



Research paper

Autonomous control of soft robots using safe reinforcement learning and covariance matrix adaptation

Shaswat Garg^a, Masoud Goharimanesh^b, Sina Sajjadi^c, Farrokh Janabi-Sharifi^c^{*}^a Department of Mechanical and Mechatronics Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada^b Department of Mechanical Engineering, University of Torbat Heydarieh, Torbat Heydarieh, Iran^c Department of Mechanical, Industrial and Mechatronics Engineering, Toronto Metropolitan University, Toronto, ON M5B 2K3, Canada

ARTICLE INFO

Keywords:

Soft robot
Safe reinforcement learning
Continuum robot
Optimization
Tendon-driven robot
Controller

ABSTRACT

The control of soft robots (such as continuum robots) poses significant challenges due to their coupled dynamics with significant inherent nonlinearities. Recently, model-free reinforcement learning algorithms have been proposed as an attractive alternative to model-based methods to address such a challenging control problem through unsupervised learning. However, the safety of robots is usually ignored while training such algorithms. This is particularly important for medical applications of soft robots. Also, the curse of dimensionality in soft robots makes it difficult for a reinforcement learning algorithm to develop an optimal controller. In this work, we propose a safe phasic soft actor-critic algorithm with a covariance matrix adaptation network which is then tested on different soft robots. We demonstrate that the proposed algorithm could learn an optimal policy quickly while satisfying the safety constraints. We formulated and tested our algorithm for (i) multigait soft robot; (ii) soft gripper robot; and (iii) soft robotic trunk. The proposed algorithm achieved an average of 150% higher rewards compared to other state-of-the-art algorithms. Also, adding the safety layer helped reduce the tracking error by 8 times when compared to the algorithm without a safety layer. The policy is validated in Simulation Open Framework Architecture (SOFA) simulations against other state-of-the-art algorithms in terms of tracking errors.

1. Introduction

Soft robots (SRs), such as continuum robots (CRs), have garnered significant attention in robotics due to their inherent compliance for interaction with their environments (Rus and Tolley, 2015). Their compliance makes SRs an attractive and relatively safe choice for operating in complex, cluttered environments in applications such as medical interventions and search-and-rescue operations (Robinson and Davies, 1999). However, their flexibility, hysteresis, and high degrees-of-freedom (DOFs) pose significant challenges for modeling and control, especially when safety and accuracy are of primary concern. For instance, despite their compliance, SRs can pose safety risks — e.g., accidental cardiac wall punctures by catheters during cardiac interventions which can lead to severe complications such as pseudoaneurysms (Kim et al., 1992).

Numerous control strategies have been proposed for the motion control of SRs (George Thuruthel et al., 2018), which can be broadly categorized into model-based (Hannan and Walker, 2001) and model-free (Dermatas et al., 1996) techniques. Model-based methods leverage either low-fidelity (Jones and Walker, 2006; Li et al., 2018) or

high-fidelity distributed parameter (He et al., 2013) models, but their performance often deteriorates due to intrinsic and extrinsic modeling uncertainties. Conversely, model-free methods using machine learning (ML) have been proposed to address these issues, though they often fail to adapt to real-time changes that are unobserved during training. Among ML methods, reinforcement learning (RL) has emerged as a promising approach due to its ability to autonomously adapt to complex and dynamic environments without prior knowledge of their configurations (Polydoros and Nalpanitidis, 2017).

Recent RL-based approaches for controlling CRs include fuzzy RL (Goharimanesh et al., 2020) as well as methods utilizing deep deterministic policy gradient (DDPG) (Liu et al., 2019; Dai et al., 2021), trust region policy optimization (TRPO) (Centurelli et al., 2022; Kargin and Kołota, 2023), Deep Q Networks (DQN) (Wei et al., 2023; Mazumder, 2023) and Q-learning (Zhang et al., 2017; Liu et al., 2020). However, these approaches are based on trained using kinematics models or simplified dynamics models, making it difficult to transfer policies from simulation to real-world applications (sim2real). Ensemble RL (Morimoto et al., 2021) and stochastic RL methods (Mo et al., 2024) have

