step becomes $dt = 0.0001667\ s$. By simulating the rod's behavior for 10 seconds while having only one actuator producing a constant force of 0.1N (Step Input, Figure 3.7), the results shown in Figure 3.6 are produced:

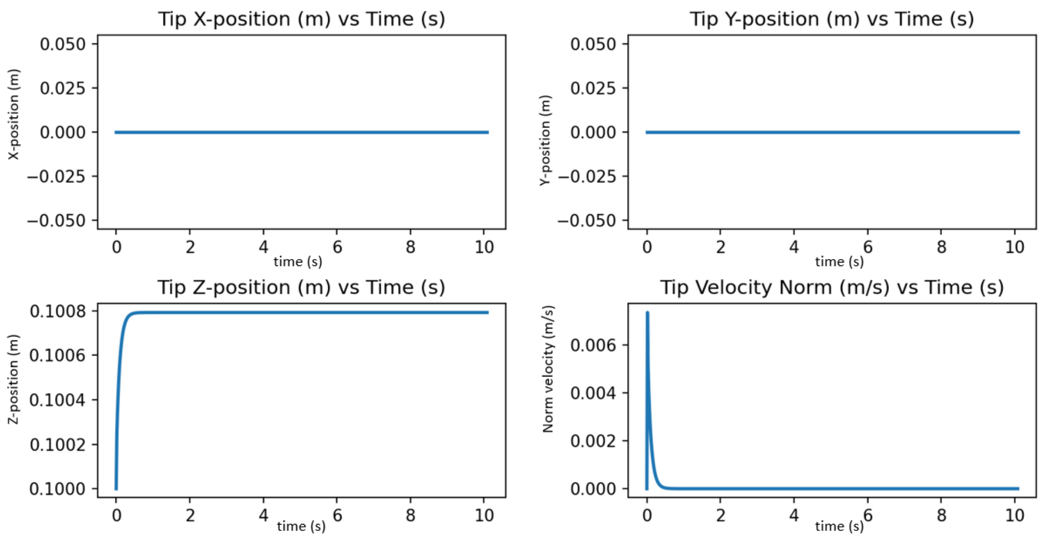


Figure 3.6: Model's end-effector position and velocity norm when a step force is given as input during time-step tuning test (Figure 3.7)

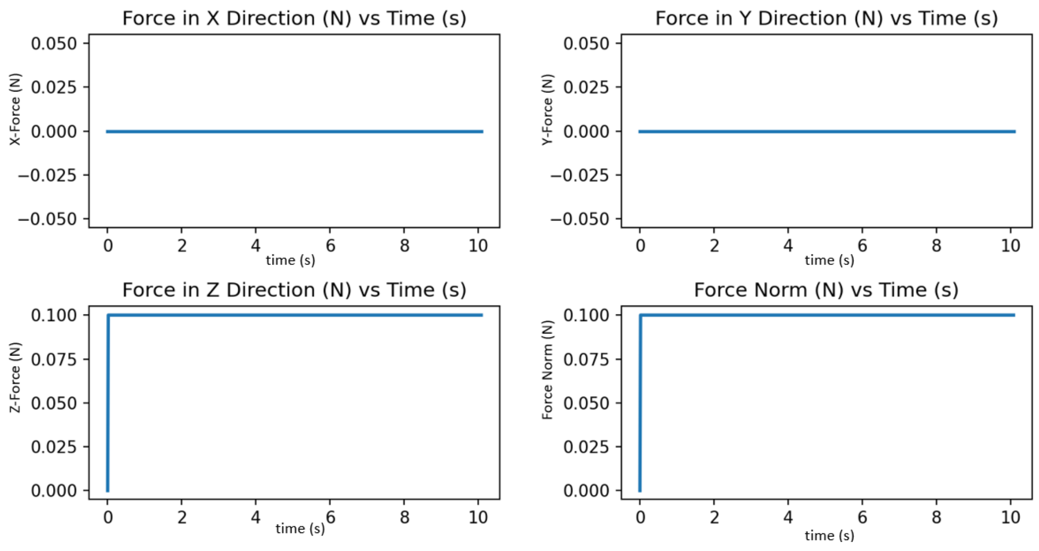


Figure 3.7: Force input during time-step tuning test

The 10 second Elastica simulation lasted 10 real seconds while being numerically stable. The choice of $a = 0.05$ strikes a balance between numerical stability and computational complexity. Thus $dt = 0.0001667\ s$ is deemed acceptable for the present case.

Based on the analysis above, the crucial parameters of spatial and time discretization are specified. However, in order to further check the competence of the obtained model, its behavior is examined in the presence of a small normal stress i.e. stress which corresponds to the material's linear stress-strain region. Thus, it is checked that when a small axial stress is acting on the free endpoint of the rod in its initial state ($\sigma = E\epsilon = 0.05\ MPa \cdot 0.0009 = 45\ Pa$), the resulted strain coincides with the expected strain from theory $\epsilon = 0.0009$. The details of this sanity check can be found in the Appendix A.2.

Configuration of the damping coefficient

From Figures 3.7 and 3.6 it can be clearly seen that even though only one actuator produced a step force of 0.1N, the robot just elongated along the tangent of its endpoint, instead of performing the expected movement i.e a bending movement along with elongation. The reason for that unexpected behavior is the choice of damping coefficient. In particular, due to the fact that a high damping coefficient was chosen during the time-step tuning experiments, the rod is over-damped when it comes to the less stiff deformation mode i.e. bending. By decreasing the value of the damping coefficient until the rod seems sufficiently damped and the expected dynamics are recovered, the damping coefficient can be tuned. By following this procedure, it is concluded that the damping coefficient should be $c = 0.1s^{-1}$ (Figure 3.8).

Putting everything together, the simulation parameters are tuned. More precisely, the number of spatial discretization elements is fixed at $n = 30$, the time-step is chosen as $dt = 0.0001667\ s$ and the damping coefficient is set equal to $c = 0.1\ s^{-1}$. By simulating the rod's behavior for 10 seconds while having only one actuator producing a constant force of 0.1N (Step Input), the end-effector of the tuned model performs the expected movement (Figure 3.9).

As it can been seen from Figure 3.9, large changes in actuation forces produce large oscillations through the excitation of natural frequencies that the analytical linear damper cannot sufficiently damp out. These oscillations can even become unstable under some circumstances. Since the modelled system is essentially a hyper-elastic cantilever beam, it is possible that there is an infinite number of eigenfrequencies [84]. Therefore to avoid the excitation of eigenfrequencies which can cause large oscillations that can even become unstable during actuation, the force change per time-step is chosen to be bounded. More precisely, the maximum tolerable actuation force change per time-step is investigated. The objective is to find the maximum force change per actuator which does not produce large/unstable oscillations. As it can been concluded from Figure 3.9, the most under-damped oscillations appear in the stiffer deformation mode of elongation. This is expected of

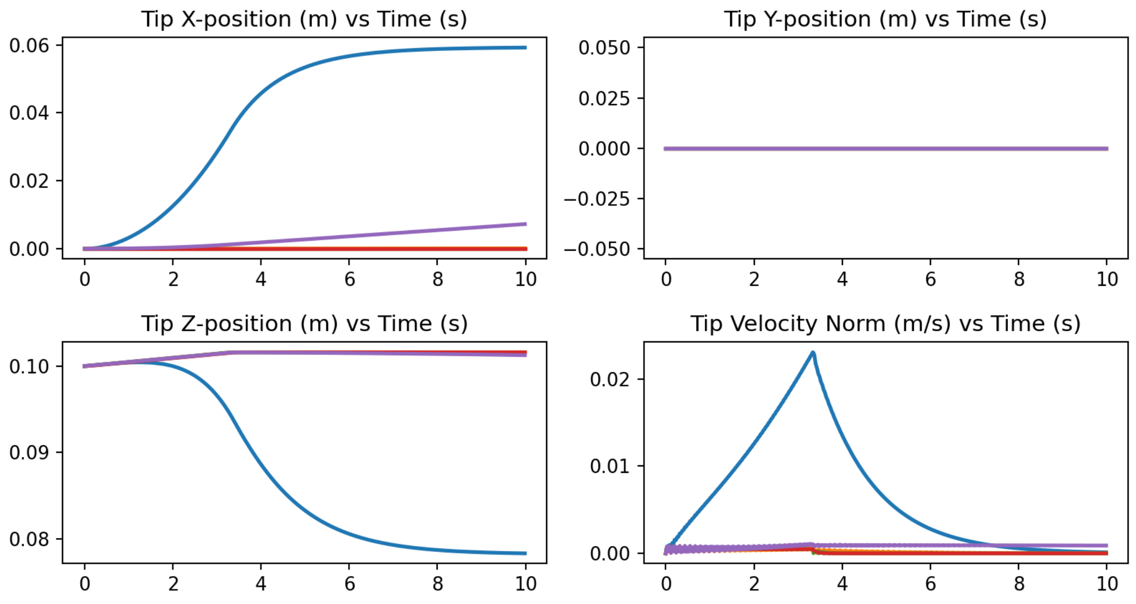


Figure 3.8: End-effector response with damping coefficients ranging from $0.1\ s^{-1}$ to $1000\ s^{-1}$. More precisely, orange line corresponds to $c = 1\ s^{-1}$, green line to $c = 10\ s^{-1}$, red line to $c = 1000\ s^{-1}$, purple line to $c = 0.5\ s^{-1}$ and blue line to $c = 0.1\ s^{-1}$. Green and orange lines are covered by the red line. It is clear that the expected dynamics are recovered for $c = 0.1\ s^{-1}$.

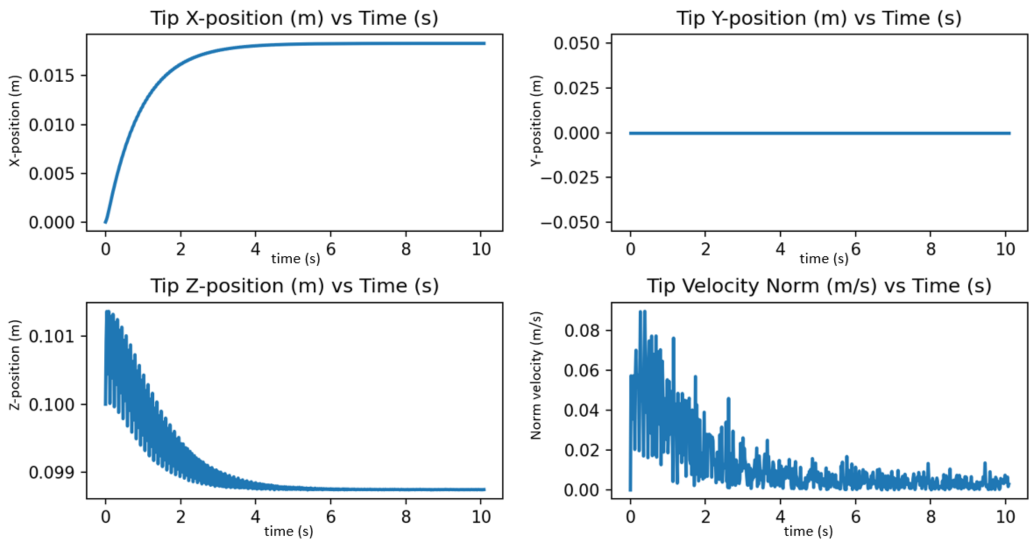


Figure 3.9: Tuned model's end-effector position and velocity norm when only one actuator produces a constant force of 0.1N (Figure 3.10, bottom right graph). The actuation profile is the same as in time-step tuning test)

course due to the fact that the shear/stretch Stiffness matrix contains much higher values compared to the bend stiffness matrix. To quantify this statement, the elements of the Bending Stiffness matrix for the last edge of the rod at its initial state

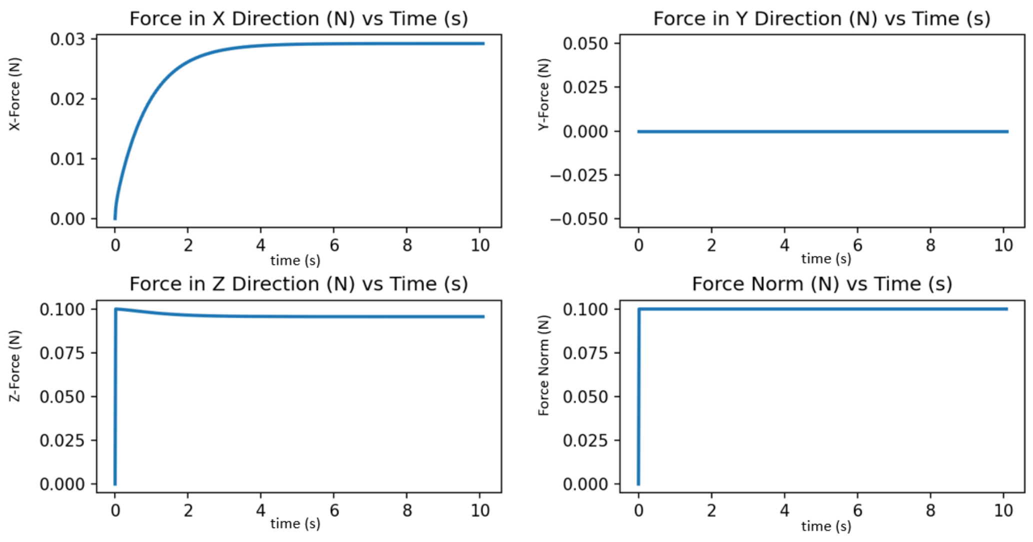


Figure 3.10: Actuation forces acting on the tuned soft robot model.

is:

$$\boldsymbol{B} = \begin{bmatrix} 0.00025765 & 0 & 0 \\ 0 & 0.00025765 & 0 \\ 0 & 0 & 0.00017177 \end{bmatrix} \tag{3.6}$$

while the corresponding shear stiffness matrix is:

$$\boldsymbol{S} = \begin{bmatrix} 4.08968044 & 0 & 0 \\ 0 & 4.08968044 & 0 \\ 0 & 0 & 12.72345025 \end{bmatrix} \tag{3.7}$$

Therefore, the effect of force change per time-step is firstly checked on a strict elongation case, i.e. actuators' forces are equal in magnitude (Appendix A.3). It is concluded that by enforcing $10^{-5}N$ force change per time-step, no instabilities emerge and the system's response can be characterized marginally stable since small under-damped oscillations are present. However these oscillations are much smaller in magnitude compared to the system's step response 3.9. The under-damped oscillations are unavoidable since only one damping coefficient can be specified and deformation modes with very different stiffnesses are present i.e. Bending and Stretching/Compression.

Considering the above, by enforcing this bounded force change, each actuator can increase/decrease the force it produces by $10^{-5}N$ N or keep it constant. Mathematically this can be stated as $df \in \{-10^{-5}, 0, 10^{-5}\}$. Finally, to test a combined movement, i.e. elongation along the tangent and bending, the force of one actuator is increasing by $10^{-5}N$ per dt until 3.3 seconds and then it is kept constant, while the rest of the actuators are not producing any force. The described actuation profile is presented in Figure 3.11 while the rod's end-effector response is depicted in Figures 3.12 3.13. As it can be seen from the graphs, the modelled robot's end-effector (or rod's endpoint) reaches its final position stably, without oscillations.

The behavior of the tuned model when choosing a larger number of spatial discretization elements is examined in the Appendix A.4.

