# Pushing with Soft Robotic Arms via Deep Reinforcement Learning

*Carlo Alessi,* Diego Bianchi, Gianni Stano, Matteo Cianchetti, and Egidio Falotico*

**Soft robots can adaptively interact with unstructured environments. However, nonlinear soft material properties challenge modeling and control. Learning-based controllers that leverage efficient mechanical models are promising for solving complex interaction tasks. This article develops a closed-loop pose/force controller for a dexterous soft manipulator enabling dynamic pushing tasks using deep reinforcement learning. Force tests investigate the mechanical properties of a soft robot module, resulting in orthogonal forces of $9-13$ N. Then, the policy is trained in simulation leveraging a dynamic Cosserat rod model of the soft robot. Domain randomization mitigate the sim-to-real gap while careful reward engineering induced pose and force control even without explicit force inputs. Despite the approximate simulation, the sim-to-real transfer achieved an average reaching distance of $34 \pm 14$ mm ($8.1\%L \pm 3.4\%L$), an average orientation error of $0.40 \pm 0.29$ rad ($23^\circ \pm 17^\circ$) and applied pushing forces up to 3 N. Such performance is reasonable for the intended assistive tasks of the manipulator. The experiments uncovered that the soft robot interacting with the environment exhibited torsional and counter-balancing movements. Although not explicitly enforced, they emerged from the mechanical intelligence of the manipulator. The results demonstrate the potential of soft robotic manipulation via reinforcement learning.**

C. Alessi, D. Bianchi, M. Cianchetti, E. Falotico
The BioRobotics Institute
Scuola Superiore Sant'Anna
Viale Rinaldo Piaggio, 34, 56025 Pontedera, PI, Italy
E-mail: carlo.alessi@santannapisa.it

C. Alessi, D. Bianchi, M. Cianchetti, E. Falotico
Department of Excellence in Robotics & AI
Scuola Superiore Sant'Anna
Piazza Martiri della Libertà 33, 56127 Pisa, PI, Italy

G. Stano
Department of Mechanics
Mathematics and Management
Polytechnic University of Bari
Via Edoardo Orabona 4, 70125 Bari, BA, Italy

The ORCID identification number(s) for the author(s) of this article can be found under https://doi.org/10.1002/aisy.202300899.

**DOI: 10.1002/aisy.202300899**

## 1. Introduction

Continuum soft manipulators (CSMs) can potentially excel in adaptive and safe interaction with unstructured environments.[1] However, hyper-redundancy, complex dynamics, and nonlinear soft material properties challenge soft robot modeling and control.[2] Notable strides in modeling techniques[3] paved the way for diverse model-based[4] and model-free control strategies.[5] Nonetheless, the use of CSMs for physical tasks remains an open problem.[6]

Popular methods for deriving forward models for CSMs are *geometrical* models like the piece-wise constant curvature (PCC) approximation.[7] Such models featured in proportional-derivative (PD) controllers to perform real-world tasks with a soft manipulator.[8] Other studies described the shape of a synthetic soft arm with a polynomial curvature model used within an extended PD steady-state regulator.[9] Despite the effectiveness of geometrical models, these assumptions degrade when soft robots are subject to significant external forces typical of interaction tasks.

Other viable modeling and control approaches are inherently *data-driven*.[10] Despite not requiring geometric and physical expertise, they rely on pseudo-random motion data to train artificial neural networks mapping actuation to task space.[11] This robot-independent method effectively derived efficient forward models employed in learned policies for tracking[12,13] and throwing.[14] However, data-driven forward models become inappropriate for interaction tasks due to the challenges of collecting and labeling representative interaction datasets, extensive optimization, and over-fitting.

Promising approaches toward CSMs interacting with unstructured environments are *continuum mechanical* models. These models characterize the deformations of soft robots in physical terms and serve as valuable simulators of physical interactions. The classical finite element methods can accurately represent complex 3D geometries.[15] Such accuracy is paid with high computational costs that complicate the control problem, although recent model order reduction techniques make these methods more affordable.[16,17] Other suitable approaches employ reduced-order mechanical models like Cosserat rods, which effectively describe slender bodies undergoing large deformations, balancing the accurate representation of complex mechanics and computational efficiency.

### 1.1. Article Contribution

In this study, we develop a closed-loop pose/force control architecture enabling CSMs to perform dynamic pushing tasks via deep reinforcement learning (RL).

First, we elaborate a simulated environment that handles the dynamics of the soft robot described by two connected Cosserat rods and the physical interaction with a spherical object (Section 2). Then, we design a learning-based control architecture that enables CSMs to perform pushing tasks, training a policy in simulation using proximal policy optimization (PPO). To mitigate the significant sim-to-real gap in soft robotics, we employ domain randomization (DR), an effective sim-to-real technique in robotics.[18] Herein, we originally applied DR to control a soft robot performing an interaction task in the real world, emphasizing soft material properties. In addition, we design a reward function such that the agent learns to rapidly approach the object with the correct pose and exert pushing forces. This enables pose/force control even without an explicit force term as an input. Finally, the controller is evaluated in simulation and transferred to the real robot (Section 3). The results include a discussion of notable simulated and physical trials that shed some light on domain randomization and the mechanical intelligence of the soft robot. Furthermore, we conduct various force tests for one arm module to investigate the mechanics of the soft robot and its ideal pushing capabilities in controlled setups. The article concludes with a summary of the main results and hints for future works (Section 4).

### 1.2. Related Works

Prior work on manufacturing[19] and modeling[20] of our CSM only characterized its workspace. For control purposes, it was employed for trajectory tracking using a data-driven model,[12] while[21] used the single-section Cosserat rod model to test the generalization of a control policy to different dynamics in simulation. This article further investigates the CSM mechanics through force tests and the sim-to-real transfer of an RL control policy for an interaction task.

Notable model-based position/force controllers were developed based on a kinetostatic model of continuum rigid arms making several kinematic and deformability assumptions;[22] based on a static Cosserat model of a soft arm neglecting oscillations caused by fast and severe contacts;[23] and for a planar soft actuator based on a quasi-static PCC with hysteresis compensation.[24] Our pose/force controller considers the effect of contact forces dynamically.

Concerning learning-based control of CSMs using rod models, Naughton et al.[25] effectively controlled Cosserat rods with RL to adaptively interact with the environment in simulation. Comparing various deep RL algorithms revealed PPO as the most consistent. However, in the real world, control policies trained on rod models were employed for position tracking[26] or quasi-static positioning with interaction limited to static payloads[27] with no exploration of the sim-to-real gap. We sim-to-real transfer a Cosserat-based RL policy for pose/force control in a dynamic physical task.

Promising manipulation tasks for CSMs include minimally invasive surgery,[28] soft robotic showers,[29] precision agriculture,[30] or even space debris collection.[31] Toward these efforts, researchers explored variants of robotic pushing as a proxy for more complex manipulation tasks.[32] Recently, Yang et al.[33] applied RL for pushing objects on a plane using a rigid arm equipped with a soft tactile sensor without DR. Also, Tang et al.[34] proposed a learning-based approach for simultaneous position and force control of a soft arm performing a planar assistive task applying forces up to 0.1 N on a limited workspace. Similarly, an optimal controller based on meta-learning to interact with changing environments was proposed.[35]

In soft robotics, the sim-to-real transfer with DR was adopted for modeling an optical tactile sensor,[36] for vision-based pose estimation of origami-inspired continuum robots,[37] and for tracking circles with an RL policy trained on a PCC model.[38] Here, we randomize a Cosserat rod-based simulation of CSMs, emphasizing the material properties.

## 2. Experimental Section

### 2.1. Physical Environment

The physical environment comprises a two-module pneumatic soft manipulator and a force sensor attached to an articulated arm as shown in **Figure 1**.