^{*} Corresponding author.E-mail addresses: s67garg@uwaterloo.ca (S. Garg), goharimanesh@torbath.ac.ir (M. Goharimanesh), sina.sajjadi@torontomu.ca (S. Sajjadi), fsharif@torontomu.ca (F. Janabi-Sharifi).<https://doi.org/10.1016/j.engappai.2025.110791>

Received 26 December 2023; Received in revised form 16 February 2025; Accepted 3 April 2025

Available online 22 April 2025

0952-1976/© 2025 Elsevier Ltd. All rights are reserved, including those for text and data mining, AI training, and similar technologies.

also been explored to enhance action quality, but they suffer from the “curse of dimensionality” and are computationally expensive due to the combination of multiple models. Additionally, these methods require complex model selection, reducing interpretability and making them sensitive to noise, which offsets the advantages they offer.

Moreover, existing RL-based methods often neglect the safety and constraints of CRs during training, resulting in suboptimal policies. For example, a shielding algorithm introduced in Ji et al. (2021) improves action accuracy near the goal position but does not ensure safety throughout the training process. A force estimation method based on long short-term memory (LSTM) for small-scale continuum robots was introduced by Xiang et al. (2023), enabling high-precision force tracking and compliance control for minimally invasive procedures. Almanzor et al. (2023) introduced a Sim-to-Real framework for pneumatic continuum manipulator control using an LSTM-based simulation and probabilistic inference for learning control (PILCO) for strategy training to fine-tune strategies to bridge the simulation-reality gap. However, both works do not consider safety and constraints during the training process. To address safety concerns, safe reinforcement learning (SRL) methods have been proposed, particularly for autonomous vehicles (Isele et al., 2018) and conventional robots (García and Shafie, 2020). These algorithms aim to maximize expected rewards while satisfying safety constraints during training and deployment (García and Fernández, 2015). Despite these advancements, SRL-based control methods have not yet been extended to SRs/CRs, which are particularly sensitive to safety issues, especially in medical applications.

This paper presents a simple and effective SRL framework to control SRs/CRs. The nomenclature used throughout the paper is shown in Table 1. The primary contributions include:

- This work develops a novel RL algorithm with an integrated safety layer to ensure safe exploration during training. To the best of the authors’ knowledge, this is the first implementation of a safety layer for SRs/CRs. Additionally, a PID controller is incorporated to assist in the initial exploration phase, enhancing stability and learning efficiency.
- From a theoretical perspective, this study introduces phasic soft actor-critic with covariance matrix adaptation (PSAC-CMA), a novel gradient-free policy optimization framework that facilitates efficient parameter sharing between the critic and actor networks. This approach improves learning stability, accelerates convergence, and enhances performance in continuous action spaces compared to traditional gradient-based RL algorithms.
- On the practical side, the proposed RL algorithm is trained within the high-fidelity simulation environment SOFA, enabling an effective sim-to-real transfer. The use of SOFA ensures that the learned policies can be seamlessly deployed on real-world SRs/CRs, thereby improving their reliability and safety in real-world applications (Lai et al., 2023; Gourey et al., 2021; Wu et al., 2022).
- To address the challenges associated with high-dimensional state spaces and computational expense, an ensemble learning approach is employed, combining multiple models to enhance predictive accuracy while maintaining interpretability and robustness against noise.
- The effectiveness of the proposed algorithm is demonstrated by benchmarking it against a state-of-the-art (SOTA) learning-based algorithm for continuum robots from the literature. The results show that the proposed method achieves superior performance in terms of reward optimization, learning efficiency, and safety maintenance during training. These findings establish the contributions of this work in both theoretical advancements and practical applicability in the field of continuum robot control.

By focusing on these contributions, this work not only addresses limitations in existing literature but also emphasizes innovations such as integrating SRL frameworks with safety constraints, enabling efficient exploration, and achieving a practical sim2real transfer.

2. Soft robot model and control architecture

2.1. Equation of motion

The equation of motion for a SR is derived from the principles of continuum mechanics, particularly in the context of deformable bodies. Unlike rigid robots, SRs are composed of highly compliant materials that undergo large deformations, requiring a more complex mathematical framework to describe their motion. The governing equations are typically formulated using the principles of solid mechanics.