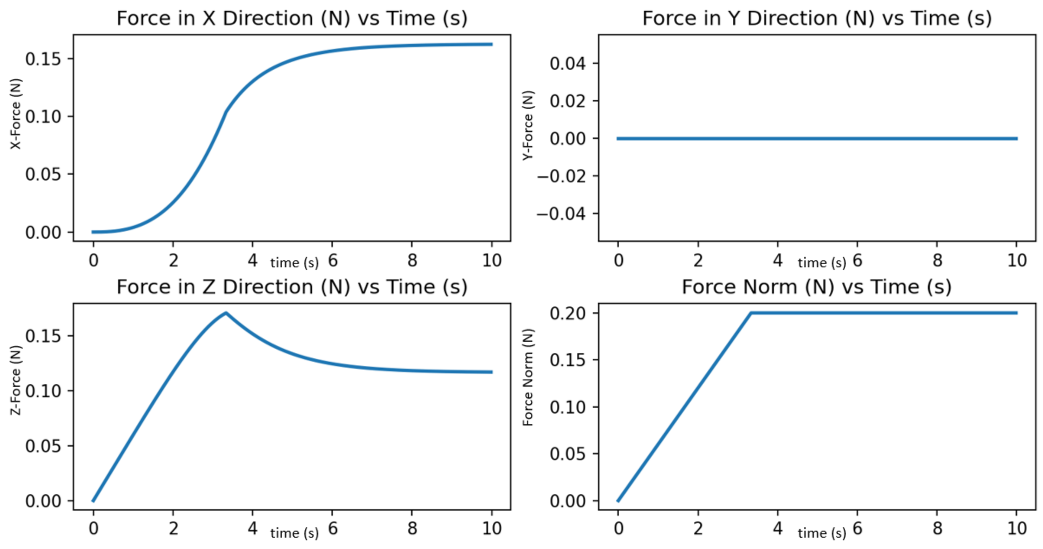


Figure 3.11: Elongation & Bending in X-Z plane. Only one actuator is activated and force Change per time step is +$10^{-5}N$ until t=3.3 seconds. Then, the force is kept constant. The force input is essentially a ramp.

3.2 Coupling Elastica with Model-Free RL

In section 3.1 the followed process in order to model and simulate the investigated soft robot and its 3D workspace, in Elastica, was analyzed. Until that point, there was no involvement of RL. During that process, all the necessary components for modelling and simulating the investigated soft continuum robot in 3D space were obtained. Thus the simulation environment is designed and presents the desired characteristics and as a result coupling it with Model-Free RL is the next step towards the present thesis' objective: Model-Free RL Control. In order to couple the produced Elastica simulation with Model-Free RL, an interface between Elastica and Stable-Baselines3 [85] is created. Stable-Baselines3 constitutes a set of implementations of Model-Free RL algorithms in PyTorch [85]. The simulation environments that are used in Model-Free RL applications based on Stable-Baselines3, are either

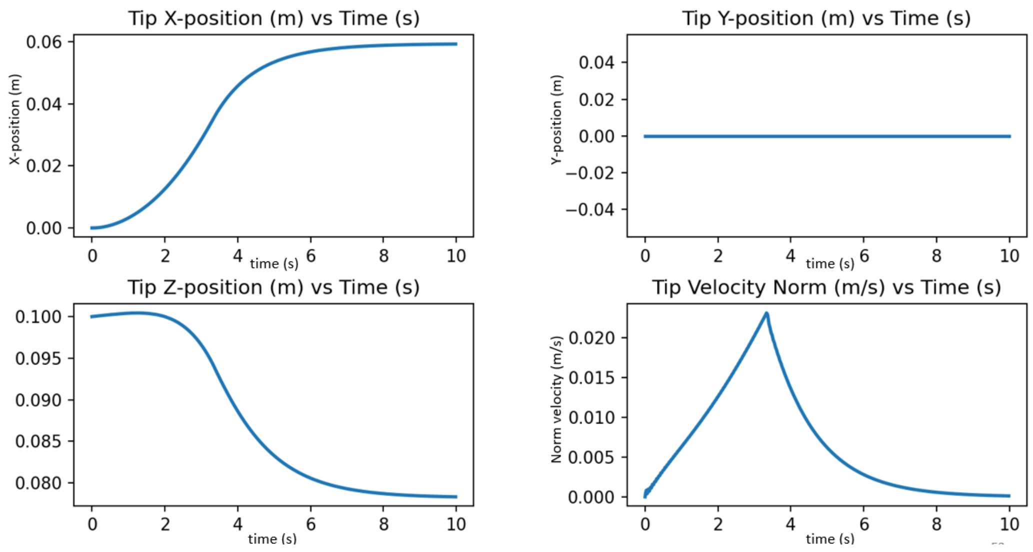


Figure 3.12: Modelled robot's end-effector position under the actuation profile described in Figure 3.11

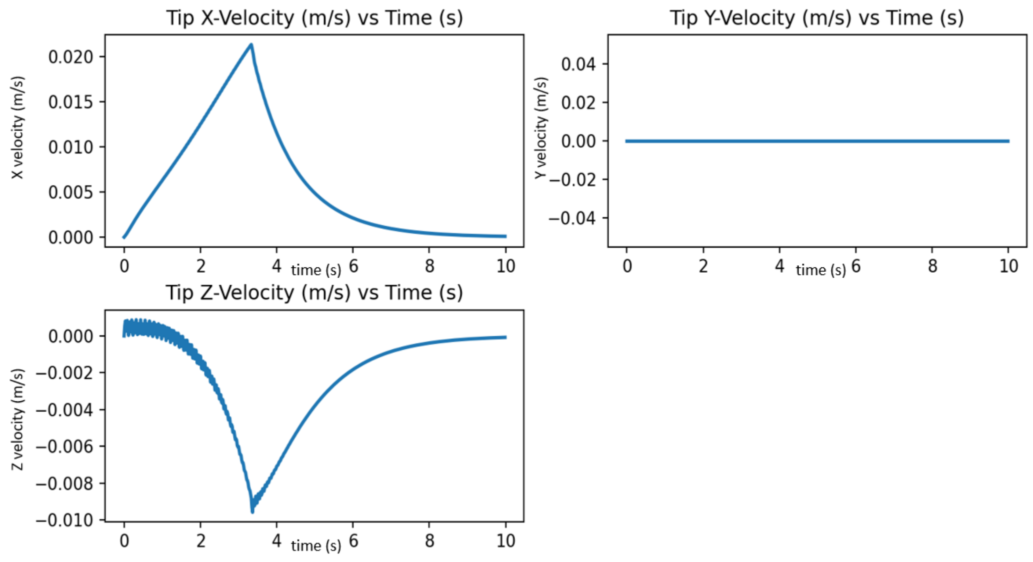


Figure 3.13: Modelled robot's end-effector velocity under the actuation profile described in Figure 3.11

derived from OpenAI Gym Environments (e.g. Mujoco) or they are custom environments which follow OpenAI Gym's application programming interface [85]. In particular, OpenAI Gym provides an application programming interface (i.e. API) for RL environments which involve one single agent [86]. Consequently, the Elastica simulation discussed in section 3.1 must be encapsulated as a single agent RL environment which follows OpenAI's API. This is possible since the soft robot is the only agent in the examined RL problem and in Elastica the soft robot and its 3D workspace are simulated. The encapsulation of the Elastica simulation environment is done through the Gym's API key functions [86]. The core idea is that by following Gym's API the created Elastica environment is reconstructed as a Markov Decision Process, rendering its coupling with RL possible. The process of this reconstruction is described in the following lines.

Firstly, a class called "Environment" is created which inherits methods and attributes from the Gym's Environment class. This is needed in order to utilize the Gym's key functions. Then the "Environment's" attributes and parameters are defined. For example spatial and time discretization parameters and simulation duration. Since an MDP representation is desired, the type of observation and actuation space associated with the environment are also defined at that point. Subsequently, in order to create the Cosserat rod (agent) and its 3D space the Gym's **reset()** function is utilized [86]. More specifically, this class method, resets and creates the simulation environment. First, the rod is initialized and the boundary conditions which act on it are defined. Any extra simulated bodies can be initialized and appended to the simulation via this method. Finally, callback functions are defined here for data collection purposes.

In the examined case, the Elastica rod (modelled soft robot) moves inside a 3D space due to the action of external forces (modelled pneumatic actuation). In other words, actions in the form of forces are picked by the agent (Elastica Rod) in order to move in 3D space. As a result, expressing this process in MDP format, the agent should receive a new observation from the updated environment as well as an immediate reward for the taken action. In a position control framework, the reward should be positive as the tip of the rod moves closer to the target and negative when it moves away. The new observation should be received since the taken action results in state transition as discussed in section 2.1. In the context of the present thesis, when an action is taken and a new observation is received as a result, a learning step is performed. Therefore, the learning step can be differentiated from the simulation time-step since a new action can be picked by the agent at a lower frequency than simulation's time-sampling frequency. Of course in the investigated case, the total number of learning steps are defined by the action-taking frequency and the total duration of the Elastica simulation i.e. the time frame the Elastica rod has at its disposal to reach the target. In MDP terminology, this time frame during which the Elastica rod interacts with its environment can be called an episode. Nonetheless, after some learning steps, the episode may end prematurely due to numerical issues or excision of rod's physical limits. In such a case, the episode ends even if the maximum number of learning steps are not reached. However, due to the modelled actuation in Elastica, these issues are very unlikely to appear. As already implied an episode has a specific duration. If the total number of learning steps which constitute the episode are exceeded, then the episode ends. OpenAI Gym's

step() class method is utilized in order to implement the aforementioned sequential state-action-reward-new state process as well as the episode termination process. In order for a new episode to start, i.e.start a new Elastica simulation, the created ”Environment” has to be restarted via **reset()** class method. Implementation details regarding the encapsulation of Elastica Environment via the Gym's functions can be found in the thesis' supplementary Python files.

Definition of Observation Space, Action Space and Reward Function

Since the soft robot simulation is reconstructed as a MDP, its observation space and actuation space have to be defined. The model-free RL approach that is followed in the present thesis renders the reconstructed MDP incomplete i.e. the transition probability matrix P is unknown and the reward function $R(s, a)$ needs to be manually designed. However, in the examined case, it is possible for the observation space to coincide with the MDP's state space since the created custom ”Environment” can be completely observable by the agent. Nevertheless, for the needs of the examined position control problem, a smaller observation space that consists of the possible positions and velocities of the robot and the target, is deemed sufficient. In particular, since only stationary targets are considered in the investigated position control problem, the observation space is composed by the 3D positions of 11 equidistant vertices along the rod as well as by the 3D position of the target and the velocity of the robot's tip (magnitude and direction). It is worth noting however that the velocity of the target can also be included in case of a position control problem with a moving target. Thus, the observation matrix consists of 40 elements and it is specified as a matrix with continuous elements that range from $-\infty$ to $+\infty$. Thus, the observation space is continuous as in almost all robotics RL position control problems. The choice of a higher dimension state-space is based on the fact that an estimation of the whole rod's position provides more information (i.e. more trajectory data) and this state representation can also be used for more complicated position control tasks such as reaching a target with a specific configuration.

Due to the fact that the force change per time-step is bounded to $10^{-5}N$, the otherwise naturally continuous action space becomes discrete. In particular, each one of the four modelled actuators can either decrease or increase its produced force by $10^{-5}N$ or keep it constant. Therefore, a multi-discrete action space is defined and its action is described by a vector of four elements where each one of them can take three values 0,1,2. These discrete values represent the three possible actions of each modelled actuator i.e. 0: keep the force constant, 1: increase the force by $10^{-5}N$ 2: decrease the force by $10^{-5}N$. An example of an action can then be $\alpha = [0, 1, 1, 2]$ where the first actuator keeps its force constant, the second and third increase their force by $10^{-5}N$ and fourth decreases its force by $10^{-5}N$. For a 10 second simulation the produced forces do not exceed 0.6N. Hence, the rod's physical limits are not violated and a realistic actuation is modelled since the physical system's pneumatic actuators typically produce small forces ($F \approx 0.4N$). Moreover, the actuation space is filtered so that the actuator's force does surpass 0.6N and simultaneously it does not reach negative values, thereby following the physical actuation as faithfully as possible.

Finally, the reward function is manually designed to reward the agent as it gets closer to the target. Therefore the immediate reward that the agent experiences when being at state s and taking action α is $r(s, a) = -||x_t - x_{tip}||$ where x_t is the current position of the target and x_{tip} is the current position of the rod's free endpoint. If the simulation becomes unstable a negative reward of -500 is given to the agent. In chapter 4, different variations of the reward function are examined and their corresponding results are presented.

3.3 Control via Proximal Policy Optimization

Stable-Baselines offer multiple Model-Free RL algorithms which can be utilized for training purposes. However, it is important to clarify that not all RL algorithms fit the examined problem. The MDP is the examined RL problem is characterized by a continuous observation space and a multi-discrete actuation space. Therefore, the Model-Free RL algorithms that are compatible with the investigated MDP are Trust-Region Policy Optimization (TRPO) [64], Proximal Policy Optimization (PPO) [63] and Advantage Actor Critic (A2C) [87]. However, Proximal Policy Optimization has been chosen for the following reasons: It combines ideas of TRPO and A2C since it uses a trust region for actor improving while utilizing multiple workers but on one hand,PPO is easier to implement and tune than TRPO [63] and on the other hand A2C is a special case of PPO [88]. Thus the transition from PPO to A2C is considered much easier [88] in case it is deemed worthy of future investigation for the examined RL control problem. Finally, PPO as an actor-critic method has characteristics which are deemed beneficial for the investigated control problem. In particular, it updates the policy directly while not experiencing the high variance of strictly policy gradient methods [48]. Furthermore, a stochastic policy can be learned by the agent and a certain degree of exploration is guaranteed. Secondly, PPO as all policy gradient methods that utilize neural network function approximators, it presents better performance in control problems with large actuation space [89].

Chapter 4

Results

Each episode is chosen to last for 10.01 seconds which is equivalent to 60059 simulation time-steps since $dt = 0.1667\ ms$. Nevertheless, the time-sampling frequency required by the Elastica simulation is considerably higher than the action-taking frequency i.e. controller update frequency, needed to control the soft robot in the designed MDP. Therefore, the controller update interval is decoupled from the simulation time-step and Elastica takes multiple simulation time-steps between each controller update. In particular, 1000 simulation time-steps are taken before the controller updates thereby rendering the controller update interval equal to 0.167 seconds while each episode consists of $int(\frac{60059}{1000}) = 60$ learning steps. As a result, the possible trajectories during an episode are significantly reduced and thus the problem's complexity is reduced while policy-training becomes faster. However, at the same time the system's accuracy is decreased since the reduction of the controller update frequency renders the reaching of certain states impossible. More precisely, The force change per controller update per actuator is 0.01 N. In order for the robot to reach a stationary target and stay on the target, the target must become the new equilibrium point for the robot's free endpoint (sum of forces/torques exerted on the free endpoint are zero i.e. $\Sigma\vec{F} = 0, \Sigma\vec{\tau} = 0$). The larger force jumps decrease the range of possible new equilibriums. For example, starting from the initial state, if one learning step with action $\alpha = [0\ 0\ 1\ 0]$ is performed and for the rest 59 learning steps $\alpha = [0\ 0\ 0\ \ 0]$ so that a bending movement in the x-z plane occur and a new equilibrium is reached, targets at euclidean distance shorter than 0.93mm cannot be reached. This can be clearly seen in Figures 4.1 and 4.2.