The AM I-Support is a 3D-printed soft robotic arm with three elliptical pneumatic chambers that can generate large movements by combining stretching and bending.[19] Two terminals (top and bottom) confine the modules, six rings distributed along the body constrain the chambers, while nuts and bolts assemble the parts. Each module has a cross-section of 30 mm radius, an overall length of ≈202 mm, and weighs about 183 g. The pneumatic chambers are ≈180 mm long, while the top and bottom terminals are ≈20 mm and ≈5 mm long, respectively. The actuators are distributed axially, at a radial distance of $\delta = 20\,\text{mm}$

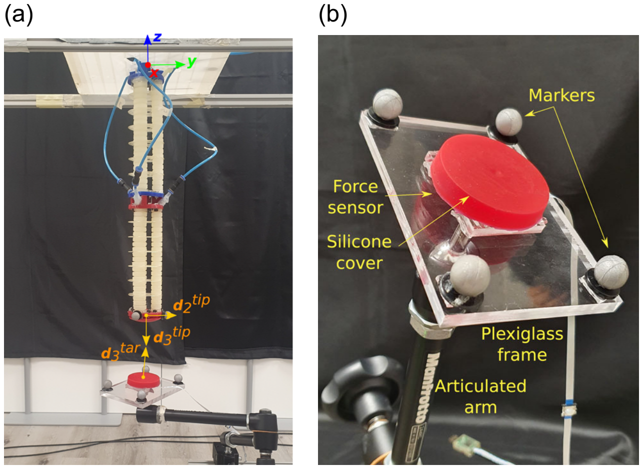

**Figure 1.** Physical environment. a) The soft robotic arm is oriented downward to reduce the effect of gravity. The force sensor attached to an articulated arm is moved in the robot workspace at different positions and orientations. b) A silicone cylinder protects the force sensor plate. The sensor position corresponds to the centroid of the four reflective markers on the supporting Plexiglass plate.

from the cross-section centroid, and equally spaced by 120° around the center.

The arm was fabricated using the soft material thermoplastic polyurethane with 80 Shore A hardness (TPU 80 A LF, by BASF), characterized by 17 MPa tensile strength and elongation at a break of 471%. The Ultimaker S5 (Ultimaker, The Netherlands) machine equipped with a 0.4 mm nozzle and the slicing software Ultimaker Cura 4.11 were used to process TPU 80 A LF and set process parameters. The reduction of voids (intra-, and inter-layer) in 3D printed parts is crucial to ensure the airtightness of the soft robotic arm when pressurized. In accordance with,[39,40] we reduced voids and fabricated leakage-free structures with a low layer thickness of 0.1 mm, an increased extrusion temperature of 240 °C, and an overlap between adjacent beads of 0.1 mm.

To allow complex movements and the coverage of a larger task space, two identical AM I-Support modules are linked by nuts and bolts (see Figure 1a). The robot is controlled by an actuation box composed of six proportional pressure micro-regulators (max pressure 4 bar) and a control unit comprising an Arduino Due, a DAC, and an amplifier to manage the pressure micro-regulators.

A protective soft cap attached to the end-effector allows a safe interaction with the environment by covering the nuts and bolts. The cup reduces stress peaks and the damage that would result from almost punctual contact. The protective cup is a sphere sector with a height of 10 mm and a radius of 40 mm, made from TPU using the Ultimaker S3 3D printer.

A six-axis force sensor (Touchence S50C1-WM155-K1-P6I, 50 N maximum range) measures pushing forces applied by the manipulator at the target pose. The sensor communicates with the computer using I$^2$C (Inter-Integrated Circuit) bus communication. The sensor plate has a radius of about 7 mm. To prevent damage to the force sensor, a disk with a 55 mm diameter made of Dragonskin30 (Smooth-On, Inc.) is placed on its plate (red disk in Figure 1b). The force sensor is mounted on an articulated arm to move it in the robot workspace. The force sensor is also adopted to conduct force tests to explore the mechanical capabilities of the soft arm and provide an upper bound of the maximum applicable force.

Finally, a vision-based motion capture system (VICON Motion Capture Ltd) tracks retroreflective markers attached to the end-effector and the sensor frame.

### 2.2. Simulated Environment

We modeled the physical setup in a computational environment, extending a state-of-the-art implementation of dynamic Cosserat rods.[41] A rod is a mono-dimensional object with a length $l$ much larger than the other dimensions. The shape of a rod is described by the kinematic pose

$$\begin{bmatrix} \boldsymbol{R}(s,t) & \boldsymbol{x}(s,t) \\ 0^{\mathrm{T}} & 1 \end{bmatrix} \tag{1}$$

where $\boldsymbol{x}(s,t) \in \mathrm{R}^3$ is the position vector of the center-line, $s \in [0,l]$ is the arc-length material coordinate and $t \geq 0$ is time. The special orthogonal rotation matrix $\boldsymbol{R}(s,t) = [\boldsymbol{d}_1(s,t), \boldsymbol{d}_2(s,t), \boldsymbol{d}_3(s,t)]$ describes the orientation of the material frame. The normal and binormal directors $\boldsymbol{d}_1 \in \mathrm{R}^3$ and $\boldsymbol{d}_2 \in \mathrm{R}^3$ span the rod cross-section, while $\boldsymbol{d}_3 = \boldsymbol{d}_1 \times \boldsymbol{d}_2$ points along the center-line tangent in absence of shear.

Previous work extended the simulator to consider the pneumatic actuation and modeled a module of the AM I-Support as a Cosserat rod with homogeneous geometrical and material properties.[20] Extensive motion tasks validated the model of the single module. **Table 1** reports the complete list of parameters used in the numerical simulation. The rod-based model accounts for geometrical properties like the module length $l$, the distance between the centroids of the actuators and the

**Table 1.** Parameters used for the numerical simulations and domain randomization in the training process.

| Parameter, Symbol (unit) | Calibrated Value | Domain Randomization Distribution |
|---|---|---|
| Robot Geometry | | |
| Module length, $l$ [m] | 0.190 | $U(0.190, 0.205)$ |
| Actuator-centroid distance, $\delta$ [m] | 0.02 | $U(\delta \pm 1e-3)$ |
| Actuator outer radius, $r_o$ [m] | 0.00779 | $U(r_o \pm 0.5e-3)$ |
| Actuator wall thickness, $t_w$ [m] | 0.0014 | – |
| Robot Material | | |
| Module mass, $m$ [kg] | 0.170 | $U(0.170, 0.185)$ |
| Density, $\rho$ [kg m$^{-3}$] | 1104 | $N(\rho, 0.01\rho)$ |
| Young modulus, $E$ [Pa] | $1.64e6$ | $N(E, 0.01E)$ |
| Poisson ratio, $\nu$ [–] | 0.5 | – |
| Translational damping constant, $\lambda_t$ [s$^{-1}$] | 800 | $N(\lambda_t, 0.01\lambda_t)$ |
| Rotational damping constant, $\lambda_r$ [m$^2$s$^{-1}$] | $1e-4$ | $N(\lambda_r, 0.01\lambda_r)$ |
| Actuator strain gains, $\boldsymbol{\gamma}$ [–] | **1** | $N(\boldsymbol{\gamma}, 0.02)$ |
| Fixed Joint | | |
| Spring constant, $k^{\text{joint}}$ [N m$^{-1}$] | $1e5$ | – |
| Spring dissipation, $\nu^{\text{joint}}$ [N sm$^{-1}$] | 0 | – |
| Rotational stiffness, $k_t^{\text{joint}}$ [Nm] | 10 | – |
| Rotational damping, $\nu_t^{\text{joint}}$ [Nm s rad$^{-1}$] | 0 | – |
| Pressure Transient | | |
| Pressure threshold, $p^0$ [bar] | 0.2 | – |
| Pressure load time, $t_{\text{rise}}$ [s] | 0.2 | – |
| Pressure unload time, $t_{\text{drop}}$ [s] | 0.56 | – |
| Environment | | |
| Gravity, $\boldsymbol{g}$ [m s$^{-2}$] | [0,0, –9.80513] | $N(\boldsymbol{g}, 0.01g)$ |
| Numerics | | |
| Segments per module, $n$ [–] | 20 | – |
| Integration timestep, $dt$ [s] | $2e-4$ | $\{dt, dt/2\}$ |
| Object and Contact | | |
| Object radius, $r^{\text{tar}}$ [m] | – | $U(0.005, 0.035)$ |
| Object density, $\rho^{\text{tar}}$ [kg m$^{-3}$] | 1000 | – |
| Collision stiffness, $k^{\text{contact}}$ [N m$^{-1}$] | – | $U(10, 500)$ |
| Collision dissipation, $\nu^{\text{contact}}$ [N sm$^{-1}$] | 0.1 | – |
| Velocity damping coefficient, $\nu^{\text{slip}}$ [N sm$^{-1}$] | $1e5$ | – |
| Friction coefficient, $\mu$ [–] | 0.5 | – |

cross-section $\delta$, the approximated outer radius of the actuators $r_o$, and the wall thickness of the 3D-printed material $t_w = 1.4\,\text{mm}$. Similarly, material properties include the module mass $m$, the material density $\rho$, and the elastic modulus $E$ fitted from experimental stretching data, a Poisson ratio of $\nu = 0.5$ assuming incompressibility, and optimized damping constants for forces $\lambda_t$ and torques $\lambda_r$. Moreover, *strain gains* $\boldsymbol{\gamma} \in \mathbb{R}^6$ capture the manufacturing variability and material degradation by tuning the pressure–strain relation $\varepsilon(p)$ for each actuator.[20] The model also approximates the pressure transient $p(t)$ of the pneumatic actuation system through a polynomial curve parameterized by minimum pressure threshold $p^0 = 0.2\,\text{bar}$, load time $t_{\text{rise}} = 0.2\,\text{s}$, and unload time $t_{\text{drop}} = 0.56\,\text{s}$.