A common approach to modeling SR dynamics is through the Lagrangian formulation, which is based on energy principles. The Lagrangian, denoted as L , is defined as the difference between the system’s kinetic energy T and potential energy U , is used to derive the equations of motion. Applying the Euler–Lagrange equation to this system yields:

$$\frac{d}{dt} \left(\frac{\partial L}{\partial \dot{q}_i} \right) - \frac{\partial L}{\partial q_i} = Q_i. \quad (1)$$

Here, q_i represents the generalized coordinates defining the system’s configuration, \dot{q}_i denotes the corresponding generalized velocities, $\frac{\partial L}{\partial \dot{q}_i}$ is the conjugate momentum, and Q_i represents the generalized forces acting on the system (Boyer et al., 2020).

For numerical simulations, these equations are often discretized using the Finite Element Method (FEM) (Thieffry et al., 2018). FEM decomposes the SR into a mesh of small elements, allowing for an accurate approximation of its deformations. The discretized equation of motion for a SR can be expressed as (Qin et al., 2024; Armanini et al., 2023):

$$M(u, t) \frac{\partial^2 u}{\partial t^2} + E(u, \dot{u}, t) \frac{\partial u}{\partial t} + K(u, t)u = g(t). \quad (2)$$

Here, u is the vector of nodal displacements, t is time, M is the mass matrix, E is the damping matrix, K is the stiffness matrix, g is the vector of external forces.

2.2. FEM and RL simulation

Modeling Eq. (2) in the previous section was achieved using SOFA (Faure et al., 2012). SOFA provides a flexible architecture and an extensive catalog of plugins, enabling efficient and accurate simulations of SRs. In this work, we utilized the SoftRobots plugin to model the behavior and control of SRs (Duriez, 2013; Largilliere et al., 2015). The framework employs continuum mechanics and Lagrangian multipliers to simulate the robot’s motion, allowing for precise computation of its position based on actuator inputs such as pressure, displacement, or current. To evaluate the effectiveness of our approach, we conducted simulations on three different SRs/CRs, (i) a multi-gait SR (Gourey and Duriez, 2018), (ii) a soft gripper robot (Hassan et al., 2015; Manti et al., 2015) and (iii) a soft robotic trunk (Wu and Zheng, 2021).

While training the following robots with constrained SRL algorithms, the environment is described using a constrained Markov decision process (CMDP). A simple CMDP can be described using a tuple $\langle C, S, A, P, R, \gamma \rangle$, where $S, A, R: S \rightarrow r, P: S \times A \rightarrow S, C$ and $\gamma \in \mathbb{R}$ denote the continuous state vector, the continuous action vector, the reward function, the state transition function, the set of constraints used to define the Markov decision process (MDP), and a discount factor, respectively. Our algorithm is designed to be versatile and can be trained with any SR. For this study, we focus on three specific robots due to their widespread use in research, well-documented characteristics, and relevance to various applications (Shepherd et al., 2011; Duanmu et al., 2021; Ferrentino et al., 2023; AboZaid et al., 2024; Wu et al., 2022). These robots serve as representative benchmarks, allowing us to demonstrate the adaptability and effectiveness of our approach across diverse soft robotic platforms.

Table 1
Nomenclature.