Due to the introduction of discrete force changes per step i.e. $-10^{-5}N, 0, +10^{-5}N$ per time-step or $-0.01N, 0, +0.01N$ per learning step, the MDP's actuation space becomes multi-discrete and Proximal Policy Optimization is utilized in order to train the RL controller (i.e. policy) [85]. In particular, the Proximal Policy Optimization variant with the clipped-surrogate is used [85]. The algorithm's hyperparameters are not optimized using an optimization method, however certain hyperparameters are

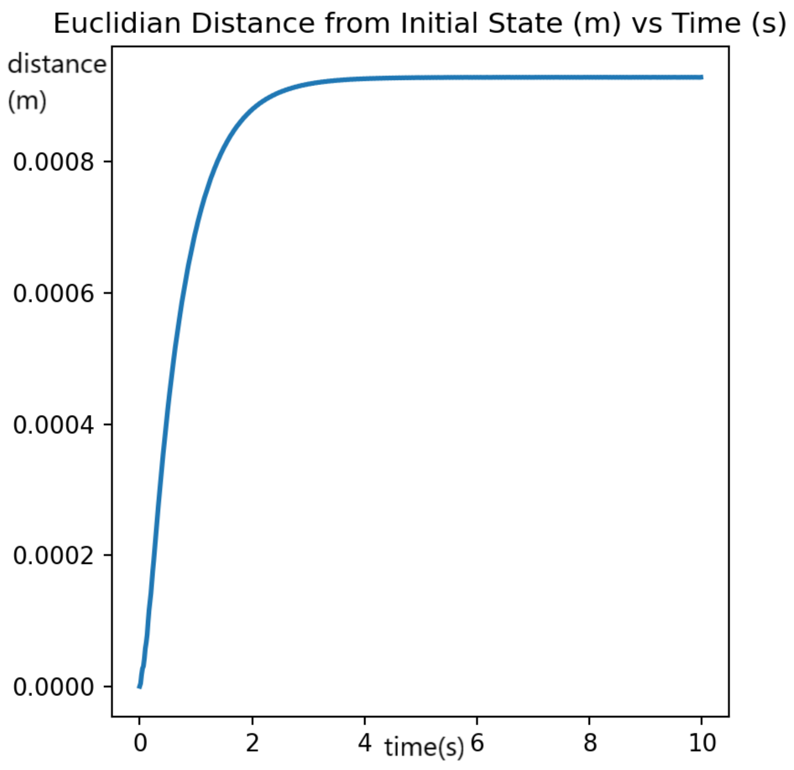


Figure 4.1: Euclidean distance of the soft robot's free endpoint from the initial state

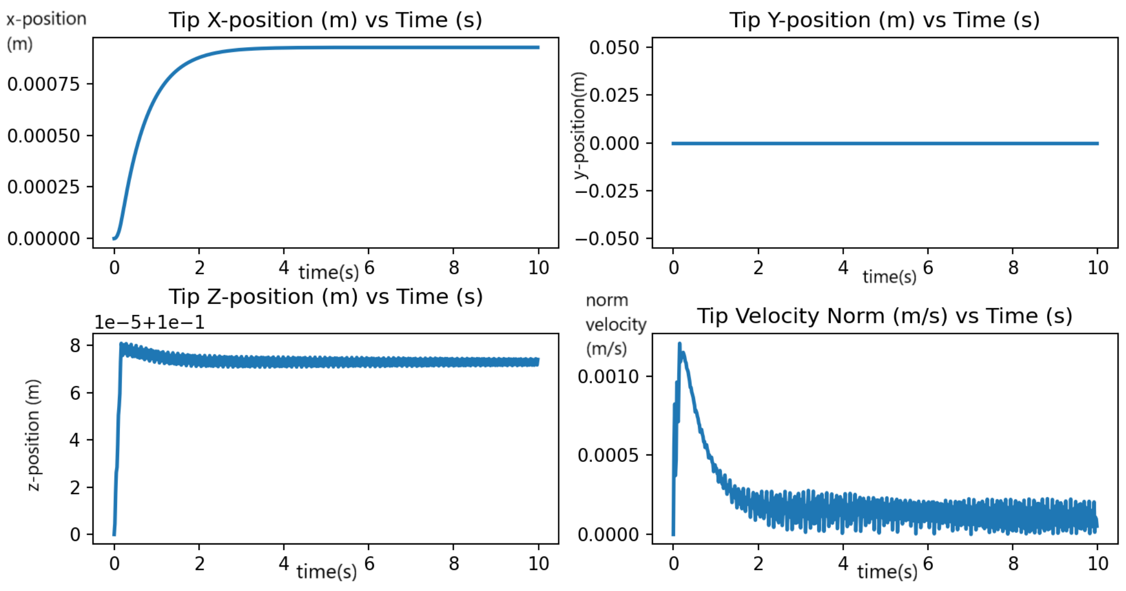


Figure 4.2: Soft Robot's free Endpoint position and velocity norm. The new equilibrium is [0.927,0, 100.07] in mm and oscillations around that point and along the tangent vector occur, due to high stretch stiffness.

tuned based on the nature of the investigated RL problem and observations coming from training data. More specifically, the hyperparameter "nsteps" is set $nsteps = 60$.

This hyperparameter expresses the number of experiences that are collected from the environment (i.e. Elastica simulation as MDP) while following the current policy's suggestions, before it is updated. Therefore it was deemed a reasonable choice to set its value equal to the episode length. The mini-batch size ($batchsize$) varies from 60 (equal to the episode length) to 15.

Variations of Reward Function Design

Firstly, the impact of the reward function design is examined through a set of 3 experiments. In all 3 experiments the goal is the same: The robot's end-effector must reach a stationary target and stay on the target, within 10.01 seconds. However, different variations in the reward function are tested to examine which of them can potentially lead to faster convergence to the optimal policy. The stationary target is set on: [0.0, -0.0836, 0.0462] thus, the robot's end-effector is 9.94 cm away from the target. It has been checked that this point is within the robot's workspace however it is not certain that the robot's end-effector can be ”sufficiently stabilized” on that target. The term ”sufficiently stabilized” is used for two reasons: 1) It is known from the analysis in chapter 3, that oscillations are under-damped in the direction of end-effector's tangent and thus these oscillations do not damp out within the 10.01 second time frame. 2) Due to the action-taking frequency it is not certain that the chosen target can become the end-effector's new equilibrium point.

Finally, the default PPO hyperparameters [85] are used in all 3 scenarios except for $nsteps$ and $batchsize$ which are both set equal to 60.

Scenario 1

The reward function is:

$$R(s) = \begin{cases} -||x_t - x_{tip}|| & \text{if} \quad stable\ simulation \\ -500 & \text{if} \quad unstable\ simulation \end{cases} \tag{4.1}$$

where x_t is the position of the target and x_{tip} is the current position of the robot's end-effector. If the euclidean distance between the robot's tip and the target becomes smaller, the reward becomes larger. If the simulation becomes unstable the episode is truncated and a negative reward of -500 is given to the agent. It is worth noting that the designed actuation cannot lead to instability though. Training is done for 500 episodes which corresponds to 30000 learning steps in total, and the produced results are shown in Figures 4.3 and 4.4.

As it can be seen from Figure 4.3, the error is rapidly decreasing until roughly t=5.7s and then it slowly settles at a point 2.03 mm away from the reference position while presenting small oscillations (the system is under-damped in the tangential direction). The corresponding cumulative reward when following the obtained policy is -2.31. Since it is not certain that the specified reference point can become an

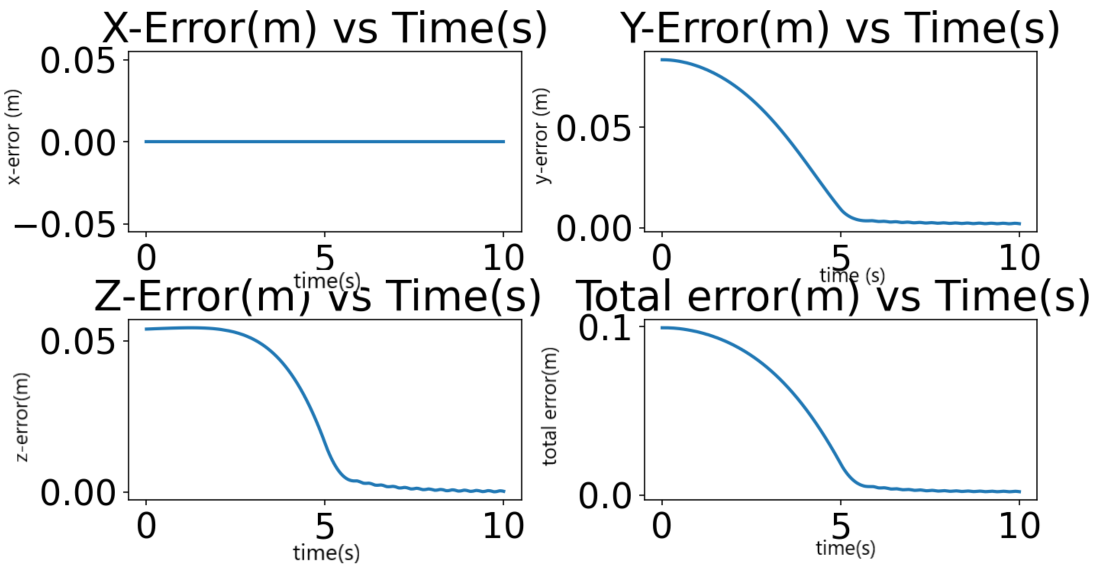


Figure 4.3: Position Error when using the trained controller from scenario 1.

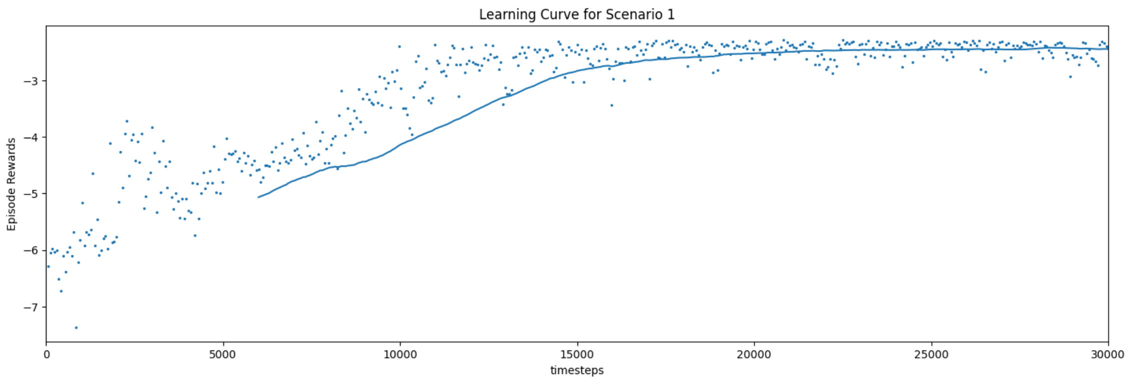


Figure 4.4: Learning Curve corresponding to scenario 1. The cumulative reward of each training episode is plotted against the learning steps thus each blue dot represents the cumulative reward of an episode. The blue continuous line represents the mean cumulative reward which stabilizes at roughly -2.46.

equilibrium point for the end-effector, it is hard to assess the produced controller. It is also uncertain if the training time is sufficient for the optimal policy to be found. On one hand the learning curve shows convergence (Figure 4.4), but on the other hand, a surge in average cumulative reward might occur after many steps. In theory, if the chosen target can be reached and the robot can stabilize its end-effector on it, the implemented reward function in this scenario should be sufficient. The end-effector behavior as presented in Figure 4.3, is very close to the desired behavior however it is difficult to assess if it is the optimal behavior or not. Nevertheless, the reward function in this case is deemed time-consuming since no extra reward is given based on the state (e.g end-effector velocity or acceleration) in which the end-effector should reach the target is provided.

Scenario 2

The reward function is:

$$R(s) = \begin{cases} -||x_t - x_{tip}(n)|| + (||x_t - x_{tip}(n-1)|| - ||x_t - x_{tip}(n)||) & \text{if} \quad stable\ simulation \\ -500 & \text{if} \quad unstable\ simulation \end{cases} \tag{4.2}$$

where x_t is the position of the target, $x_{tip}(n)$ is the current position of the robot's free endpoint $x_{tip}(n-1)$ is the robot's previous position. The basic difference with the previous scenario is that if an action leads to a larger jump towards the target, reward becomes larger, whereas an action that moves the robot away from the target results in a more severe punishment. The idea here is to check if the robot can learn to reach the target faster, i.e. move faster towards the target. Again, if the simulation becomes unstable the episode is truncated and a negative reward of -500 is given to the agent. Training is done again for 500 episodes which corresponds to 30000 learning steps in total, and the obtained results are presented in Figures 4.5 and 4.6.

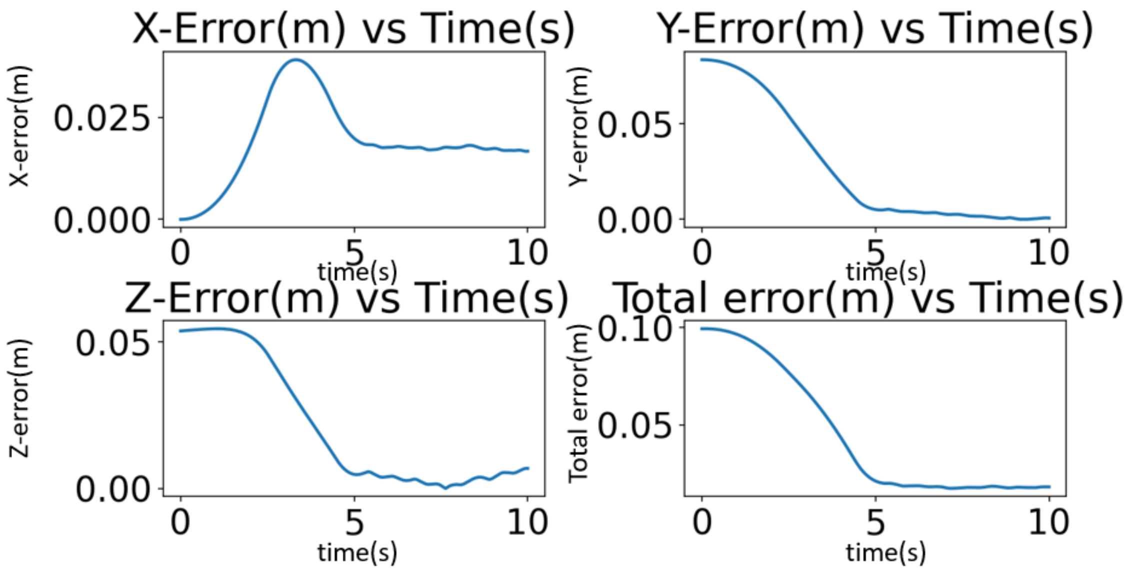


Figure 4.5: Position Error when using the trained controller from scenario 2.