To achieve pushing tasks with a dexterous CSM, we further extend the simulated environment by modeling the two-module pneumatic CSM and the interaction with the object (see **Figure 2**). In particular, a fixed joint connects two identical rods sequentially by restricting the relative motion and rotation between rod nodes and elements by applying restoring forces and torques.[42] The joint is modeled as a stiff spring with constant translational stiffness $k_t^{\text{joint}} = 1e5\,\text{N}\,\text{m}^{-1}$, translational damping $\nu_t^{\text{joint}} = 0\,\text{N}\,\text{sm}^{-1}$, rotational stiffness $k_r^{\text{joint}} = 10\,\text{Nm}$, and rotational damping $\nu_r^{\text{joint}} = 0\,\text{Nm}\,\text{s}\,\text{rad}^{-1}$, found empirically (see Table 1).

The end-effector of the simulated soft robotic arm interacts with and pushes objects modeled as spheres of radius $r^{\text{tar}}$ and density $\rho^{\text{tar}}$. We approximated the contact between the tip and the object through repulsive forces of springs uniformly distributed on the contact surface with stiffness $k^{\text{contact}}$ and constant damping $\nu^{\text{contact}} = 0.1\,\text{N}\,\text{sm}^{-1}$. In addition to the contact forces, we employed friction forces combining Coulomb friction with coefficient $\mu = 0.5$ and a damper opposing the tangential slip velocity with coefficient $\nu^{\text{slip}} = 1e5\,\text{N}\,\text{sm}^{-1}$ as detailed in.[43]

Finally, we achieved a good tradeoff between computing time, accuracy, and numerical stability by discretizing each rod into $n = 20$ segments of equal length and integrating the dynamics with time-step $dt = 2e - 4\,\text{s}$. Both rods were subject to gravitational forces $\mathbf{g}$. This simulated environment was used to train the controller.

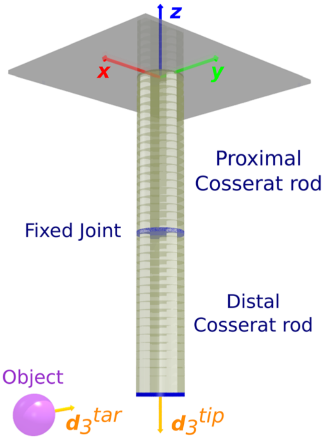

**Figure 2.** Rendering of the simulation environment including a Cosserat rod model of the soft robot and a spherical object.

### 2.3. Control Architecture for Pushing

To address the pushing task using the soft robotic arm, we designed a closed-loop dynamic controller that considers the pose of the end-effector, the object pose, and the object dimension. **Figure 3** shows an overview of the control scheme.

The controller consists of a feed-forward neural network with two hidden layers, each with 512 neurons with *tanh* activation function. The output layer has a linear activation function. The neural network takes as input a vector $\boldsymbol{x}_t \in \mathbb{R}^{28}$ composed as

$$\boldsymbol{x}_t = \left[\boldsymbol{d}_t, \boldsymbol{d}_{t-1}, \boldsymbol{x}_t^{\text{tip}}, \boldsymbol{x}_{t-1}^{\text{tip}}, \boldsymbol{x}_{t-2}^{\text{tip}}, \boldsymbol{q}^{\text{tar}}, \boldsymbol{q}_t^{\text{tip}}, \boldsymbol{q}_{t-1}^{\text{tip}}, r^{\text{tar}}\right] \qquad (2)$$

where $\boldsymbol{x}_t^{\text{tip}} \in \mathbb{R}^3$ is the current end-effector position while $\boldsymbol{q}_t^{\text{tip}} \in \mathbb{R}^4$ is the current end-effector orientation expressed as a unit quaternion. The vector $\boldsymbol{d}_t = \boldsymbol{x}^{\text{tar}} - \boldsymbol{x}_t^{\text{tip}} \in \mathbb{R}^3$ is the current distance between the end-effector and the object surface $\boldsymbol{x}^{\text{tar}}$, while $\boldsymbol{q}^{\text{tar}}$ is the object pose and $r^{\text{tar}}$ expresses the object dimension (e.g., the sphere radius). Observe that for the tracking error $\boldsymbol{d}$, the end-effector position $\boldsymbol{x}^{\text{tip}}$ and orientation $\boldsymbol{q}^{\text{tip}}$ we include previous observations. This inclusion enables the network to capture the rate of change of these quantities and potentially infer their velocity and acceleration. Although the controller does not receive an explicit force term, the designed reward enables an implicit force control to achieve pushing tasks with CSMs.

The controller outputs differential pressures $\Delta\boldsymbol{p} \in \mathbb{R}^6$ for the six pneumatic actuators, with $\Delta p_i \in \{0.0, \pm 0.1, \pm 0.2\}$ bar. The control policy operates at a frequency of 10 Hz.

### 2.4. Reinforcement Learning for Pushing

We derive a controller for pushing tasks with CSMs using deep reinforcement learning (RL). Generally, an RL agent receives at each time step $t$ an observation $o_t$ from the environment (a partial view of the complete environment state $s_t$). The agent acts according to a policy $\pi$ mapping states/observations to actions $a_t$, which can be deterministic or stochastic. The agent receives a scalar reward signal $r(s, a)$ reflecting the level of task performance. Here, the agent is the controller implementing a feed-forward neural network. The RL environment comprises the two-section soft robotic arm and an object, simulated with two sequential Cosserat rods and a sphere. Therefore, the agent receives continuous observations $o_t = \boldsymbol{x}_t$ and outputs discrete actions $a_t = \Delta\boldsymbol{p}_t$. Observation and action spaces are min-max scaled in the interval $[-1, 1]$ to improve the numerical stability of the training process.

#### *2.4.1. Proximal Policy Optimization*

We trained the controller using proximal policy optimization (PPO), a policy-gradient method appropriate for continuous control tasks.[44] The algorithm jointly optimizes a stochastic policy $\pi(a|s)$ and a value-function approximator alternating between two phases: 1) sampling data with the policy through episodic interactions with the environment, and 2) optimizing the policy on the sampled data using stochastic gradient ascent. The objective function penalizes aggressive policy updates by clipping the probability ratio of the current and old policy. We selected PPO

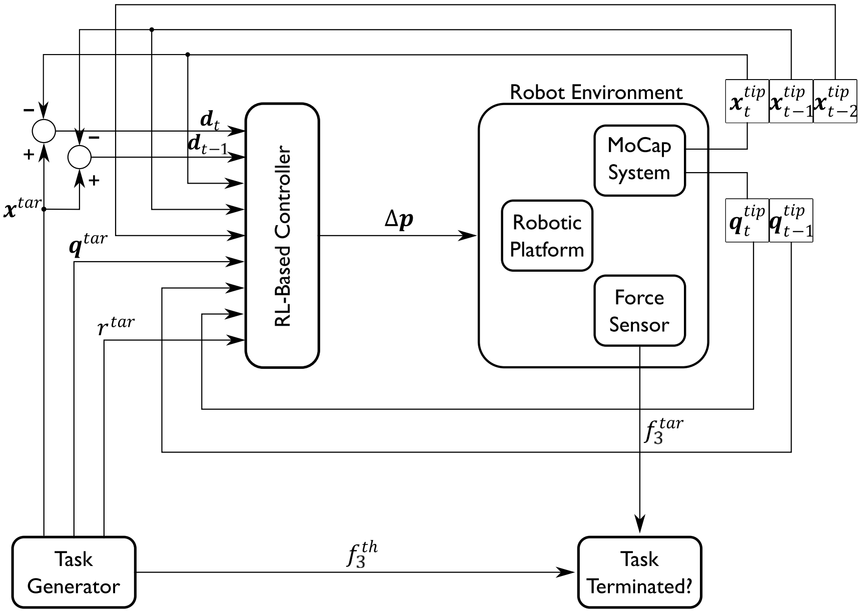

**Figure 3.** Control scheme for pushing tasks. An arbitrary task generator provides target positions $\boldsymbol{x}^{\text{tar}}$ and orientations $\boldsymbol{q}^{\text{tar}}$ of a spherical object of radius $r^{\text{tar}}$ with required pushing force $f_3^{\text{th}}$. The robot environment comprising the AM I-Support, a force sensor, and a motion capture system provides the complete end-effector pose (i.e., $\boldsymbol{x}^{\text{tip}}$ and $\boldsymbol{q}^{\text{tip}}$) at various time steps $t$, and the pushing force applied to the object $f_3^{\text{tar}}$. The reinforcement learning policy combines the observations to control the soft robot through differential pressures $\Delta \boldsymbol{p}$. The task terminates for time limits or for surpassing the required pushing force.