Symbol	Description	Symbol	Description
α	Temperature	A	Action vector
α_{clone}	Clone of original policy	a_{ag}	Action released by RL policy
β_1^*	Lagrange multiplier	a_c	Corrected action
β_π	Actor learning rate	a_f	Action fused by RL and PID agents
β_Q	Critic learning rate	a_{PID}	Action released by the PID controller
β_V	V-Network learning rate	B	Replay buffer
Δ	Distance between current position and goal	B_{aux}	Auxiliary replay buffer
θ^π	Weights of policy network	C	Constraint function
θ^V	Weights of V-Network	C_i	Constraint
$\theta^{V'}$	Weights of target V-Network	\bar{c}_i	Safety signals
θ^Q	Weights of Q-Network	D	Covariance matrix
γ	Discount factor	$d_1, d_c, d_\sigma, d_\mu$	Covariance Matrix Adaptation (CMA) learning rates
λ	Sample iterations	e_σ	Damping factor for σ
L^{aux}	Auxiliary loss	$E(u, \dot{u}, t)$	Damping matrix of FEM model
L^{jt}	Joint loss function to optimize the policy	$f(\mu_h)$	Evaluation criteria of offspring's
L^{val}	Loss function to optimize state value	$f(s; w_i)$	Safety-layer MLP model
μ_c	Number of candidates in CMA	$g(t)$	Vector of external forces
μ_h	Individual candidate in CMA	J_V	Loss function of the V-Network
μ_w	Variance effective selection mass	J_Q	Loss function of the Q-Network
π	Policy network	K_p, K_f, K_D	Derivative gain of PID
a_i^{old}	Old actions before auxiliary step	$K(u, t)$	Stiffness matrix of FEM model
τ	Tau	L	Lagrangian of system
σ	Step size for CMA	m	Initial mean vector for CMA
ψ	Weights given to CMA candidates	$M(u, t)$	Mass matrix of FEM model
		N_{aux}	Number of auxiliary training iterations
		P	Transition function
		q_c	Anisotropic evolution path
		q_σ	Isotropic evolution path
		q_i	Generalized coordinates describing configuration of the system
		\dot{q}_i	Generalized velocities of the system
		Q_i	Generalized forces acting on the system
		Q	Action value network
		Q_i^{ar}	Target state value function
		R	Reward function
		S	State vector
		s'	Next state
		T	Kinetic Energy of System
		u	Nodal displacement in FEM
		U	Potential Energy of System
		V	State value
		w_{ag}	Weight given to RL action
		w_i	Weights of the safety layer model
		w_{PID}	Weight given to PID action
		X	Lagrangian action integral

2.3. Multigait robot

A multigait robot (MGR) (Shepherd et al., 2011) is a soft locomotive robot composed of elastomeric polymers, and designed without any rigid internal skeleton. It utilizes pneumatic actuation to control its limbs, enabling multiple gaits for sophisticated locomotion. The robot's structure consists of silicon-containing cavities along with stiffer and thinner layers of polydimethylsiloxane (PDMS). The MGR features five independently actuated air cavities, as illustrated in Fig. 1(a).

To model the MGR, we employ a volumetric force distribution for the main body while treating each PDMS layer using a two-dimensional formulation. Actuation is modeled by assuming a uniform pressure distribution within the cavities, oriented orthogonally to the cavity surfaces.

The CMDP's characteristics for the MGR are defined as follows:

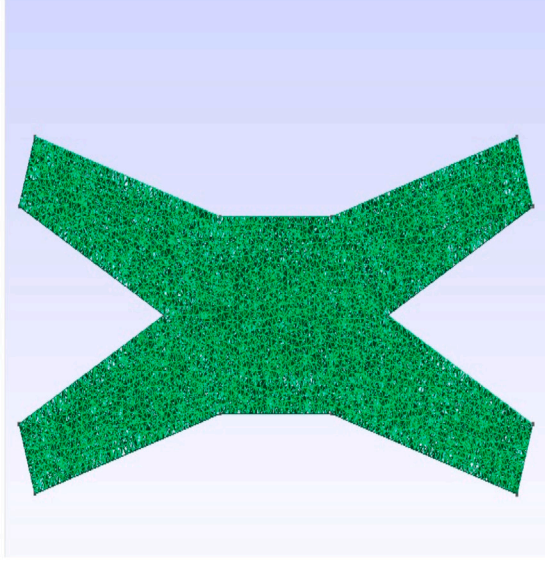
- State (s): Every state $s \in S$ is a $\mathbb{R}^{32 \times 1}$ vector consisting of a position of 32 nodes along the reduced model of the robot (with each element in the model posing 1 DoF).
- Action (a): Every action $a \in A$ is defined as a $\mathbb{R}^{5 \times 1}$ vector consisting of the pressure in the five air cavities. The actions given by the policy cannot be directly used in SOFA. Using the actions, goal pressure is calculated at the left and right cavities of front and rear parts of the robot and the central cavity. The change in

pressure for the five cavities is calculated, which is then provided to the simulation model.