As it can be observed from Figure 4.5, the total error is steadily reducing until roughly t=5s and then it starts stabilizing at 0.018m. This means that the end-effector is stabilized at a point approximately 1.8cm away from the target while collecting a cumulative reward of -2.55. The obtained policy is substantially worse

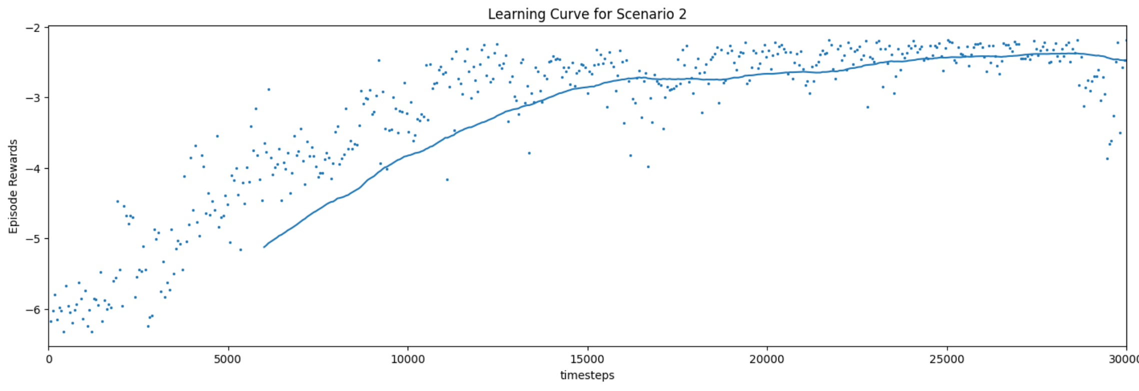


Figure 4.6: Learning Curve corresponding to scenario 2. The cumulative reward of each training episode is plotted against the learning steps thus each blue dot represents the cumulative reward of an episode. The blue continuous line represents the mean cumulative reward which is approximately -2.58 by the end of the training while showing decreasing tendencies.

compared to the previous scenario. From Figure 4.6 it can be concluded that the training time is probably not sufficient since the expected cumulative reward is not stabilized and even tends to decrease. Nonetheless, it seems that the reward design in this case resulted in a substantially worse performing policy while utilizing the same training time as in the previous scenario. This is expected since higher reward is given to the agent when the end-effector is getting closer to the target faster. However, reaching the target with higher velocity results in the end-effector diverging from the desired target and convergence to the optimal policy might need a lot more training episodes thereby rendering this reward function sub-optimal for the investigated control task.

Scenario 3

The reward function is:

$$R(s) = \begin{cases} -||x_t - x_{tip}|| & \text{if} \quad ||x_t - x_{tip}|| > 1.3\, mm \\ -||x_t - x_{tip}|| + ((initial\ dist) - ||x_t - x_{tip}||) - 0.1||a_{tip}|| & \text{if} \quad ||x_t - x_{tip}|| \leq 1.3\, mm\ and\ ||a_{tip}|| < 0.0035 m/s^2 \end{cases} \tag{4.3}$$

where x_t is the position of the target, x_{tip} is the current position of the robot's free endpoint and $initial\ dist = ||x_t - x_{tip}(0)||$ is the initial distance between the end-effector and the target. If the robot's free endpoint is away from the target at a distance less or equal to 1.3mm (defined tolerance for the problem) and its acceleration is practically zero, an extra reward is given, which gets larger as the acceleration norm of the tip gets closer to zero and current distance to the target becomes smaller. In that way, a new equilibrium as close to the reference position (target) as possible, can be found for the robot's end-effector. In order to account for the fact that a target might be inside the robot's workspace but cannot be reached due to the actio-taking frequency, the tolerance ratio of 1.3mm is introduced. Simultaneously, since the reward function gives an extra reward when a point within the tolerance range of 1.3mm, becomes the end-effector's new equilibrium, exploration should be reduced thereby leading to faster convergence to the optimal policy. Furthermore, the impact of the acceleration in the extra reward is scaled by 0.1 since already end-effector acceleration norms less than $3.5mm/s^2$ are considered practically zero given the fact that under-damped oscillations occur in the direction of the end-effector's tangent. The penalty of -500 is removed since the designed actuation system cannot lead to simulation instabilities. Training is done for 500 episodes which corresponds to 30000 learning steps in total, and the corresponding results are shown in Figures 4.7 and 4.8. As it can be seen from Figure 4.7, the error is steadily reducing until roughly t=4s and then it stabilizes at roughly 0.006m. The robot's end-effector reaches its final position which deviates from the target by 6mm in the x-axis, while presenting an oscillatory behavior in the tangential direction (system is underdamped in that direction). The produced policy did not lead the end-effector inside the tolerance range, however the accumulated reward while following this policy amounts to -1.82, which significantly surpasses the cumulative reward from scenario 1. That means that a better policy was found while utilizing the same training time. The improvement of the policy can also be confirmed by Figure 4.7, since the total error decrease is steeper compared to scenario 1. In particular, even though the final total position error is larger by 4mm compared to scenario 1, the end-effector gets closer to the target faster, which means that the series of actions chosen by the trained policy were keeping the end-effector closer to the target during the episode. The evolution of the average episodic cumulative reward during training for scenarios 1-3 is presented in Figure 4.9. It can be observed that the reward function of scenario 3 achieved the highest mean cumulative reward. Considering the above, the reward function of scenario 3 is deemed the most suitable choice for the present research. This is confirmed by the corresponding obtained numerical results as well as its theoretical foundation. Hence, the reward function of scenario 3 is used in the next series of experiments. This time, a stationary target located at a point that the robot's end-effector can reach and settle on, is considered.

Tuning Mini-Batch size

In the next series of experiments, the reward function presented in scenario 3 is utilized. The stationary target's coordinates expressed in global frame are [0.0592, 0.0, 0.0783]. It is verified that this point is within the robot's workspace and the end-effector can be stabilized at that point with insignificant error i.e. $< 0.5mm$

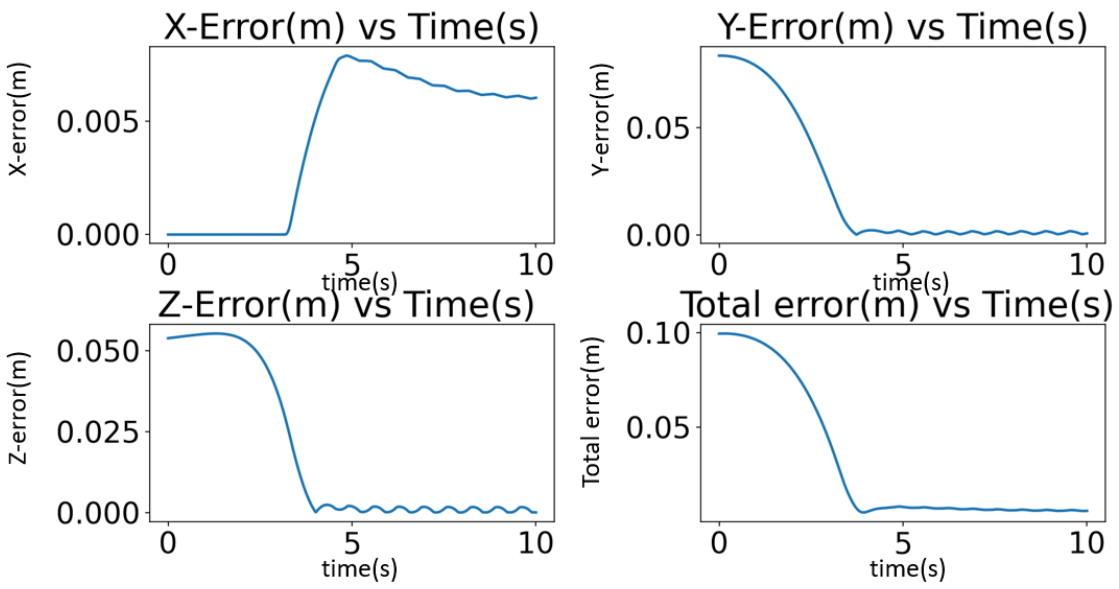


Figure 4.7: Position Error when using the trained controller from scenario 3

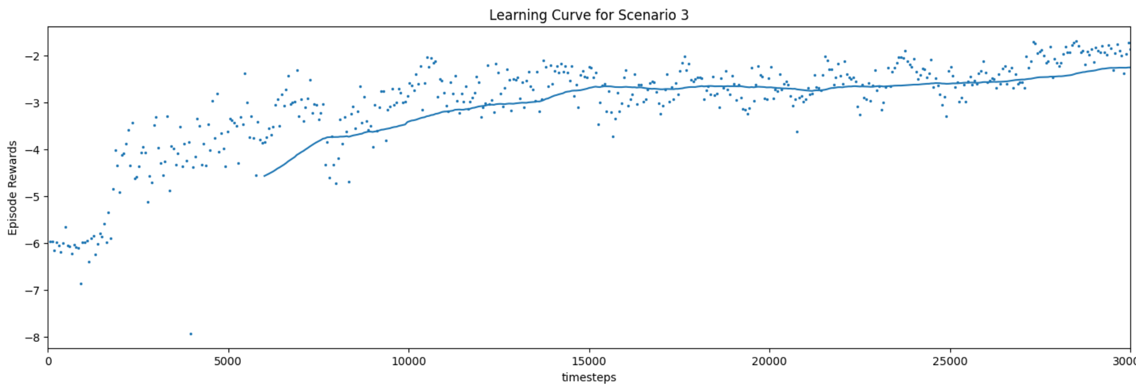


Figure 4.8: Learning Curve corresponding to scenario 3. The cumulative reward of each training episode is plotted against the learning steps thus each blue dot represents the cumulative reward of an episode. The blue continuous line represents the average cumulative reward which is approximately -2.22 by the end of the training while showing with increasing tendencies.

(Figure 3.12). Again, the simulation lasts for 10.01 seconds which corresponds to 60599 simulation time-steps and episode length of 60 learning steps. Hence, in the following experiments, $nsteps = 60$.

Scenario 1B: Mini-Batch size = 60 and nsteps = 60

Training is done for 1500 episodes which corresponds to 90000 simulation time-steps in total, and the corresponding results are shown in Figures 4.10 and 4.11.

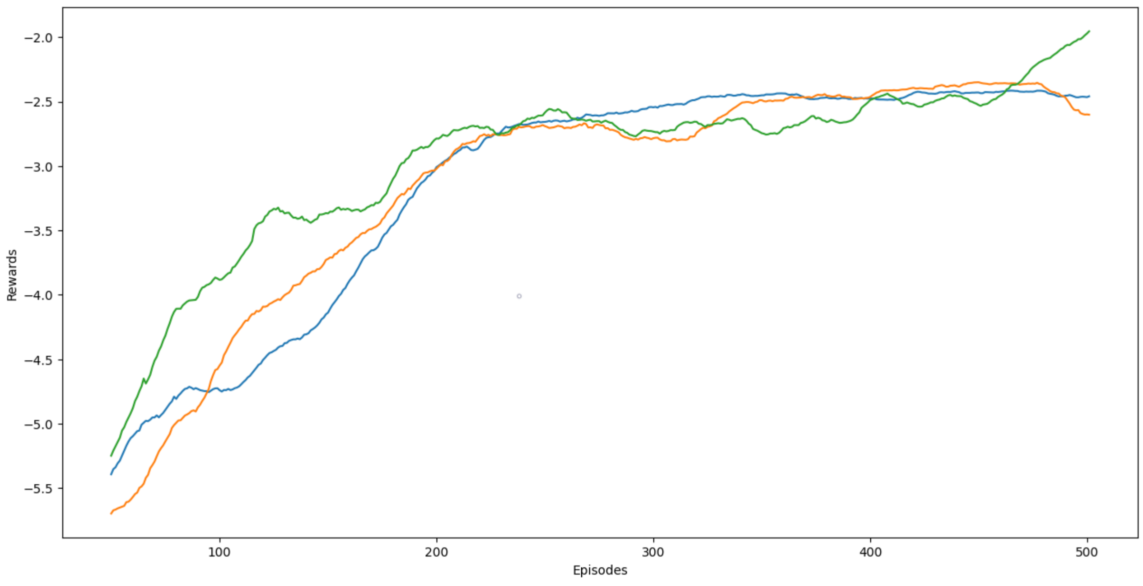


Figure 4.9: Average episodic rewards during training. Blue line corresponds to scenario 1, orange line to scenario 2 and green line to scenario 3. After training, the average episodic returns are -2.46.-2.58,-2.22 for scenarios 1,2 and 3 respectively.

By following the trained policy, the robot does not reach the desired target even though the error is consistently decreasing until roughly t=7s. After that point, the robot starts deviating from the target and by the end of the episode it is 5.6 mm away from the target. The cumulative reward collected while following the policy amounts to -1.22 while the average cumulative reward stabilized at approximately -1.3. It can be concluded that the optimal policy is not found, however the learning curve confirms that training is successful and stable since the average cumulative reward consistently increased (Figure 4.11).

Scenario 2B: Mini-Batch size = 30 and nsteps = 60

In the previous scenario, mini-batch size is set equal to batch size. Therefore, the benefits of mini-batch gradient descent used in PPO is not exploited and batch gradient descent is essentially used instead. Batch gradient descent is more pronoun to converging at a local minimum thereby slowing down the training process. On the other hand, the mini-batch gradient descent enables more gradient updates thereby ensuring transitions that will avoid local minima convergence while speeding up convergence [90, 91].