because it often outperforms other online policy gradient methods, striking a good tradeoff between sample complexity, simplicity in hyperparameter tuning, and computing time.[44] We adopt the reliable implementation provided by.[45]

### 2.4.2. Reward Engineering for Pose/Force Control

We define a reward function for pushing with a soft manipulator considering position, orientation, and applied force

$$r_t = \begin{cases} -10 & \text{if NaN} \\ r_{\text{approach}} + r_{\text{push}} + r_{\text{click}} & \text{otherwise} \end{cases} \tag{3}$$

The penalty term of −10 discourages actions that would cause numerical instabilities.[25]

A combined approach term induces the agent to reach the position of the object with a correct orientation:

$$r_{\text{approach}} = 5.0 \cdot r_{\text{dist}} + 0.02 \cdot r_{\text{ori}} \tag{4}$$

where $r_{\text{dist}} = -||\boldsymbol{d}||^2$ rewards proportional to the distance between the end-effector and the target, while the orientation term $r_{\text{ori}} = -\boldsymbol{d}_3^{\text{tip}} \cdot \boldsymbol{d}_3^{\text{tar}} - 1$ is a linear function of the cosine formed by the tangent directors of the end-effector and the object. The reward for approaching the target is always nonpositive and contributes only when the end-effector is not touching the sphere.

Conversely, when there is contact, the agent receives a positive reward proportional to the pushing force:

$$r_{\text{push}} = \min\{(-\boldsymbol{f}^{\text{tar}} \cdot \boldsymbol{d}_3^{\text{tar}})/f_3^{\text{th}}\ ,\ 1\} \tag{5}$$

where $\boldsymbol{f}^{\text{tar}} \in \mathrm{R}^3$ is the force vector exerted by the robot on the sphere. We project the force vector on the object director $\boldsymbol{d}_3^{\text{tar}}$, then normalize the quantity dividing by $f_3^{\text{th}} \in \mathrm{R}$, the force threshold needed to terminate the task.

Finally, the agent receives a positive bonus reward $r_{\text{click}}$ for quickly terminating the pushing task

$$r_{\text{click}} = 0.1 + \exp(-t_f/T) \tag{6}$$

where $t_f$ is the final episode time and $T$ is the time limit. In summary, closed-loop pose control is attained through explicit control inputs (2) and rewards (4). In addition, the combination of (5) and (6) provides an implicit mean to enable force control with CSMs.

## 2.5. Training Process

### 2.5.1. Task Generation

The objective of the reinforcement learning agent is to make the soft arm approach a position $\boldsymbol{x}^{\text{tar}}$ with a desired orientation $\boldsymbol{q}^{\text{tar}}$ and apply a force to push an object along the orthogonal director $\boldsymbol{d}_3^{\text{tar}}$. Each episode starts with the soft robotic arm unactuated and in resting position, $\boldsymbol{x}_0^{\text{tip}} = [0, 0, -L]$, and the sphere oriented and fixed in space as shown in Figure 2. The training episodes terminated in case of numerical errors, if the task reached the time limit of $T = 7$ s, or when the magnitude of the pushing force surpassed a random threshold $f_3^{\text{th}}$.

For each episode, an arbitrary task generator produced a target position $\boldsymbol{x}^{\text{tar}}$ and orientation $\boldsymbol{q}^{\text{tar}}$. First, it sampled the object positions from the lower hemisphere of a unit sphere, virtually centered at the initial end-effector position. Then, the 3D coordinates were scaled on each axis by a random absolute value between 100 and 200 mm. Also, the object positions were shifted randomly by a maximum of ±10 mm in each coordinate. The baseline object orientation $\boldsymbol{d}_3^{\mathrm{tar}}$ was given by the inward normal of the unit sphere perturbed randomly by a rotation around an axis with an angle between 0 and 7 degrees. Finally, the complete object pose was obtained by appropriately sampling the normal-binormal object directors. Moreover, we randomized the object's physical properties (e.g., dimension and stiffness) to set a force threshold $f_3^{\mathrm{tar}}$. The process ensured that the controller visited different poses in a large workspace (about 400 × 400 × 150 mm) and experienced a wide range of forces.

### 2.5.2. Model Selection and Learning

We trained a stochastic policy using PPO to solicit exploration in the environment. Following an empirical model selection and fine-tuning of hyper-parameters, the learning lasted 5.0 million time steps, ≈120 k episodes, equivalent to about 1400 h of simulated learning experience. Due to hardware constraints, the training episodes were collected by $N = 8$ parallel agents interacting with the environment for $M = 64$ time steps per policy update. The policy weights were updated at each iteration based on the collected $N \cdot M$ samples using stochastic gradient ascent for ten epochs, with four mini-batches and $3e-4$ learning rate. The training took place in simulation for about 88 wallclock hours on a standard PC (Intel i7-7700k CPU @ 4.20 GHz × 8, 32 GB RAM). However, the obtained policy allowed real-time control thanks to high inference frequency (about 1000 Hz).

As shown in **Figure 4**, the trend of the exponential moving average of the rewards collected during the training episodes is increasing and converging. Similarly, the episode duration tends to decrease thanks to the reward incentive (6) to complete the task quickly.

## 2.6. Domain Randomization and Noise

To cope with the significant gap between simulation and reality in soft robotics, improve the generalization capabilities, and aid the sim-to-real transfer of the control policy, we employ domain randomization.[46] In each episode, we randomized thirteen simulation parameters about the robot morphology, the object, the environment, and numerics (see Table 1). Moreover, observation and actuation noise were applied at every control step.

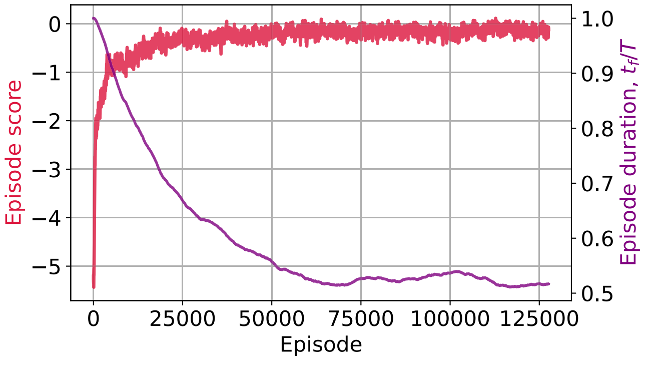

**Figure 4.** Learning trends: rewards and duration of the training episodes (exponential moving averages). The agent learned to terminate the pushing task rapidly.