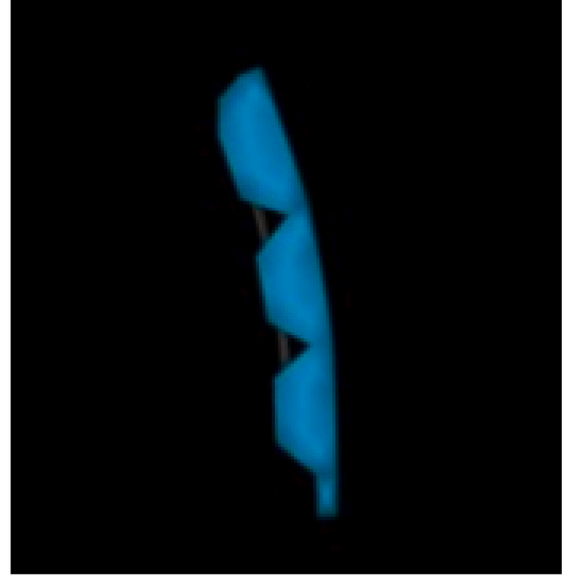
- Constraint (c): During training, it is crucial to ensure that pressure within the air cavities remains within its limits $c_i \in C$. For MGR, we maintain pressure in the range of $P_i \in [-50, 50]$ Pa. With two constraints for each cavity and a total of five actuators, the robot has a total of ten constraints. Constraints are defined as $[-50 - P_i, P_i - 50]$ Pa.
- Rewards (R): A reward function is used to provide feedback to the agents. For our experiments, we set that an episode is successful if the robot is within the goal region i.e., within 1.5 cm of the goal node. For our research, we define the following reward function as shown in (3) to give optimal results upon training for all three robots.

$$R_t = \begin{cases} -0.14 & \text{if the actuators (tendons/pressure) are stable at time } t \\ 100 & \text{if robot is in goal region at time } t \\ -50 & \text{otherwise} \end{cases} \quad (3)$$

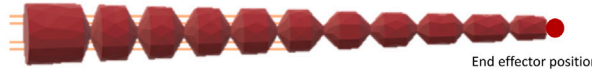
The agent is rewarded with a significant positive value if the robot reaches the goal region. When the robot gets an undesirable condition, e.g., reaching beyond the tendon tensions limit or upon collision with an obstacle, it receives a significant negative value.



(a) Multigait robot



(b) Soft gripper robot



(c) Soft robotic trunk

Fig. 1. Three types of robots used for the study.

2.4. Soft gripper robot

A soft gripper robot (SGR) is designed to emulate the dexterity of a human hand utilizing tendon-driven actuation without a rigid skeleton. Unlike traditional robotic grippers that rely on mechanical joints, the SGR employs soft, compliant materials to securely grasp objects. Each finger incorporates triangular incisions at a 45° angle, enabling an angular displacement of up to 90° . For simplicity, the gripper is modeled with two fingers to perform grasping tasks. The tendon routing and actuation mechanism are illustrated in Fig. 1(b).

In our simulation, a cubical object is randomly placed and grasped using the two-fingered gripper. To simplify modeling, we assume a frictional contact between the gripper and the object while neglecting finger rotation. The applied load is modeled as a 3D force vector that induces deformation in the fingers, mimicking real-world interactions.

The CMDP's characteristics for this tendon-driven robot are defined as follows:

- State (s): Every state $s \in S$ is a $\mathbb{R}^{31 \times 1}$ vector consisting of position of the cubical object, displacement of the tendons, positions of the tips of the finger, and the goal position. The position of the cube is defined by the location of its vertices, providing 24 values. The displacement vector and goal position are defined by 1 and 3 floating values, respectively. Summing all the above vectors, we get a $\mathbb{R}^{31 \times 1}$ state vector.
- Action (a): Every action $a \in A$ is defined as a $\mathbb{R}^{3 \times 1}$ vector representing the translation of finger mesh coordinates. The translation is limited to $[-1, 1]$ mm by using a $\tanh(\cdot)$ function in the output layer of the policy. The actions given by the policy are added to all the coordinates of the mesh, and the changes (if feasible) are then implemented.