Considering the above, the mini-batch size is reduced to 30. Again, Training is done for 1500 episodes which corresponds to 90000 simulation time-steps in total, and the corresponding results are shown in Figures 4.12 4.13. As it can be seen from Figure 4.12, there is a slight improvement in the system's behavior

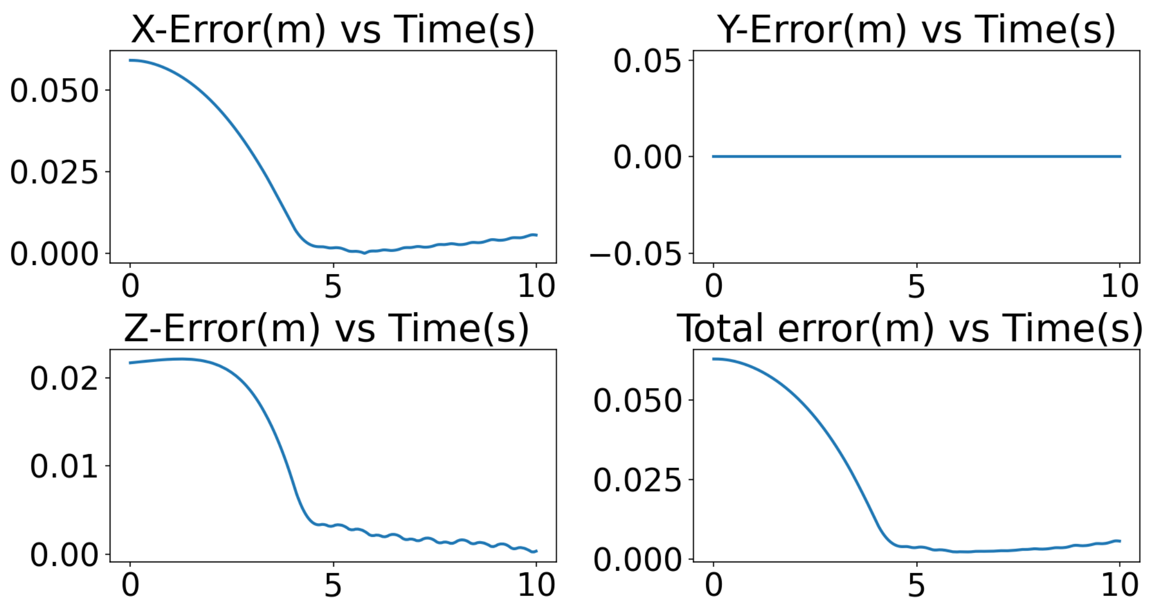


Figure 4.10: Position Error to the target against time for the obtained policy of scenario 1B

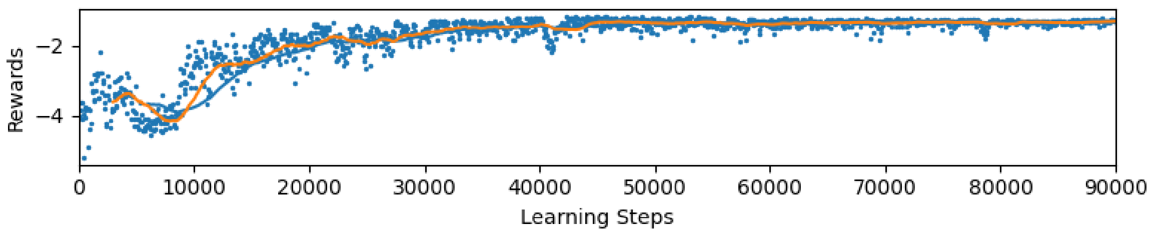


Figure 4.11: Cumulative rewards plotted against the learning steps (learning curve) for scenario 1B. Each blue dot represents the cumulative reward of an episode. Thus, each dot appears every 60 learning steps. The blue and orange continuous lines represent the average cumulative reward averaged over 100 and 50 episodes respectively.

since the obtained policy tends to stabilize the end-effector at a position roughly 9mm away from the target while the obtained cumulative reward for the followed trajectory (dictated by the trained policy) is -1.11, which is higher by 0.11 compared to the previous one in scenario 1B. The average cumulative reward settles at -1.20 (Figure 4.16

Scenario 3B: Mini-Batch size = 20 and nsteps = 60

In this scenario, the mini-batch size is set equal to 20. Therefore, each batch consists of 3 mini-batches. The produced results are shown in Figures 4.14 4.15. As it can be observed from Figure 4.14, the robot gets closer to the target and stays closer to the target for longer compared to scenarios 1B and 2B and a cumulative

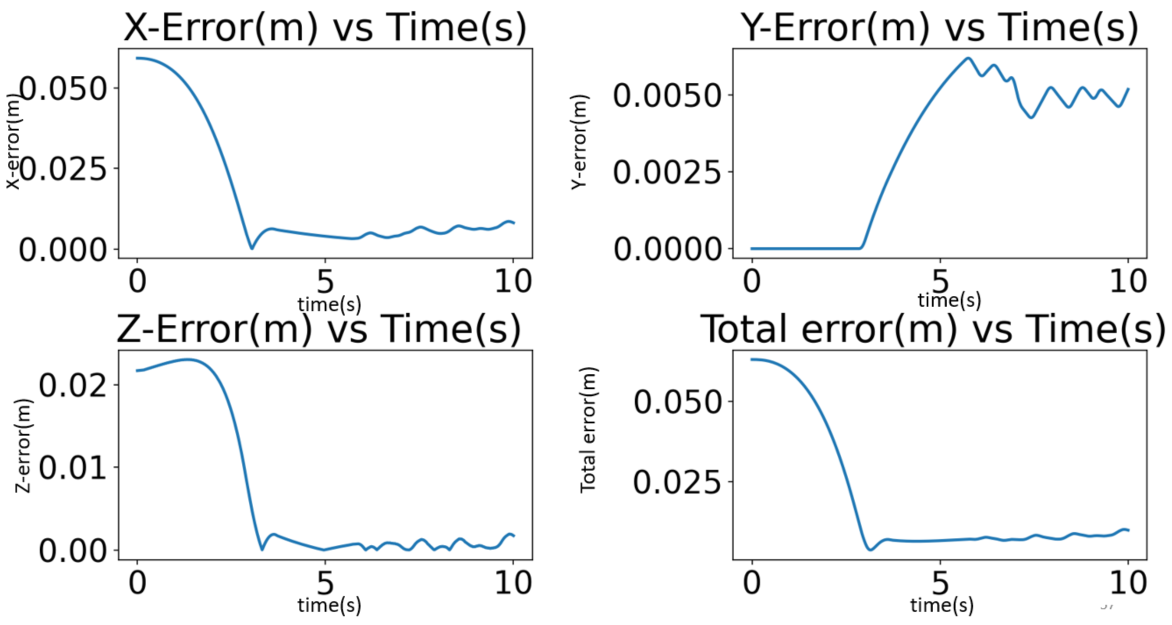


Figure 4.12: Position Error to the target against time for scenario 2B

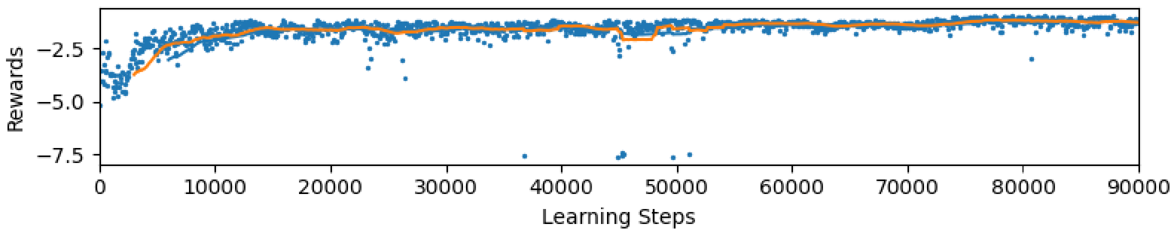


Figure 4.13: Cumulative rewards plotted against the learning steps (learning curve) for scenario 2B. Each blue dot represents the cumulative reward of an episode. Thus, each dot appears every 60 learning steps. The blue and orange continuous lines represent the average cumulative reward averaged over 100 and 50 episodes respectively

reward of -1.04 is collected. However, again the optimal policy is not found, and even though the end-effector stays approximately 6mm away from the target, deviating tendencies are observed in Y-axis (Figure 4.14). The average cumulative reward is roughly -1.22 by the end of the training period, matching the average cumulative of scenarios 1B and 2B. Nevertheless, the obtained policy is improved.

Scenario 4: Mini-Batch size = 15, nsteps = 60

In this scenario, the mini-batch size is slightly reduced and is set at 15. Thus, each training batch consists of 4 mini-batches. Furthermore, the training time is significantly increased, i.e. from 1500 episodes to 6000 episodes. The obtained

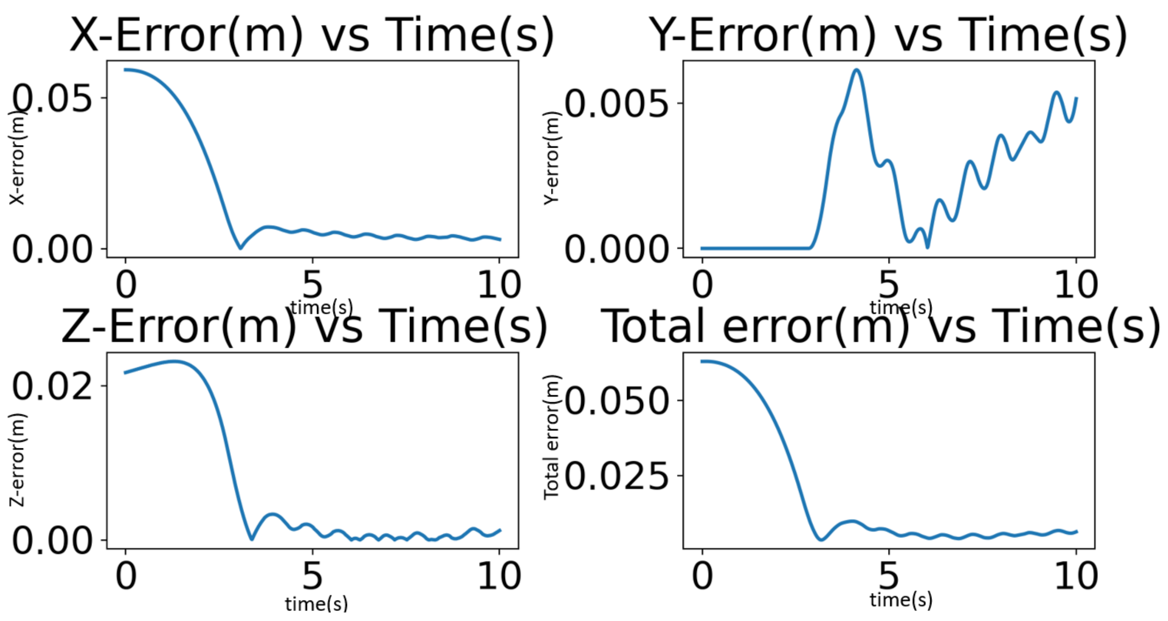


Figure 4.14: Position Error to the target against time for scenario 3B

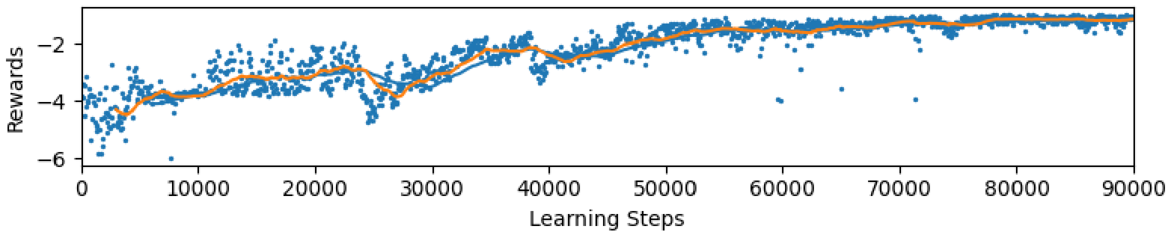


Figure 4.15: Cumulative rewards plotted against the learning steps (learning curve) for scenario 3B. Each blue dot represents the cumulative reward of an episode. Thus, each dot appears every 60 learning steps. The blue and orange continuous lines represent the average cumulative reward averaged over 100 and 50 episodes respectively

results are depicted in Figures 4.17 and 4.18. The produced policy is successful in executing the desired task. More precisely, the robot's end-effector after gradual diminishing oscillations reaches the target with a deviation of 0.3mm. However, it can easily be concluded that the obtained policy is not the optimal one since it does not produce the expected behavior of Figure 3.12. This can also be confirmed by the corresponding learning curve, since the average cumulative reward converged at -1.13 which is considerably lower than the cumulative reward that corresponds to the optimal policy.

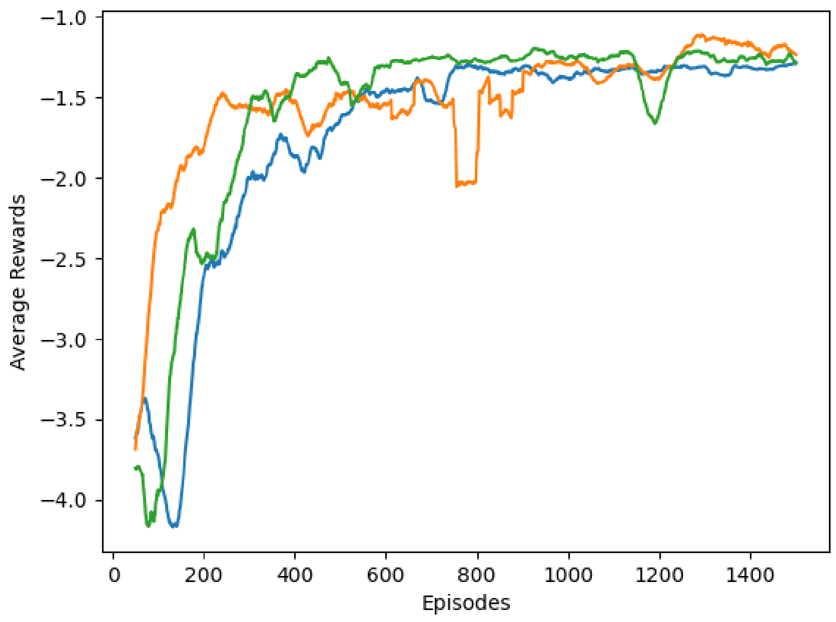


Figure 4.16: Comparison of average cumulative reward curves for scenarios 1B-3B. Blue,orange and green lines correspond to scenarios 1B, 2B and 3B respectively. Reduction of the mini-batch size tends to speed-up training and promote convergence to improved policies.

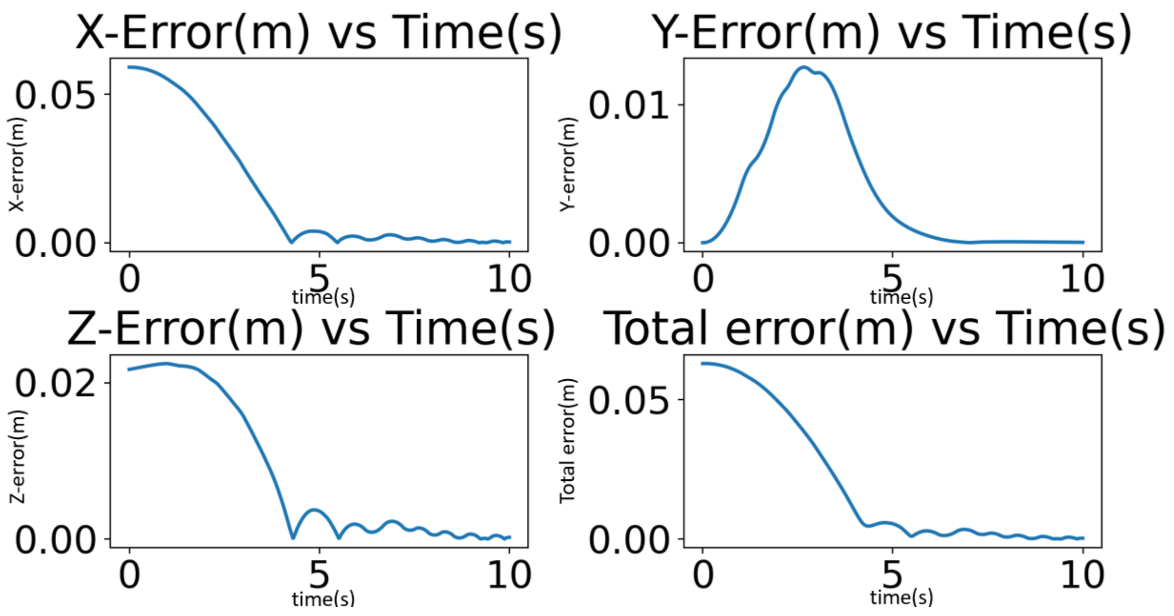


Figure 4.17: Position Error to the target against time for scenario 4

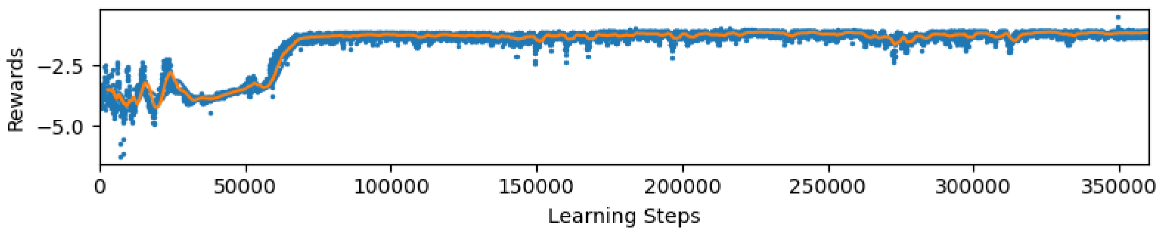


Figure 4.18: Cumulative rewards plotted against the learning steps (learning curve) for scenario 4. Each blue dot represents the cumulative reward of an episode. Thus, each dot appears every 60 learning steps. The blue and orange continuous lines represent the average cumulative reward averaged over 100 and 50 episodes respectively