### 2.6.1. Geometry Randomization

Geometrical properties significantly influence the kinematics of soft robots. For instance, the length $l$ of each arm module was arbitrarily sampled uniformly from an interval exceeding the design length of 202 mm. This approach addresses the tendency of the physical robot to remain partially stretched after prolonged usage. The Cosserat model used does not consider this phenomenon. Therefore, concerning the sim-to-real gap, the control policy would benefit from experience operating with arms of slightly different lengths. Moreover, randomizing the distance between the cross-section centroid and the actuator $\delta$ allows us to cope with offsets of actuator positioning on the cross-section resulting from manufacturing variabilities[47] or the interaction with the environment. Finally, the training process sampled the outer radius of the elliptical pneumatic actuators from a uniform distribution in the interval $[r_o - 0.5, r_o + 0.5]$ mm to consider the uneven geometry of the actuator cross-section along the length of the arm.[19]

### 2.6.2. Material Randomization

Materials are one of the most important contributors to the dynamics of soft robots. For instance, previous work has shown how changes in material properties affected a control policy in tracking tasks.[21] We sampled the material density $\rho$, the elastic modulus $E$, the coefficients of translational $\lambda_t$ and rotational $\lambda_r$ damping from normal distributions centered at calibrated values and with 1% of standard deviation. The *strain gains* of the actuators $\boldsymbol{\gamma}$ were randomized using a wider normal distribution with a 2% standard deviation. The parameter distribution addressed the gap between the physical and simulated robot due to the irregularities and degradation of the 3D-printed actuators as studied in Alessi et al.[20] Finally, the mass $m$ of the module is sampled uniformly from an interval that loosely includes the calibrated value to counter model approximations and unmodelled effects.

### 2.6.3. Object and Contact Randomization

Varying the parameters of the object and contact is fundamental for addressing the sim-to-real gap in interaction tasks. Hence, we uniformly sampled the sphere radius $r^{\mathrm{tar}}$ in the interval $[5, 35]$ mm and the contact stiffness $k^{\mathrm{contact}}$ between 10 and 500 N m$^{-1}$.

### 2.6.4. Environment and Numerical Randomization

To prepare the controller to adapt to unknown environments and interactions, we changed the direction and strength of the gravitational acceleration $\tilde{\boldsymbol{g}}$ acting on the robot. The three vector components were sampled from a normal distribution with mean $\boldsymbol{g}$ and standard deviation 1%g. This method includes uniformly distributed forces $\boldsymbol{f}(s) = dm \cdot (\tilde{\boldsymbol{g}} - \boldsymbol{g})$ that perturb the system efficiently, where $dm = m/n$ is the elemental mass.

In addition, we sampled the integration timestep $dt$ from the set $\{1e-4, 2e-4\}$ s with equal probability to let the agent experience different simulation timescales. This choice also maintained the simulation stable.

#### 2.6.5. Observation and Actuation Noise

Noise was applied to observations and actions at each control step to deal with the inherent uncertainty of sensors and actuators. Specifically, an additive multivariate Gaussian noise with 0 mean and 1 mm of standard deviation were applied to the current distance vector $\boldsymbol{d}_t$ and tip position $\boldsymbol{x}_t^{\text{tip}}$ to mitigate the uncertainties of the vision system. Moreover, we perturbed the object orientation $\boldsymbol{q}^{\text{tar}}$ and the current tip orientation $\boldsymbol{q}_t^{\text{tip}}$ through a rotation of 3 degrees around a random axis.

Finally, we added noise $p_{\text{noise}} \approx N(0, 0.0175)$ bar to the differential actuation $\Delta\boldsymbol{p}$. This process not only captured the variability of the pneumatic system but also helped the agent explore the state-action space during training.

## 3. Results and Discussion

After training the reinforcement learning policy, we tested the controller in simulation and on the physical robot. We report results about three metrics that define the pushing task. First, the distance between the end-effector and target $d = ||\boldsymbol{d}||$ quantifies the reaching quality. The distance is reported in mm and normalized relative to the total manipulator length $L = 2l$ plus the length of the soft cap. Then, the angular error $\theta = |\pi - \arccos(\boldsymbol{d}_3^{\text{tip}} \cdot \boldsymbol{d}_3^{\text{tar}})|$ between the end-effector and the target tangent directors, where $\pi$ is the optimal angle. An angular error $\theta < \pi/2$ is sufficient to enable pushing. Finally, we compute the pushing force $f_3^{\text{tar}} = -\boldsymbol{f}^{\text{tar}} \cdot \boldsymbol{d}_3^{\text{tar}}$ as the force projected on the normal director. We discuss notable simulated and experimental trials, which may offer insights into domain randomization and sim-to-real gap.

### 3.1. Force Tests for the AM I-Support

We conducted six force tests for a single module to assess the mechanical capabilities of the 3D-printed pneumatic soft arm. We clamped the module at one end oriented downward at various vertical distances $\Delta z$ from the force sensor (see **Figure 5**). To better mimic the real pushing scenario, the robot operated without the support of a central backbone. For each distance, the robot stretched by actuating the three chambers equally with incremental steps of $\Delta p = 0.5$ bar. The pressure incremented until the robot exhibited buckling instability to safeguard the soft actuators (see Figure 5a). We repeated each experiment for five trials and recorded the orthogonal forces $f_3^{\text{tar}}$.

As shown in Figure 5b, the average maximum forces measured along the orthogonal of the sensor plate were between 9 and 13 N. The force tests provided an upper bound of the applicable forces. Indeed, the robot cannot attain such maximum forces for tilted poses and for the mechanical complexity of the additional robot module. The buckling instabilities resulted from the different pressure–strain performance of the pneumatic actuators. Although a supporting backbone would allow more precise force measurements and derivation of parametric force–pressure curves, the results would deviate further than those expected in uncontrolled settings.

### 3.2. Control with Domain Randomization

The control policy was evaluated in simulation for experiments generated with domain randomization. For example, **Figure 6** shows the soft robot successfully approaching the object and applying a pushing force with precise orientation. Here, the policy smoothly reduced the initial pose error $d_0 = 125$ mm and $\theta_0 = 1.3$ rad to $d = 4$ mm and $\theta = 0.02$ rad.

**Figure 7**a reports overall statistics for the simulated tasks. The controller achieved an average reaching distance $d$ of $17 \pm 11$ mm (i.e., $4.53\%L \pm 2.95\%L$). This value is about half the 30 mm end-effector radius, which allows the robot to touch the object and exert forces. The distance distribution spreads between a minimum of 0.5 mm and a maximum of 89 mm. The latter value corresponds to a notable trial where domain

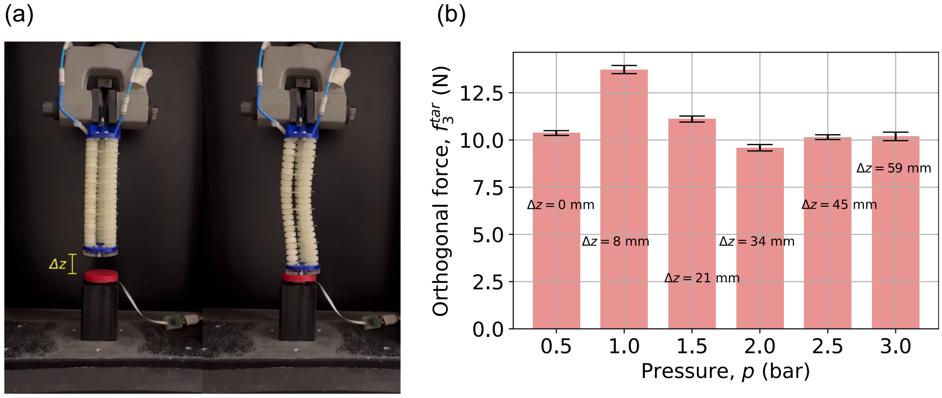

**Figure 5.** Force tests for the AM I-Support. a) Soft robot with one end fixed and one free end at distance $\Delta z$ from the sensor. The arm exhibits buckling instability. b) Mean and standard deviation of the maximum forces applied.

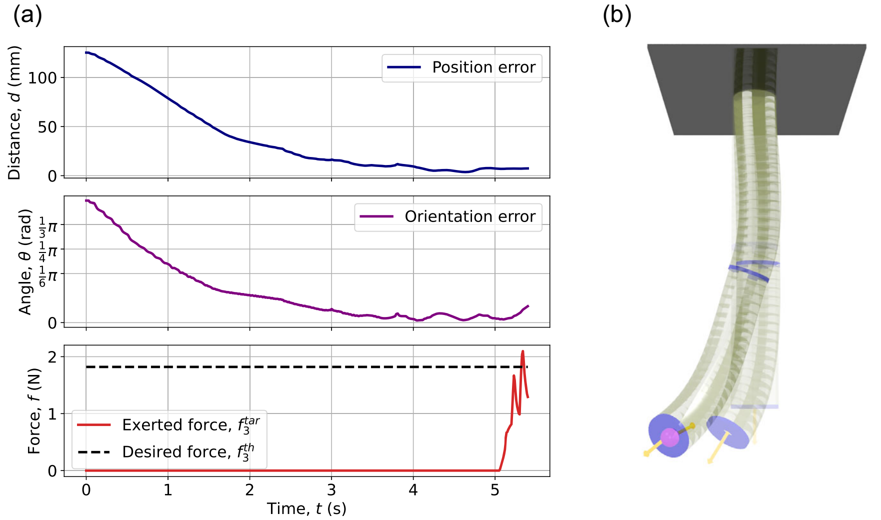

**Figure 6.** Control policy in simulation. a) The soft robot pushes the object at distance $\boldsymbol{d}_0 = [80,\ -91,\ -33]$ mm and relative angle $\theta_0 = 1.3$ rad (75°) with force $f_3^{\text{tar}} = 2.1$ N. b) The simulated soft manipulator pushing in 3D space.