- Constraint (c): For the soft gripper, a constraint is placed on the position of the mesh coordinates, as follows. Given a position vector $[x, y, z]$, then $\text{abs}(x) < 1.2$ cm, $\text{abs}(y) < 1.4$ cm, $\text{abs}(z) < 1.2$ cm.
- Rewards (R): A reward function is used to provide feedback to the agents. The episode is successful if the robot is within 1.2 cm of the goal node.

2.5. Soft robotic trunk

A soft robotic trunk (SRT) is a robot that simulates a behavior similar to an elephant trunk. Tendons actuate the model used and consist of eight cables placed symmetrically at a phase difference of 90° and can be independently actuated. The material used to model the robot is isotropic silicon rubber to ensure elastic behavior. The schematic diagram of the robot is shown in Fig. 1(c). Similar to the MGR, a reduced-order model simulates the trunk robot.

The CMDP's characteristics for the SRT robot are defined as follows:

- State (s): Every state $s \in S$ is a $\mathbb{R}^{66 \times 1}$ vector consisting of the goal position and points along which tendons are attached to the robot in the continuous space. This is because, in addition to the goal coordinates $\in \mathbb{R}^3$, a total of the coordinates of 21 points (21×3 values $\in \mathbb{R}^3$) along the tendons are taken, where each point coordinate is obtained by averaging the coordinates of the corresponding four points on each of the first four tendons (out of eight).
- Action (a): Every action $a \in A$ is continuous and defined as a $\mathbb{R}^{8 \times 1}$ vector consisting of the displacement of tendons. The translation is limited to $[-1, 1]$ mm by using a $\tanh(\cdot)$ function in the output layer of the policy. The actions given by the policy are added to

Table 2
Summary of CMDP characteristics for MGR, SGR, and STR.

Parameter	Multigait robot	Soft gripper robot	Soft robotic trunk
dim(S)	32	31	66
dim(A)	5	3	8
S	Location of nodes in the reduced model of the robot (each element has 1 DoF)	Position of the cubical object, displacement of the cables, the tips of the finger, and the goal position	Goal position and points along which tendons are attached to the robot
A	Pressure in air cavities	Tendons displacement is limited to $[-1, 1]$ mm	Tendons displacement is limited to $[-1, 1]$ mm
C	Making sure the pressure in the cavities is between $[-50, 50]$ Pa	Position of the mesh coordinates. Given a position vector $[x, y, z]$ then, $\text{abs}(x) < 1.2$ cm, $\text{abs}(y) < 1.4$ cm, $\text{abs}(z) < 1.2$ cm	Cables lengths

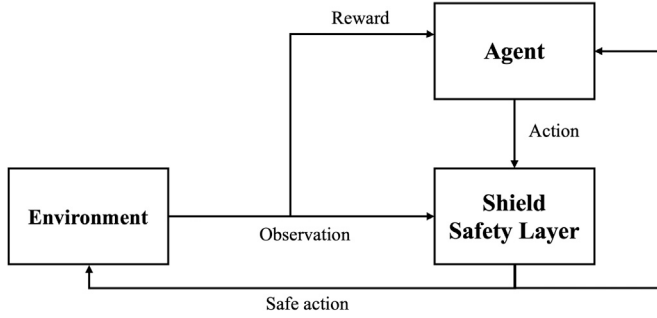


Fig. 2. Reinforcement learning algorithm integrated with a safety layer. The actions given by the agent pass through a safety layer to give the corrected action.

all the cables of the trunk, and the changes (if feasible) are then implemented.

- Constraint (c): The constraint is placed on the cable lengths for the robotic trunk.
- Rewards (R): A reward function is used to provide feedback to the agents. The episode is successful if the robot is within 2 cm of the goal node.

The aforementioned parameters are summarized in Table 2.

3. Algorithm

The algorithm consists of two parts: the safety layer and PSAC-CMA. The complete RL procedure is presented in Algorithms 1–2.