Chapter 5

Conclusion

In the present thesis, the utilization of Model-Free RL as an autonomous control strategy for a pneumatic-driven soft continuum robot is investigated. The objective is to train the soft continuum robot to stabilize its end-effector on a stationary target in 3D space. In this research, the body of the examined soft continuum robot as well as its actuation and workspace are modelled in a simulator based on Cosserat rod theory and discrete differential geometry (Elastica). The chosen modelling approach, based on a consistently discretized Cosserat rod, is able to approximate the robot's dynamics and kinematics, thereby sufficiently capturing the physical system's behavior. The forces produced by the rPAM actuators can also be modelled with the use of the chosen simulator. The system's actuation is modelled as external forces which act along the free endpoint's tangent vector causing axial strain, and which upon magnitude difference produce torques that result in bending moments, thereby simulating the robot's pneumatic actuation. Furthermore, the system's expected behavior is obtained after proper tuning of the simulator's parameters such as number of spatial discretization elements, discretization time-step and damping coefficient while ensuring that the modelled actuation forces do not exceed realistic values and are applied in way that does not lead to instability. The created simulation environment is successfully transformed into a Markov Decision Process with a manually designed reward function compatible with the latest implementation of Proximal Policy Optimization (in Stable-Baselines3). Thus, the application of Model-Free RL control is made possible for the examined position control problem. More specifically, the use of a discontinuous reward function that provides extra rewards when the end-effector finds a new equilibrium point as close as possible to the set-point, is deemed the most suitable choice for the investigated control problem. It is also concluded that the use of mini-batches instead of the whole batch results in faster convergence to improved policies. Finally, through the implementation of the designed control scheme, the robot (Elastica rod) learns to stabilize its end-effector just 0.3mm away from the desired target.

The designed control scheme offers a series of benefits. Firstly, all policy training data are derived from the physical simulation environment and thus no real robot-environment interactions are needed. Hence, potential damages to the physical system due to its interaction with the environment are avoided. Furthermore, since virtual data are used, there is no dependency on sensor data which are difficult to collect and require filtering due to the presence of noise. Moreover, the investigated control scheme can easily be extended and employed for more complicated control tasks. In particular, the designed control scheme can easily be modified in order to train the robot to track a moving target. On top of that, through the utilization of PPO the implemented controller is essentially a stochastic policy. Therefore, through the usage of a stochastic policy, the investigated robot could learn to perform complicated tasks inside a dynamic and complex environment. For example, the robot could learn to avoid randomly moving obstacles. This is a particularly interesting scenario in the context of a biomedical application of the investigated robot. For instance, if the soft continuum robot is employed inside a confined space such as the human body during a surgery, it would be extremely important to ensure that the robot does not get in contact with vulnerable points (e.g. arteries). On the other hand, the followed approach is associated with certain limitations as well. Firstly, even though the utilized Elastica model is consistent in modelling the geometry, dynamics and kinematics of the soft continuum robot, its accuracy suffers compared to models based on FEM. In any case however, modelling the physical system without real data could lead in inaccuracies. Therefore, in the model-free approach of the present thesis, deviations when transferring the trained RL controller to the physical system, may occur.

Regarding future extensions of this research, it would be valuable to include gravitational forces in the model to achieve higher modelling accuracy and therefore higher control accuracy in real conditions. In the context of the present thesis, gravitational forces are omitted due to the fact that the system is extremely underactuated and the design of an efficient gravity compensation scheme is impossible. Nevertheless, gravitational forces should be taken into consideration in order to ensure the proposed control scheme's feasibility in a real situation. Moreover, since there is no knowledge of the real system's dynamics, it is difficult to assess the model's accuracy. Thus it is important to conduct model verification in the future in order to minimize deviations when transferring the controller to the physical system. Finally, hyperparameter optimization for PPO could potentially improve policy training and thus convergence to the optimal policy could be achieved.

Bibliography

[1] J. F. Wilson, “Compliant Robotic Structures - Part I,” School of Engineering, Duke University, Tech. Rep., 1985.

[2] ——, “Compliant Robotic Structures - Part II,” School of Engineering, Duke University, Tech. Rep., 1986.

[3] ——, “Compliant Robotic Structures - Part III,” School of Engineering, Duke University, Tech. Rep., 1987.

[4] O. Yasa, Y. Toshimitsu, M. Y. Michelis, L. S. Jones, M. Filippi, T. Buchner, and R. K. Katzschmann, “An overview of soft robotics,” *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 6, pp. 1–29, 2023.

[5] D. Rus and M. T. Tolley, “Design, fabrication and control of soft robots,” *Nature*, vol. 521, no. 7553, pp. 467–475, 2015.

[6] P. K. Singh and C. M. Krishna, “Continuum arm robotic manipulator: A review,” *Universal Journal of Mechanical Engineering*, vol. 2, no. 6, pp. 193–198, 2014.

[7] R. J. Webster III and B. A. Jones, “Design and kinematic modeling of constant curvature continuum robots: A review,” *The International Journal of Robotics Research*, vol. 29, no. 13, pp. 1661–1683, 2010.

[8] J. Burgner-Kahrs, D. C. Rucker, and H. Choset, “Continuum robots for medical applications: A survey,” *IEEE Transactions on Robotics*, vol. 31, no. 6, pp. 1261–1280, 2015.

[9] T. da Veiga, J. H. Chandler, P. Lloyd, G. Pittiglio, N. J. Wilkinson, A. K. Hoshiar, R. A. Harris, and P. Valdastri, “Challenges of continuum robots in clinical context: a review,” *Progress in Biomedical Engineering*, vol. 2, no. 3, p. 032003, 2020.

[10] K.-H. Lee, D. K. Fu, M. C. Leong, M. Chow, H.-C. Fu, K. Althoefer, K. Y. Sze, C.-K. Yeung, and K.-W. Kwok, “Nonparametric online learning control for soft continuum robot: An enabling technique for effective endoscopic navigation,” *Soft robotics*, vol. 4, no. 4, pp. 324–337, 2017.

[11] M. S. Espinoza, J. Gonçalves, P. Leitao, J. L. G. Sánchez, and A. Herreros, "Inverse kinematics of a 10 dof modular hyper-redundant robot resorting to exhaustive and error-optimization methods: A comparative study," in *2012 Brazilian Robotics Symposium and Latin American Robotics Symposium*. IEEE, 2012, pp. 125–130.

[12] I. D. Walker, "Continuous backbone "continuum" robot manipulators," *International Scholarly Research Notices*, vol. 2013, 2013.

[13] A. Bajo and N. Simaan, "Hybrid motion/force control of multi-backbone continuum robots," *The International journal of robotics research*, vol. 35, no. 4, pp. 422–434, 2016.

[14] Z. Mitros, B. Thamo, C. Bergeles, L. Da Cruz, K. Dhaliwal, and M. Khadem, "Design and modelling of a continuum robot for distal lung sampling in mechanically ventilated patients in critical care," *Frontiers in Robotics and AI*, vol. 8, p. 611866, 2021.

[15] X. Wang, Y. Li, and K.-W. Kwok, "A survey for machine learning-based control of continuum robots," *Frontiers in Robotics and AI*, p. 280, 2021.

[16] H.-C. Fu, J. D. Ho, K.-H. Lee, Y. C. Hu, S. K. Au, K.-J. Cho, K. Y. Sze, and K.-W. Kwok, "Interfacing soft and hard: a spring reinforced actuator," *Soft robotics*, vol. 7, no. 1, pp. 44–58, 2020.

[17] J. Burgner-Kahrs, D. C. Rucker, and H. Choset, "Continuum robots for medical applications: A survey," *IEEE Transactions on Robotics*, vol. 31, no. 6, pp. 1261–1280, 2015.

[18] G. Robinson and J. B. C. Davies, "Continuum robots-a state of the art," in *Proceedings 1999 IEEE international conference on robotics and automation (Cat. No. 99CH36288C)*, vol. 4. IEEE, 1999, pp. 2849–2854.

[19] B. A. Jones and I. D. Walker, "Practical kinematics for real-time implementation of continuum robots," *IEEE Transactions on Robotics*, vol. 22, no. 6, pp. 1087–1099, 2006.

[20] C. Hegde, J. Su, J. M. R. Tan, K. He, X. Chen, and S. Magdassi, "Sensing in soft robotics," *ACS nano*, vol. 17, no. 16, pp. 15 277–15 307, 2023.

[21] M. Cianchetti, T. Ranzani, G. Gerboni, T. Nanayakkara, K. Althoefer, P. Dasgupta, and A. Menciassi, "Soft robotics technologies to address shortcomings in today's minimally invasive surgery: the stiff-flop approach," *Soft robotics*, vol. 1, no. 2, pp. 122–131, 2014.

[22] A. T. Asbeck, R. J. Dyer, A. F. Larusson, and C. J. Walsh, “Biologically-inspired soft exosuit,” in *2013 IEEE 13th International Conference on Rehabilitation Robotics (ICORR)*. IEEE, 2013, pp. 1–8.

[23] S. Aracri, F. Giorgio-Serchi, G. Suaria, M. E. Sayed, M. P. Nemitz, S. Mahon, and A. A. Stokes, “Soft robots for ocean exploration and offshore operations: A perspective,” *Soft Robotics*, vol. 8, no. 6, pp. 625–639, 2021.

[24] W. Zhao, Y. Zhang, and N. Wang, “Soft robotics: Research, challenges, and prospects,” *Journal of Robotics and Mechatronics*, vol. 33, no. 1, pp. 45–68, 2021.

[25] J. Wang and A. Chortos, “Control strategies for soft robot systems,” *Advanced Intelligent Systems*, vol. 4, no. 5, p. 2100165, 2022.

[26] C. Tawk and G. Alici, “Finite element modeling in the design process of 3d printed pneumatic soft actuators and sensors,” *Robotics*, vol. 9, no. 3, p. 52, 2020.

[27] M. Gazzola, L. Dudte, A. McCormick, and L. Mahadevan, “Forward and inverse problems in the mechanics of soft filaments,” *Royal Society open science*, vol. 5, no. 6, p. 171628, 2018.

[28] J. N. Reddy, *Introduction to the finite element method*. McGraw-Hill Education, 2019.

[29] M. Khalid Jawed, X. Huang, and C. Majidi, “A discrete geometric approach to simulation of soft multi-limbed robots,” in *APS March Meeting Abstracts*, vol. 2019, 2019, pp. R56–008.

[30] N. Naughton, J. Sun, A. Tekinalp, T. Parthasarathy, G. Chowdhary, and M. Gazzola, “Elastica: A compliant mechanics environment for soft robotic control,” *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 3389–3396, 2021.

[31] C. Armanini, F. Boyer, A. T. Mathew, C. Duriez, and F. Renda, “Soft robots modeling: A structured overview,” *IEEE Transactions on Robotics*, 2023.

[32] M. C. Yip, J. A. Sganga, and D. B. Camarillo, “Autonomous control of continuum robot manipulators for complex cardiac ablation tasks,” *Journal of Medical Robotics Research*, vol. 2, no. 01, p. 1750002, 2017.

[33] M. Li, R. Kang, D. T. Branson, and J. S. Dai, “Model-free control for continuum robots based on an adaptive kalman filter,” *IEEE/ASME Transactions on Mechatronics*, vol. 23, no. 1, pp. 286–297, 2017.

[34] M. C. Yip and D. B. Camarillo, “Model-less feedback control of continuum manipulators in constrained environments,” *IEEE Transactions on Robotics*, vol. 30, no. 4, pp. 880–889, 2014.

[35] ——, “Model-less hybrid position/force control: a minimalist approach for continuum manipulators in unknown, constrained environments,” *IEEE Robotics and Automation Letters*, vol. 1, no. 2, pp. 844–851, 2016.

[36] M. Giorelli, F. Renda, G. Ferri, and C. Laschi, “A feed forward neural network for solving the inverse kinetics of non-constant curvature soft manipulators driven by cables,” in *Dynamic Systems and Control Conference*, vol. 56147. American Society of Mechanical Engineers, 2013, p. V003T38A001.

[37] T. G. Thuruthel, E. Falotico, M. Cianchetti, F. Renda, and C. Laschi, “Learning global inverse statics solution for a redundant soft robot,” in *Proceedings of the 13th International Conference on Informatics in Control, Automation and Robotics*, vol. 2, 2016, pp. 303–310.

[38] G.-B. Huang, Q.-Y. Zhu, and C.-K. Siew, “Extreme learning machine: theory and applications,” *Neurocomputing*, vol. 70, no. 1-3, pp. 489–501, 2006.

[39] S. Vijayakumar and S. Schaal, “Locally weighted projection regression: An o (n) algorithm for incremental real time learning in high dimensional space,” in *Proceedings of the seventeenth international conference on machine learning (ICML 2000)*, vol. 1. Morgan Kaufmann, 2000, pp. 288–293.

[40] J. Wang, “An intuitive tutorial to gaussian processes regression,” *arXiv preprint arXiv:2009.10862*, 2020.

[41] J. D. Ho, K.-H. Lee, W. L. Tang, K.-M. Hui, K. Althoefer, J. Lam, and K.-W. Kwok, “Localized online learning-based control of a soft redundant manipulator under variable loading,” *Advanced Robotics*, vol. 32, no. 21, pp. 1168–1183, 2018.

[42] G. Fang, X. Wang, K. Wang, K.-H. Lee, J. D. Ho, H.-C. Fu, D. K. C. Fu, and K.-W. Kwok, “Vision-based online learning kinematic control for soft robots using local gaussian process regression,” *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1194–1201, 2019.

[43] B. Subudhi and A. S. Morris, “Soft computing methods applied to the control of a flexible robot manipulator,” *Applied Soft Computing*, vol. 9, no. 1, pp. 149–158, 2009.

[44] Z. Q. Tang, H. L. Heung, K. Y. Tong, and Z. Li, “A novel iterative learning model predictive control method for soft bending actuators,” in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 4004–4010.

[45] ——, “Model-based online learning and adaptive control for a “human-wearable soft robot” integrated system,” *The International Journal of Robotics Research*, vol. 40, no. 1, pp. 256–276, 2021.

[46] R. F. Reinhart and J. J. Steil, “Hybrid mechanical and data-driven modeling improves inverse kinematic control of a soft robot,” *Procedia Technology*, vol. 26, pp. 12–19, 2016.

[47] R. F. Reinhart, Z. Shareef, and J. J. Steil, “Hybrid analytical and data-driven modeling for feed-forward robot control,” *Sensors*, vol. 17, no. 2, p. 311, 2017.