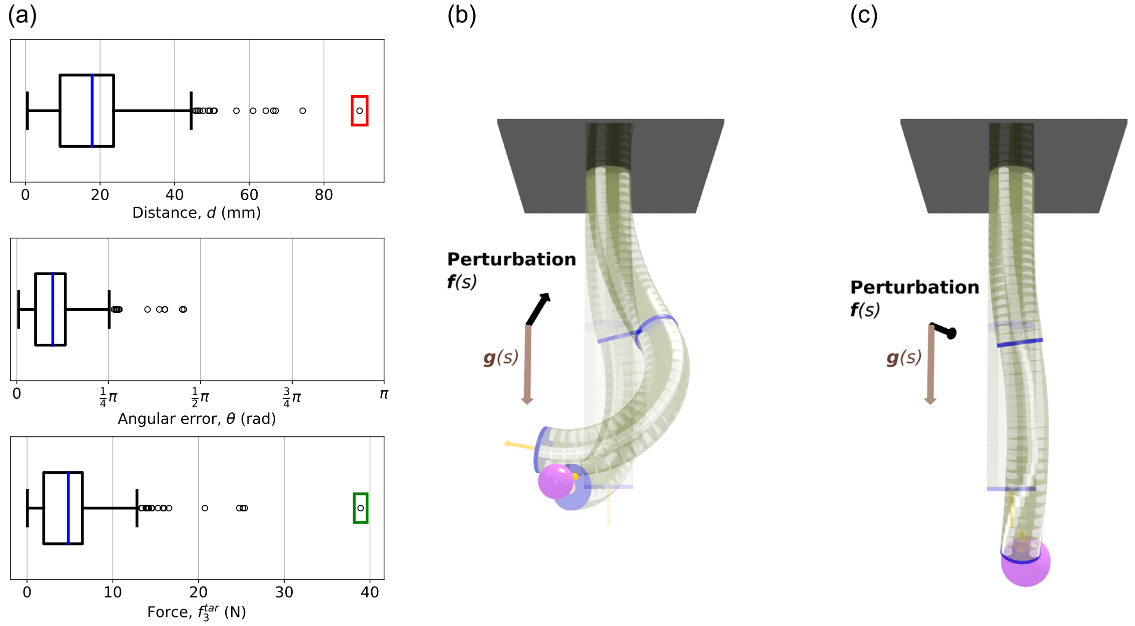

**Figure 7.** Control in simulation with domain randomization. a) Box plots of the distribution of reaching distance $d$, angular error $\theta$ between the end-effector and the object orientation, and the pushing force $f_3^{\text{tar}}$ applied (blue line shows the mean). b) Experiment with high position error due to strong perturbation (red). c) Experiment with high pushing force thanks to high contact stiffness and concord perturbation (green).

randomization sampled a challenging combination of simulation parameters (see Figure 7b). Indeed, we inspected the experiment and uncovered a target in a distant position $\boldsymbol{x}^{\text{tar}} = [172, -45,\ -393]$ mm. Whatsmore, the robot was subject to a distributed acceleration $\tilde{\boldsymbol{g}} = [-0.034,\ 0.131,\ -9.567]\ \text{m s}^{-1}$, which caused a net perturbation in direction $[-0.034,\ 0.131,\ 0.239]\ \text{m s}^{-1}$. The perturbation has relatively strong components in the opposite direction of the task. This limited the robot's dexterity and workspace.

Concerning orientation, the angular error $\theta$ corresponding to the position with minimum distance attained a mean of $0.3 \pm 0.2$ rad (17° ± 12°). The orientation error spanned the interval $[0.02, 1.43]$ rad (i.e., between 1° and 81°). These angular errors are below the threshold $\pi/2$. Therefore, they enable the robot to apply a force component along the orthogonal director of the target, $\boldsymbol{d}_3^{\text{tar}}$, which is functional to the task.

Finally, the policy controlled the end-effector to push objects with an average force $f_3^{\text{tar}}$ of $4.81 \pm 4.18$ N, spanning the interval

[0.04, 38] N. A few maximal forces are beyond the expected upper bounds of 9–13 N given by the force tests. Here, high forces were allowed by a relatively central position of the object coupled with a perturbation that favored the task and the absence of buckling instability (see Figure 7c). A sphere of radius $r^{\text{tar}} = 32$ mm with high contact stiffness $k = 426\ \text{N m}^{-1}$ also contributed to high simulated forces. It is worth observing the torsion exhibited by the distal module due to the interaction with the object.

### 3.3. Sim-to-Real Transfer of Pushing Policy

After the validation in simulation and discussing a few domain randomization examples, we transferred the control policy to the physical environment without further training (i.e., zero-shot sim-to-real transfer). The small neural network size allowed real-time control. We generated real-world pushing tasks with various poses in the robot workspace by moving the articulated arm supporting the force sensor.

**Figure 8** showcases a notable sim-to-real transfer in a challenging trial. The policy controls the robot to approach the object and push along the target orientation. The initial reaching distance $d = 148$ mm decreases to $d = 20$ mm. Upon contact between the end-effector and the object, the robot adapts to the interaction and achieves a perfect orientation of $\theta = 0.0$ rad. Then, the pushing force $f_3^{\text{tar}}$ increases and finally surpasses the threshold $f_3^{\text{th}}$ in less than 5 s (see Figure 8a). Surprisingly, the soft arm during the pushing tasks exhibited interesting mechanical behaviors like non-actuated arm torsion and the counter-balancing of the proximal part as a result of the interaction with the environment (see Figure 8b and Supporting Information). These behaviors were not explicitly programmed or imposed in the reward function but naturally emerged from the mechanical intelligence of the soft robot coupled with the rich sensory information of the control policy. We show several trials in the Supporting Information video. It is worth noting that the simulated soft arm did not exhibit buckling instability like the proximal part of the physical arm. The discrepancy could be due to using a single Cosserat rod to represent the entire robot module with three pneumatic actuators. Perhaps modeling each actuator with a dedicated rod would have increased the simulation fidelity. However, this would have significantly increased the computational cost. Thanks to the approximate robot model developed, we could train a control policy efficiently.

Regarding statistics, **Figure 9**a reports an overview of the physical experiments. In particular, the control policy achieved a satisfactory average reaching distance $d$ of 34 ± 14 mm (the equivalent of $8.1\%L \pm 3.4\%L$). The reaching distance spans the interval between 12 and 90 mm. The trial with the maximum distance value corresponds to a target with a challenging pose (shown in Figure 9b). Despite a high position error for this outlier, the policy achieved an accurate orientation with $\theta = 0.14$ rad (8°), which would have allowed the robot to push.

Similarly, the control policy attained a mean absolute angular error $\theta$ of 0.40 ± 0.29 rad (23° ± 17°), far below the maximum orientation error of $\pi/2$ needed to exert forces. Also, the spread of the angular error measured with an interquartile range of 0.26 rad was low. The value of $\theta$ ranged between a minimum of 0.05 (3°) rad and a maximum of 2.29 rad (131°). The trial related to this outlier is visualized in Figure 9c. While the end-effector achieved a poor orientation, the reaching distance $d = 41$ mm was acceptable.