3.1. Safety layer

When optimizing policy using safe exploration (Dalal et al., 2018), we seek to maximize the following expected return,

$$\max_{\theta^\pi} \mathbb{E} \sum_{t=0}^{\infty} \gamma^t R(s_t, \pi(s_t, \theta^\pi)), \quad (4)$$

s.t. $\bar{c}_i \leq C_i \forall i \in [K]$, where \bar{c}_i is the safety signal, C_i is the constraint, and π is the parameterized policy. The above equation is approximated using a multi-layer perceptron (MLP) trained using an Adam Optimizer that takes (s, a) as inputs and returns the corrected action a_c as output. A policy network integrated with a safety layer is shown in Fig. 2. Given a safety replay buffer $B = (s_j, a_j, s'_j)$, we train the MLP model $f(s; w_i)$ by minimizing the following loss,

$$\arg \min_{w_i} \sum_{(s, a, s' \in B)} (\bar{c}_i(s') - (\bar{c}_i(s) + f(s; w_i)^T a))^2. \quad (5)$$

Upon training the layer, the safe action a_c can be calculated using the optimal Lagrange multiplier β_i^* linked with the i th constraint. Then,

$$\beta_i^* = \frac{f(s; w_i)^T \pi(s) + \bar{c}_i(s) - C_i}{f(s; w_i)^T f(s; w_i)} \quad (6)$$

$$a_c = \pi(s) - \beta_i^* f(s; w_i^*),$$

where $i^* = \arg \max_i \beta_i^*$.

3.2. Covariance matrix adaptation

To ease the process of target policy calculation, we attempt to replace the standard Kullback–Leibler (KL) divergence algorithm with covariance matrix adaptation (CMA) which is stochastic in nature and yields a derivative-free optimization method (Hansen and Ostermeier, 1996). Due to gradient-free optimization, CMA is computationally efficient (Igel et al., 2007). The pseudocode for a general CMA is shown in Algorithm 1.

This leads to the introduction of a new variable, sample iterations λ . The mean vector m and step size σ are randomly initialized in the beginning, the covariance matrix D is taken as an identity matrix, and the isotropic (q_σ) and anisotropic (q_c) evolution paths are set to zero.

Algorithm 1 Covariance Matrix Adaptation Evolution Strategy

Initialize: $m, \lambda, \sigma, \mu_c, q_c = 0, q_\sigma = 0$ and $D = I$

while not terminating **do**

$\mu_h = m + \sigma y_h, y_h \sim N(0, D), h = 1, \dots, \lambda$ ▷ Sampling

Evaluating: $f(\mu_h), h = 1, \dots, \lambda$

$m \leftarrow m + \sigma \bar{y}$, where $\bar{y} = \sum_1^{\mu_c} \psi_h y_{h:\lambda}$ ▷ mean update

$q_\sigma \leftarrow (1 - d_\sigma) q_\sigma + \sqrt{\frac{d_\sigma(2 - d_\sigma) \mu_w}{D}} \bar{y}$

$\sigma \leftarrow \sigma \exp(\frac{d_\sigma}{\mathbb{E}[\|N(0, I)\|]} - 1)$ ▷ step-size control update

$q_c \leftarrow (1 - d_c) q_c + \sqrt{d_c(1 - d_c) \mu_w} \bar{y}$

$D \leftarrow (1 - d_1 - d_\mu) D + d_1 q_c q_c^T + d_\mu \sum_1^{\mu_c} \psi_h y_{h:\lambda} y_{h:\lambda}^T$ ▷ covariance matrix update

end while

3.3. Integration with PID controller

To support the initial exploration of state and action space, we integrate a controller with the PSAC-CMA algorithm. The proportional integral derivative (PID) controller continuously keeps track of an error term $e(t)$ calculated as the difference between the desired setpoint and feedback value.

For our simulation, we tuned $K_p = 0.2$, $K_I = 0.01$, and $K_D = 0.01$ to give the best results. We used the PID controller for our study due to its ability to take advantage of the three methods. The P term of the controller produces a constant steady-state error with a stable gain. The I controller reduces the steady-state error. The D controller minimizes the rate of change of error. The action given by the RL algorithm and the PID controller is fused using a weighted average method.