[48] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[49] A. S. Polydoros and L. Nalpantidis, “Survey of model-based reinforcement learning: Applications on robotics,” *Journal of Intelligent & Robotic Systems*, vol. 86, no. 2, pp. 153–173, 2017.

[50] T. G. Thuruthel, E. Falotico, F. Renda, and C. Laschi, “Model-based reinforcement learning for closed-loop dynamic control of soft robotic manipulators,” *IEEE Transactions on Robotics*, vol. 35, no. 1, pp. 124–134, 2018.

[51] X. You, Y. Zhang, X. Chen, X. Liu, Z. Wang, H. Jiang, and X. Chen, “Model-free control for soft manipulators based on reinforcement learning,” in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2017, pp. 2909–2915.

[52] H. Jiang, Z. Wang, Y. Jin, X. Chen, P. Li, Y. Gan, S. Lin, and X. Chen, “Hierarchical control of soft manipulators towards unstructured interactions,” *The International Journal of Robotics Research*, vol. 40, no. 1, pp. 411–434, 2021.

[53] S. Satheeshbabu, N. K. Uppalapati, G. Chowdhary, and G. Krishnan, “Open loop position control of soft continuum arm using deep reinforcement learning,” in *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 2019, pp. 5133–5139.

[54] H. You, E. Bae, Y. Moon, J. Kweon, and J. Choi, “Automatic control of cardiac ablation catheter with deep reinforcement learning method,” *Journal of Mechanical Science and Technology*, vol. 33, pp. 5415–5423, 2019.

[55] Y. Ansari, M. Manti, E. Falotico, M. Cianchetti, and C. Laschi, “Multiobjective optimization for stiffness and position control in a soft robot arm module,” *IEEE Robotics and Automation Letters*, vol. 3, no. 1, pp. 108–115, 2017.

[56] S. Satheeshbabu, N. K. Uppalapati, T. Fu, and G. Krishnan, “Continuous control of a soft continuum arm using deep reinforcement learning,” in *2020 3rd IEEE International Conference on Soft Robotics (RoboSoft)*. IEEE, 2020, pp. 497–503.

[57] X. Liu, R. Gasoto, Z. Jiang, C. Onal, and J. Fu, “Learning to locomote with artificial neural-network and cpg-based control in a soft snake robot,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2020, pp. 7758–7765.

[58] J. Kober, J. A. Bagnell, and J. Peters, “Reinforcement learning in robotics: A survey,” *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013.

[59] D. Silver, “Lectures on reinforcement learning,” URL: https://www.davidsilver.uk/teaching/, 2015.

[60] F. Ayık, “Utilizing deep reinforcement learning in control: Unleashing the power of intelligent systems,” URL: https://medium.com/@ayikfurkan1/utilizing-deep-reinforcement-learning-in-control-unleashing-the-power-of-intelligent-systems-a0 2019.

[61] R. Bellman, “A markovian decision process,” *Journal of mathematics and mechanics*, pp. 679–684, 1957.

[62] “Uc berkeley cs294 lecture slides.” [Online]. Available: https://rail.eecs.berkeley.edu/deeprlcourse-fa17/f17docs/lecture_4_policy_gradient.pdf

[63] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, “Proximal policy optimization algorithms,” *arXiv preprint arXiv:1707.06347*, 2017.

[64] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, “Trust region policy optimization,” in *International conference on machine learning*. PMLR, 2015, pp. 1889–1897.

[65] C. R. Jones, “The 3d dynamics of the cosserat rod as applied to continuum robotics,” Ph.D. dissertation, Mississippi State University, 2011.

[66] D. Trivedi, A. Lotfi, and C. D. Rahn, “Geometrically exact models for soft robotic manipulators,” *IEEE Transactions on Robotics*, vol. 24, no. 4, pp. 773–780, 2008.

[67] H. Lang and J. Linn, *Lagrangian field theory in space-time for geometrically exact Cosserat rods*. ITWM, Fraunhofer Inst. Techno-und Wirtschaftsmathematik, 2009, vol. 3.

[68] H. Lang and M. Arnold, “Numerical aspects in the dynamic simulation of geometrically exact rods,” *Applied Numerical Mathematics*, vol. 62, no. 10, pp. 1411–1427, 2012.

[69] J. Spillmann and M. Teschner, “Corde: Cosserat rod elements for the dynamic simulation of one-dimensional elastic objects,” in *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*, 2007, pp. 63–72.

[70] H. Habibi, C. Yang, I. S. Godage, R. Kang, I. D. Walker, and D. T. Branson III, “A lumped-mass model for large deformation continuum surfaces actuated by continuum robotic arms,” *Journal of mechanisms and robotics*, vol. 12, no. 1, p. 011014, 2020.

[71] J. Wang, X. Zhang, J. Park, I. Park, E. Kilicarslan, Y. Kim, Z. Dou, R. Bashir, and M. Gazzola, “Computationally assisted design and selection of maneuverable biological walking machines,” *Advanced Intelligent Systems*, vol. 3, no. 5, p. 2000237, 2021.

[72] O. Aydin, X. Zhang, S. Nuethong, G. J. Pagan-Diaz, R. Bashir, M. Gazzola, and M. T. A. Saif, “Neuromuscular actuation of biohybrid motile bots,” *Proceedings of the National Academy of Sciences*, vol. 116, no. 40, pp. 19 841–19 847, 2019.

[73] S. Kehrbaum and J. Maddocks, “Elastic rods, rigid bodies, quaternions and the last quadrature,” *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, vol. 355, no. 1732, pp. 2117–2136, 1997.

[74] S. Timoshenko, *History of strength of materials: with a brief account of the history of theory of elasticity and theory of structures*. Courier Corporation, 1983.

[75] M. Bergou, M. Wardetzky, D. Harmon, D. Zorin, and E. Grinspun, “Discrete quadratic curvature energies,” in *ACM SIGGRAPH 2006 Courses*, 2006, pp. 20–29.

[76] L. Verlet, “Computer” experiments” on classical fluids. i. thermodynamical properties of lennard-jones molecules,” *Physical review*, vol. 159, no. 1, p. 98, 1967.

[77] E. H. Skorina, M. Luo, W. Y. Oo, W. Tao, F. Chen, S. Youssefian, N. Rahbar, and C. D. Onal, "Reverse pneumatic artificial muscles (rpams): Modeling, integration, and control," *PloS one*, vol. 13, no. 10, p. e0204637, 2018.

[78] B. Tondu, "Modelling of the mckibben artificial muscle: A review," *Journal of Intelligent Material Systems and Structures*, vol. 23, no. 3, pp. 225–253, 2012.

[79] "Web site of the product line of smooth-on, inc." [Online]. Available: https://www.smooth-on.com/product-line/ecoflex/

[80] M. Hollenstein, "Mechanical characterization of soft materials: Comparison between different experiments on synthetic specimens," *Rapport technique, ETH Zurich*, p. 117, 2005.

[81] J. Vaicekauskaite, P. Mazurek, S. Vudayagiri, and A. L. Skov, "Mapping the mechanical and electrical properties of commercial silicone elastomer formulations for stretchable transducers," *Journal of Materials Chemistry C*, vol. 8, no. 4, pp. 1273–1279, 2020.

[82] "Web site of Elastica Software." [Online]. Available: https://docs.cosseratrods.org/en/latest/guide/workflow.html

[83] M. Russo, S. M. H. Sadati, X. Dong, A. Mohammad, I. D. Walker, C. Bergeles, K. Xu, and D. A. Axinte, "Continuum robots: An overview," *Advanced Intelligent Systems*, vol. 5, no. 5, p. 2200367, 2023.

[84] "Web site of Virtual Amrita Laboratories." [Online]. Available: https://vlab.amrita.edu/?sub=3&brch=175&sim=1080&cnt=1

[85] "Web site of Stable-Baselines3." [Online]. Available: https://stable-baselines3.readthedocs.io/en/master/

[86] "Web site of Open-AI-Gym." [Online]. Available: https://gymnasium.farama.org/

[87] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*. PMLR, 2016, pp. 1928–1937.

[88] S. Huang, A. Kanervisto, A. Raffin, W. Wang, S. Ontañón, and R. F. J. Dossa, "A2c is a special case of ppo," *arXiv preprint arXiv:2205.09123*, 2022.

[89] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[90] L. Bottou, “Large-scale machine learning with stochastic gradient descent,” in *Proceedings of COMPSTAT'2010: 19th International Conference on Computational StatisticsParis France, August 22-27, 2010 Keynote, Invited and Contributed Papers*. Springer, 2010, pp. 177–186.

[91] R. Ge, F. Huang, C. Jin, and Y. Yuan, “Escaping from saddle points—online stochastic gradient for tensor decomposition,” in *Conference on learning theory*. PMLR, 2015, pp. 797–842.

[92] J. L. Sparks, N. A. Vavalle, K. E. Kasting, B. Long, M. L. Tanaka, P. A. Sanger, K. Schnell, and T. A. Conner-Kerr, “Use of silicone materials to simulate tissue biomechanics as related to deep tissue injury,” *Advances in skin & wound care*, vol. 28, no. 2, pp. 59–68, 2015.

Appendix A

Modelling and Simulation in Elastica

In this section, particular details regarding the experiments that were run in order to verify the competence of the obtained model are presented.

A.1 Tests for Simulation Time-Step Tuning

Test 1

By applying $a = 0.01$, time-step becomes $dt = 0.000033\ s$. By simulating the rod's behavior for 10 seconds while having only one actuator producing a constant force of 0.1N (Step Input, Figure 3.7), the results shown in Figure A.1 are produced. The produced simulation is numerically stable. Nevertheless, the physical simulation of 10 seconds requires 35 real seconds. This computational cost is considered high for Model-Free RL standards. Therefore larger time-steps need to be examined.

Test 2

By applying $a = 0.05$, time-step becomes $dt = 0.0001667\ s$. By simulating the rod's behavior for 10 seconds while having only one actuator producing a constant force of 0.1N (Step Input, Figure 3.7), the results shown in Figure A.1 are produced. The 10 second Elastica simulation needed 10 real seconds to be completed while being numerically stable. As it can be seen from Figure A.1, the endpoint presented the same behavior as in test 1 while being less expensive computationally.

Test 3

By applying $a = 0.15$, time-step becomes $dt = 0.0005\ s$. By simulating the rod's behavior for 10 seconds while having only one actuator producing a constant force of 0.1N (Step Input, Figure 3.7), the results shown in Figure A.1 are produced. The 10 second Elastica simulation needed 3 real seconds to be completed while being numerically stable. As it can be seen from Figure A.1, the endpoint presented the same behavior as in tests 1,2 while being the least expensive computationally.

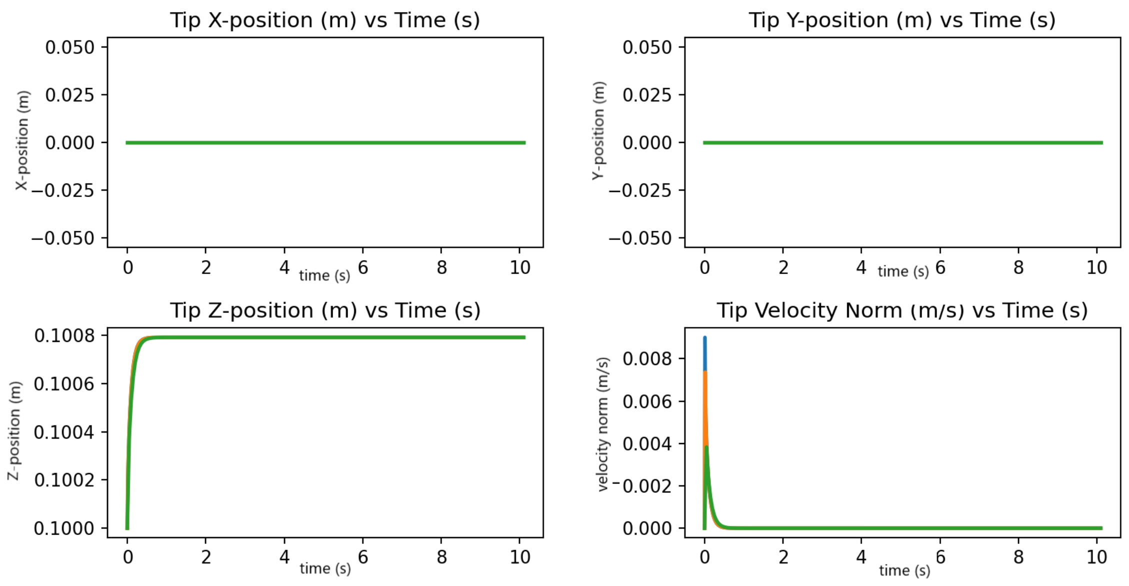


Figure A.1: Rod's Endpoint Position and Velocity norm during tests 1-3. Blue line corresponds to test 1, orange line to test 2 and green line to test 3. The rod's endpoint presents the same behavior in all tests. However magnitude differences exist due to the fact that as time-step becomes larger and data-collection frequency remains the same, data points are getting lost.

Test 4

By applying $a = 0.2$, time-step becomes $dt = 0.00066\ s$. By simulating the rod's behavior for 10 seconds while only one actuator produces a constant force of 0.1N (Step Input, Figure 3.7), the results shown in Figure A.2 are produced.

The chosen time-step leads to instability as the plot in Figure A.2 confirms. Based on the obtained results, parameter a can be between 0.05 and 0.15 (i.e. $0.0001667 \leq dt \leq 0.0005$ for the purposes of the present thesis. Nevertheless, in order to be on the safe side, the a is chosen as 0.05 and thus $dt = 0.0001667$

A.2 Response of the model in small axial stresses

Based on the analysis in chapter 3, the crucial parameters of spatial and time discretization are specified. However, in order to verify that the obtained model is competent, its behavior is examined in the presence of a small normal stress i.e. stress

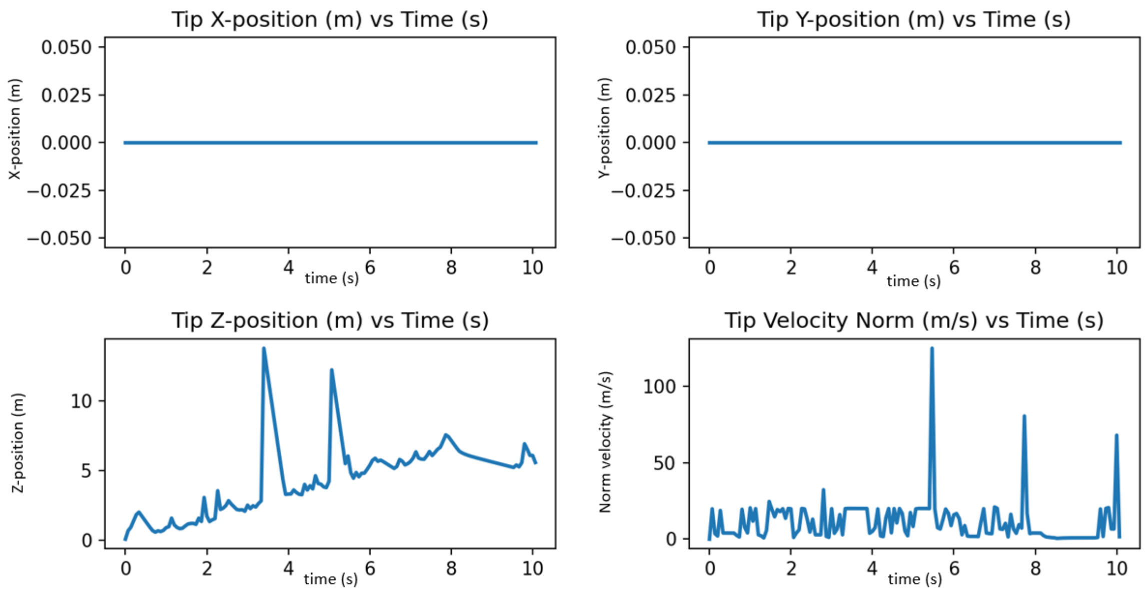


Figure A.2: Rod's Endpoint Position and Velocity norm during test 4

which corresponds to the material's linear stress-strain region. It is known that for most engineering materials linear stress-strain relationship is observed for strains $< 0.1\%$. For Eco-Flex 00-10 this is confirmed by the work presented in [92]. Therefore, assuming that a small normal stress $\sigma = E\epsilon = 0.05\ MPa \cdot 0.0009 = 45\ Pa$ is acting on the free endpoint of the Elastica rod in its initial state and along its axis, a force of $45Pa \cdot \pi(0.009\ m)^2 = 0.01145\ N$ acts on the rod's free endpoint. Hence, a constant force acting on the rod's free endpoint is simulated. The input force can be seen in Figure A.3 and the rod's response in Figure A.4. As it can be observed from Figure A.4, the rod elongated and its length increased by 0.00009 m which corresponds to the expected from theory 0.0009 strain.