Concerning the pushing forces, the force sensor registered loads $f_3^{\text{tar}}$ in the axis orthogonal to the sensor plate in the range [0.46, 3.08] N with mean 0.96 ± 0.52 N. This force range is reasonable for the assistive purposes of the soft arm in interaction tasks.[29]

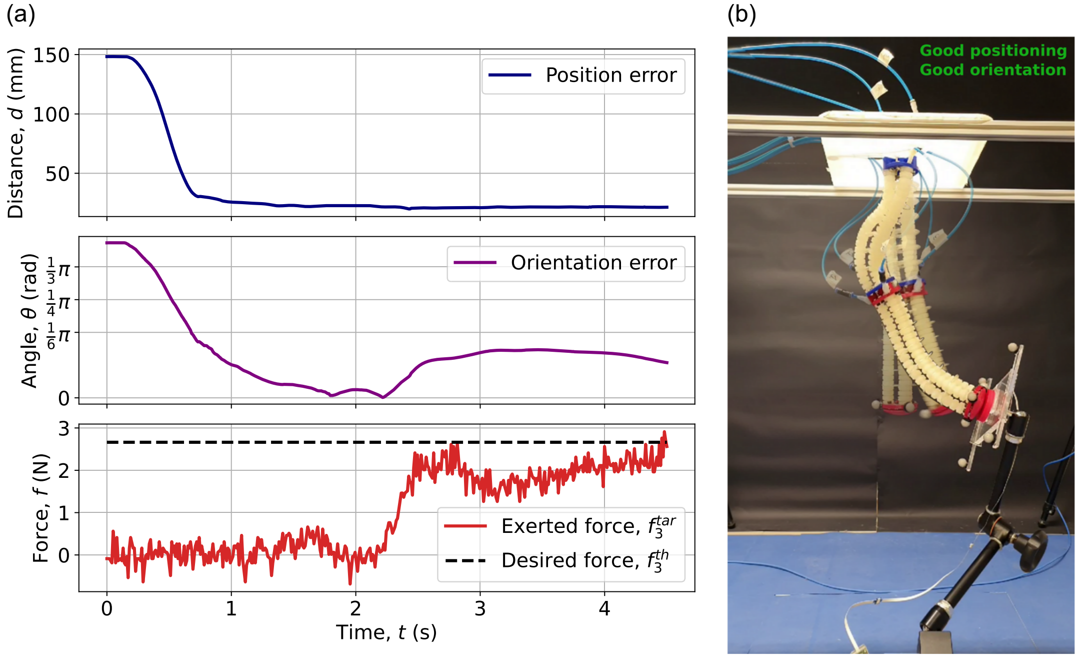

**Figure 8.** Sim-to-real transfer. a) The soft robot rapidly pushes the object displaced at distance $\boldsymbol{d}_0 = [20, -147, 10]$ mm and relative angle $\theta_0 = 1.24$ rad (71°) with force $f_3^{\text{tar}} = 2.91$ N. b) The soft manipulator pushing in 3D space.

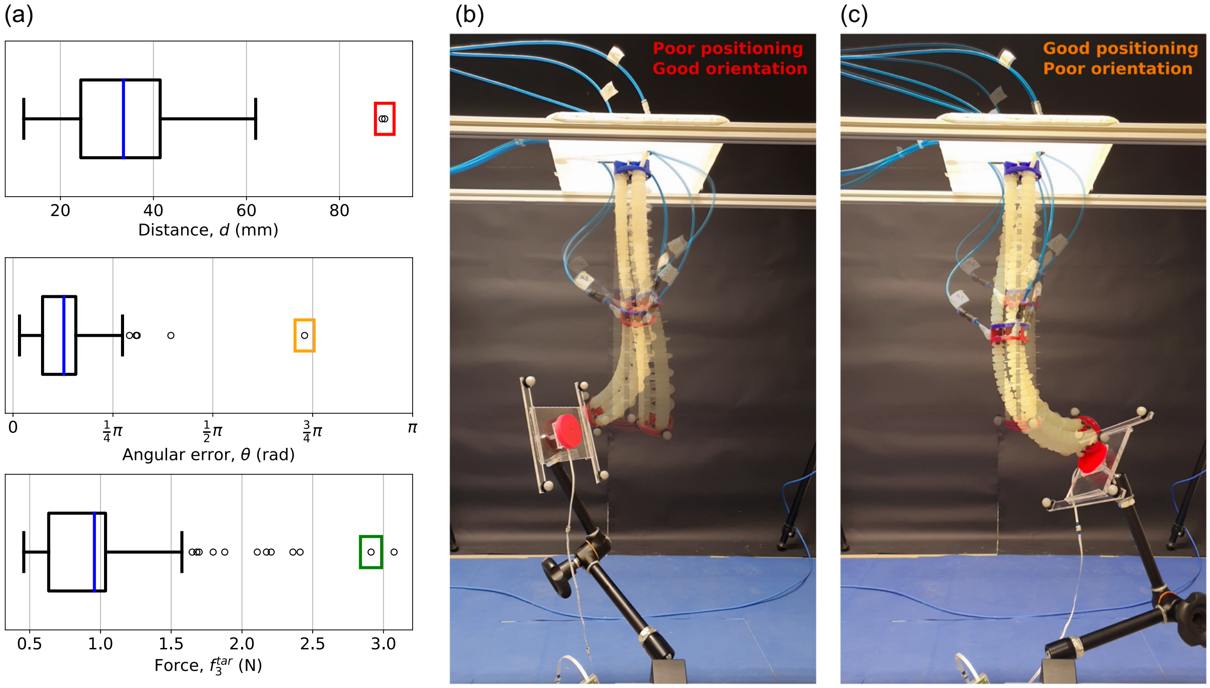

**Figure 9.** Control policy evaluation in the real world. a) Box plots of the distribution of reaching distance $d$, angular error $\theta$ between the end-effector and the object orientation, and the pushing force $f_3^{\text{tar}}$ applied (blue line shows the mean). b) Experiment with a high reaching distance but low orientation error (red). c) Experiment with a low reaching distance but high orientation error (orange). The trial marked green refers to Figure 8.

## 4. Conclusion

Empowering soft robots to perform interaction tasks autonomously still presents challenges. However, the sim-to-real transfer of reinforcement learning policies trained on efficient mechanical models seems promising. This article developed a closed-loop pose/force controller for soft robotic arms to rapidly perform pushing tasks parametrized by position, orientation, and force. The policy trained on a simulated environment comprising a dynamic Cosserat rod model of the soft arm. In particular, PPO was augmented with domain randomization to mitigate the persistent sim-to-real gap in soft robotics. The validation of the control policy investigated the effects of domain randomization in simulation. Despite the approximate training environment, the sim-to-real transfer of the controller was satisfactory, achieving an average pose error of 34 ± 14 mm and 0.40 ± 0.29 rad and applying forces up to 3 N. Moreover, the real-world experiments revealed interesting twisting and balancing behaviors, which naturally emerged from the interaction between the robot and the object. Overall, this article contributes toward soft manipulators learning physical interaction tasks.

### 4.1. Limitation and Future Work

Although the results demonstrated the effectiveness of reinforcement learning for interaction tasks with soft robots, there are opportunities to improve the control policy.

Future work could improve the sim-to-real transfer by reducing the gap between reality and simulation through an effort in model calibration. For instance, a data-driven approach could augment the Cosserat rod model to address unmodelled effects. In addition, the contact model could be improved and validated with experimental data. Moreover, more sophisticated domain randomization strategies could better direct the training process.[48]

Concerning task generation, despite the manipulator's dexterity, some combinations of positions and orientations could be unattainable. Therefore, we envisage improvements in pose control by generating tasks from experimental motion data, which better consider the physiological robot capabilities.

Regarding force measurements in the pushing task, the manipulator could solicit the force sensor only when the end-effector touched the tiny sensor plate. In addition, the soft disk protecting the sensor from damage also attenuated the pushing force. Conversely, forces in the simulation were distributed over the object surface and could be accessed easily. We envisage that engineering ad-hoc physical environments could give more accurate force measurements. Further improvements in the control policy could apply filters to simulated and measured forces to reduce noise, allowing force as a control input in complex manipulation tasks.

## Supporting Information

Supporting Information is available from the Wiley Online Library or from the author.

## Acknowledgements

This work was supported by the Future and Emerging Technologies (FET) program of the European Commission, through the PROBOSCIS project (G.A. ID 863212).

## Conflict of Interest

The authors declare no conflict of interest.