A.3 Defining force change per time-step

Large force changes (e.g. step input force) lead to excitation of under-damped natural frequencies which are associated with the stiff deformations and especially elongation. Therefore, in this section, the largest force-change per time-step that does not lead to unstable behavior when the robot elongates, is searched. In the most extreme elongation case (when the force change per time-step is bounded) each actuator would produce equal force per time-step which would increase per time-step by a fixed amount. Since, each actuator should not produce more than 0.6N, during a 10 second simulation (10s/0.0001667s = 59999 simulation steps),the maximum elongation force should be 2.4N. Hence, the largest tolerable force change per time-step is a bit larger than $10^{-5}N$.

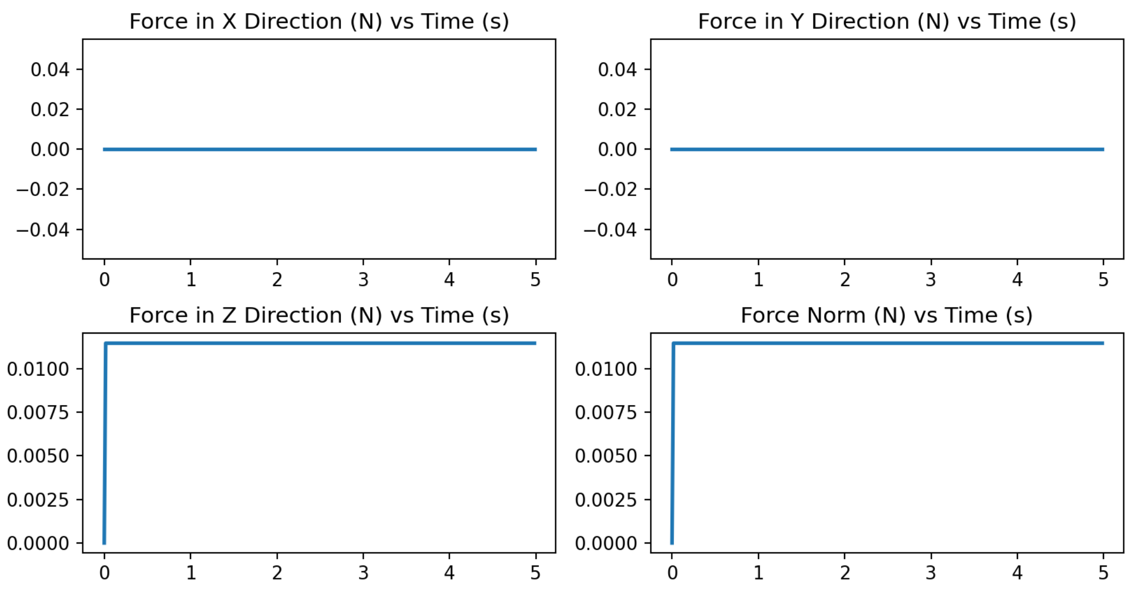


Figure A.3: The Force that simulates the axial stress acting on Elastica rod's free endpoint.

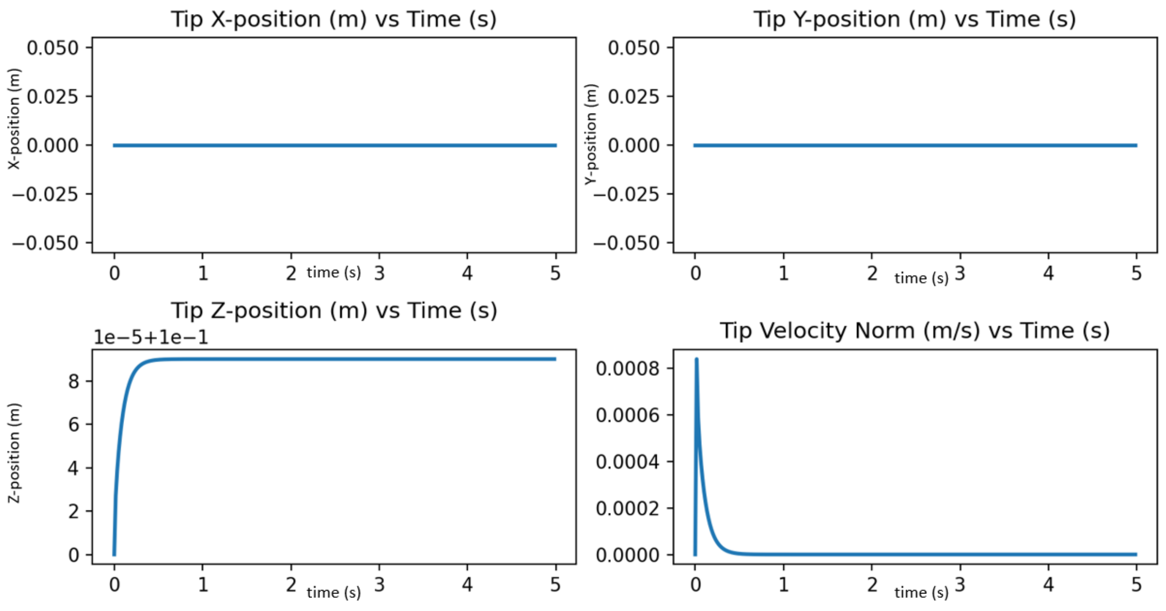


Figure A.4: The response of the Elastica rod to the axial stress. As it can be seen in the bottom left plot, the rod elongated and its endpoint stabilized at 0.10009 m.

Based on the above, three different force changes per time-step are considered: $10^{-5}N$,$10^{-7}N$ and $10^{-9}N$. In each case, the force of each actuator increases with time until t=3.3s. Beyond that time instant, the actuator forces are kept constant. As it can be observed from Figure A.5, force changes $10^{-7}N$ and $10^{-9}N$ are unable to produce acceptable elongation at a reasonable time frame. On the other hand, the force change of $10^{-5}N$ seems a reasonable choice. This force change per time step does not cause numerical issues the system's response can be characterized marginally stable since small under-damped oscillations are present. However these oscillations are much smaller in magnitude compared to the system's step response 3.9. The under-damped oscillations are unavoidable since only one damping coefficient can be specified and we have two deformation modes with very different stiffnesses i.e. Bending and Stretching/Compression. The described system behavior can be observed in Figure A.5. Hence, it is concluded that a force change $df = 10^{-5}N$ is suitable for the scope of the present thesis.

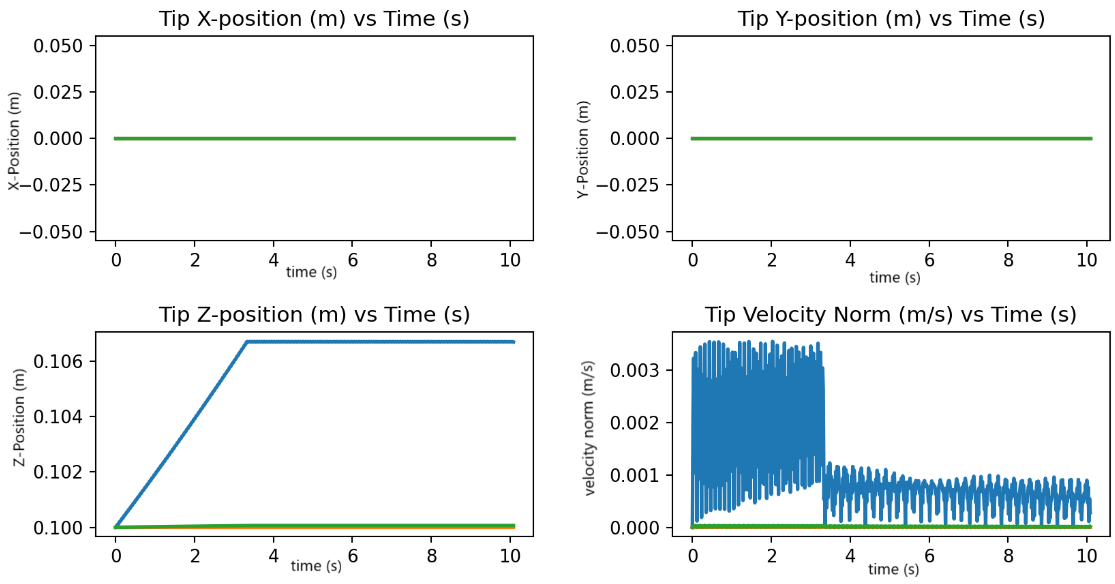


Figure A.5: End-Effector's behavior under $10^{-5}N$ (blue line),$10^{-7}N$ (green line) and $10^{-9}N$ (orange line) force change per time-step.

A.4 Higher Flexibility Test

In this section the model that was used in the present thesis is used again just with higher number of spatial discretization elements i.e 40 and 50. It can be seen from Figures A.6 and A.7 that with higher number of spatial discretization elements the end-effector's workspace is increased. Nevertheless the behavior of the End-Effector remains the same in all cases.

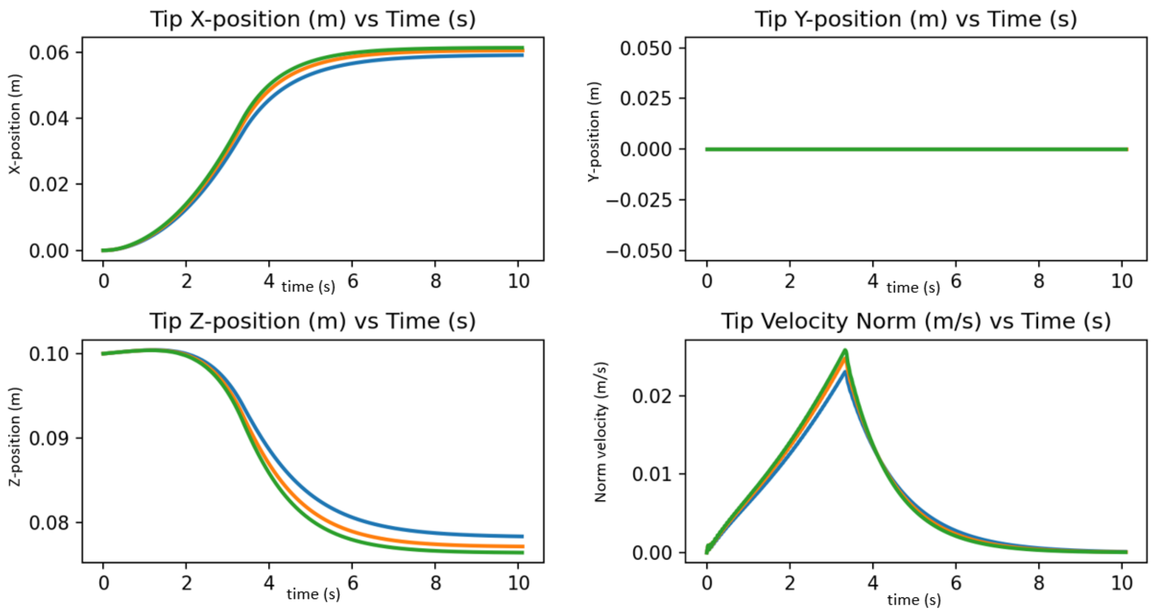


Figure A.6: End-Effector's behavior under the same actuation profile and different number of spatial discretization elements. Blue line corresponds to $n = 30$, orange line to $n = 40$ and green to $n = 50$.

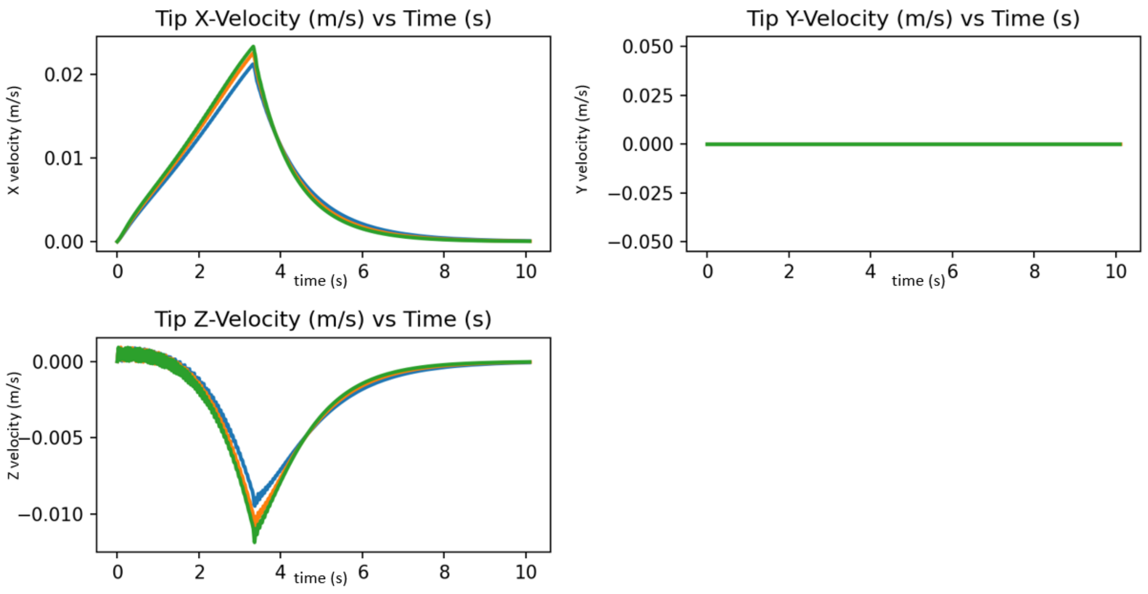


Figure A.7: End-Effector's behavior under the same actuation profile and different number of spatial discretization elements. Blue line corresponds to $n = 30$, orange line to $n = 40$ and green to $n = 50$.