## Keywords

dynamic control, manipulation, reinforcement learning, sim-to-real, soft robots, system modeling

Received: January 2, 2024
Revised: May 3, 2024
Published online:

[1] C. Laschi, M. Cianchetti, *Front. Bioeng. Biotechnol.* **2014**, *2*, 3.
[2] D. Rus, M. T. Tolley, *Nature* **2015**, *521*, 467.
[3] C. Armanini, F. Boyer, A. T. Mathew, C. Duriez, F. Renda, *IEEE Trans. Robot.* **2023**, *39*, 1728.
[4] C. Della Santina, C. Duriez, D. Rus, *IEEE Control Syst. Mag.* **2023**, *43*, 30.
[5] T. George Thuruthel, Y. Ansari, E. Falotico, C. Laschi, *Soft Robot.* **2018**, *5*, 149.
[6] J. Wang, A. Chortos, *Adv. Intell. Syst.* **2022**, *4*, 2100165.
[7] R. J. Webster III, B. A. Jones, *Int. J. Robot. Res.* **2010**, *29*, 1661.
[8] O. Fischer, Y. Toshimitsu, A. Kazemipour, R. K. Katzschmann, *Adv. Intell. Syst.* **2022**, *5*, 2200024.
[9] C. Della Santina, D. Rus, *IEEE Robot. Autom. Lett.* **2019**, *5*, 290.
[10] C. Laschi, T. G. Thuruthel, F. Lida, R. Merzouki, E. Falotico, *IEEE Control Syst. Mag.* **2023**, *43*, 100.
[11] K. Chin, T. Hellebrekers, C. Majidi, *Adv. Intell. Syst.* **2020**, *2*, 1900171.
[12] A. Centurelli, L. Arleo, A. Rizzo, S. Tolu, C. Laschi, E. Falotico, *IEEE Robot. Autom. Lett.* **2022**, *7*, 4741.
[13] F. Piqué, H. T. Kalidindi, L. Fruzzetti, C. Laschi, A. Menciassi, E. Falotico, *IEEE Robot. Autom. Lett.* **2022**, *7*, 5469.
[14] D. Bianchi, M. G. Antonelli, C. Laschi, A. M. Sabatini, E. Falotico, *IEEE Robot. Autom. Mag.* **2023**, https://ieeexplore.ieee.org/abstract/document/10268286.
[15] E. Coevoet, T. Morales-Bieze, F. Largilliere, Z. Zhang, M. Thieffry, M. Sanz-Lopez, B. Carrez, D. Marchal, O. Goury, J. Dequidt, C. Duriez, *Adv. Robot.* **2017**, *31*, 1208.
[16] J. Allard, S. Cotin, F. Faure, P.-J. Bensoussan, F. Poyer, C. Duriez, H. Delingette, L. Grisoni, in *MMVR 15-Medicine Meets Virtual Reality*, IOP Press, Bristol **2007**, Vol. *125*, pp. 13–18.
[17] M. Dubied, M. Y. Michelis, A. Spielberg, R. K. Katzschmann, *IEEE Robot. Autom. Lett.* **2022**, *7*, 5015.
[18] W. Zhao, J. P. Queralta, T. Westerlund, in *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, IEEE, Piscataway, NJ **2020**, pp. 737–744.
[19] L. Arleo, G. Stano, G. Percoco, M. Cianchetti, *Prog. Addit. Manuf.* **2021**, *6*, 243.
[20] C. Alessi, E. Falotico, A. Lucantonio, *IEEE Access* **2023**, *11*, 37840.
[21] C. Alessi, H. Hauser, A. Lucantonio, E. Falotico, in *2023 IEEE Inter. Conf. on Soft Robotics (RoboSoft)*, IEEE, Piscataway, NJ **2023**, pp. 1–7.
[22] A. Bajo, N. Simaan, *Int. J. Robot. Res.* **2016**, *35*, 422.
[23] H. Wang, H. Ni, J. Wang, W. Chen, *IEEE Trans. Control Syst. Technol.* **2019**, *29*, 661.
[24] P. Abbasi, M. A. Nekoui, M. Zareinejad, P. Abbasi, Z. Azhang, *Soft Robot.* **2020**, *7*, 550.
[25] N. Naughton, J. Sun, A. Tekinalp, T. Parthasarathy, G. Chowdhary, M. Gazzola, *IEEE Robot. Autom. Lett.* **2021**, *6*, 3389.
[26] X. Wang, N. Rojas, in *2022 IEEE 5th Inter. Conf. on Soft Robotics (RoboSoft)*, IEEE, Piscataway, NJ **2022**, pp. 247–254.
[27] S. Satheeshbabu, N. K. Uppalapati, T. Fu, G. Krishnan, in *2020 3rd IEEE Inter. Conf. on Soft Robotics (RoboSoft)*, IEEE, Piscataway, NJ **2020**, pp. 497–503.
[28] M. Cianchetti, T. Ranzani, G. Gerboni, T. Nanayakkara, K. Althoefer, P. Dasgupta, A. Menciassi, *Soft Robot.* **2014**, *1*, 122.
[29] Y. Ansari, M. Manti, E. Falotico, Y. Mollard, M. Cianchetti, C. Laschi, *Int. J. Adv. Robot. Syst.* **2017**, *14*, 1729881416687132.
[30] F. Stella, C. Della Santina, J. Hughes, *Nat. Mach. Intell.* **2023**, *5*, 561.
[31] C. Agabiti, E. Ménager, E. Falotico, in *2023 IEEE Inter. Conf. on Soft Robotics (RoboSoft)*, IEEE, Piscataway, NJ **2023**, pp. 1–6.
[32] J. Stüber, C. Zito, R. Stolkin, *Front. Robot. AI* **2020**, *7*, 8.
[33] M. Yang, Y. Lin, A. Church, J. Lloyd, D. Zhang, D. A. Barton, N. F. Lepora, *IEEE Robot. Autom. Lett.* **2023**, *8*, 5480.
[34] Z. Tang, P. Wang, W. Xin, C. Laschi, *IEEE Robot. Autom. Lett.* **2022**, *7*, 8315.
[35] Z. Tang, P. Wang, W. Xin, Z. Xie, L. Kan, M. Mohanakrishnan, C. Laschi, in *2023 IEEE Inter. Conf. on Robotics and Automation (ICRA)*, IEEE, Piscataway, NJ **2023**, pp. 982–988.
[36] Z. Ding, N. F. Lepora, E. Johns, in *2020 IEEE Inter. Conf. on Robotics and Automation (ICRA)*, IEEE, Piscataway, NJ **2020**, pp. 1639–1645.
[37] R. Lal, R. Swaminathan, L. Seenivasan, L. Qiu, H. Ren, in *2021 IEEE Inter. Conf. on Development and Learning (ICDL)*, IEEE, Piscataway, NJ **2021**, pp. 1–6.
[38] Y. Li, X. Wang, K.-W. Kwok, in *2022 IEEE/RSJ Inter. Conf. on Intelligent Robots and Systems (IROS)*, IEEE, Piscataway, NJ **2022**, pp. 7074–7081.
[39] X. Sun, M. Mazur, C.-T. Cheng, *Addit. Manuf.* **2023**, *67*, 103463.
[40] R. Côté, V. Demers, N. R. Demarquette, S. Charlon, J. Soulestin, *Addit. Manuf.* **2023**, *68*, 103509.
[41] M. Gazzola, L. Dudte, A. McCormick, L. Mahadevan, *R. Soc. Open Sci.* **2018**, *5*, 171628.
[42] X. Zhang, F. K. Chan, T. Parthasarathy, M. Gazzola, *Nat. Commun.* **2019**, *10*, 4825.
[43] T. Preclik, C. Popa, U. Rüde, in *Proceedings of Multibody Dynamics*, Brussels, Belgium July **2011**, Vol. *2011*.
[44] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, arXiv preprint arXiv:1707.06347, **2017**.
[45] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, N. Dormann, *J. Mach. Learn. Res.* **2021**, *22*, 12 348.
[46] O. M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, J. Schneider, S. Sidor, J. Tobin, P. Welinder, L. Weng, W. Zaremba, *Int. J. Robot. Res.* **2020**, *39*, 3.
[47] S. R. Eugster, J. Harsch, M. Bartholdt, M. Herrmann, M. Wiese, G. Capobianco, *IEEE Robot. Autom. Lett.* **2022**, *7*, 2471.
[48] B. Mehta, M. Diaz, F. Golemo, C. J. Pal, L. Paull, in *Conf. on Robot Learning*, PMLR November **2020**, pp. 1162–1176.