The weights are assigned so that, during the initial learning phase, more importance is given to the actions derived using the PID controller. As the policy learns, the weightage given to the controller is decreased, with the RL algorithm making all the decisions toward the end of the learning phase. The fused action is given by (7),

$$a_f = w_{PID} a_{PID} + w_{ag} a_{ag}, w_{PID} = \frac{N - j}{N}, w_{ag} = \frac{j}{N}, \quad (7)$$

where, w_{PID} and w_{ag} are the weights assigned to the PID controller and RL algorithm, respectively. N is the total number of episodes, and j is the current episode.

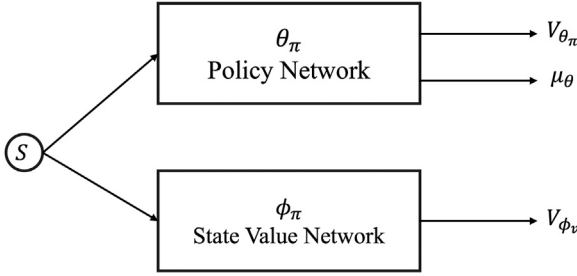


Fig. 3. Network architecture of phasic soft actor-critic (PSAC) showing both shared and separate networks.

3.4. PSAC-CMA

Taking inspiration from the phasic policy gradient (PPG) (Cobbe et al., 2021), we introduce a phasic soft-actor critic. While working with policy and value networks, selecting between separate and shared networks is crucial. Shared networks help facilitate the sharing of valuable features, but different networks are used to avoid interference between objectives. PPG takes advantage of both worlds by following a two-phase optimization one that distills features and another advances training. As shown in Algorithm 2, there are two alternating phases: the policy and the auxiliary phases. During the policy phase, we train the agent using soft actor-critic (SAC) (Haarnoja et al., 2018). We distill features from the state value function into the agent in the auxiliary phase. The PSAC network used in the schematic algorithm is shown in Fig. 3. The value function is trained by minimizing the residual error,

$$J_V = \mathbb{E}_{s_t \sim B_t} \left[\frac{(V(s_t; \theta^V) - \mathbb{E}_{a_t \sim \pi} [Q(a_t, s_t; \theta^Q) - \log \pi(a_t | s_t; \theta^\pi)])^2}{2} \right]. \quad (8)$$

Here, the states are sampled from replay buffer B and action is sampled from the current policy. To train the Q function, the following error is used,

$$J_Q = \mathbb{E}_{(s_t, a_t, s_{t+1}) \sim B_t} \left[\frac{(Q(a_t, s_t; \theta^Q) - Q_t^{tar})^2}{2} \right], \quad (9)$$

where,

$$Q_t^{tar} = (r_t + \gamma V'(s_{t+1}; \theta^{V'})). \quad (10)$$

The weights of the policy function are updated using the CMA algorithm. The PSAC-CMA algorithm benefits from the combination of phasic optimization and covariance matrix adaptation, which allows for efficient learning and adaptation. The phasic optimization approach ensures that the policy and value networks do not interfere with each other, leading to stable and robust learning processes. To facilitate the transfer of knowledge between the state function and policy, auxiliary training is also introduced. At each training step, an auxiliary replay buffer (B_{aux}) is maintained that stores the (s_t, Q_t^{tar}, a_t) tuple. During the auxiliary phase, the policy weights are trained by optimizing the error,

$$L^{aux} = L^{val} + L^{jt}, \quad (11)$$

where,

$$L^{val} = \frac{\mathbb{E}_{(s_t, Q_t^{tar}) \sim B_{aux}} [(V(s_t; \theta^\pi) - Q_t^{tar})^2]}{2} \quad (12)$$

$$L^{jt} = \mathbb{E}_{a_t^{old} \sim B_{aux}} [K L_{loss}(a_t^{old} - \pi(a_t | s_t; \theta^\pi))].$$

In Fig. 4, we display the process of the proposed controller to control a circular trajectory for a SR.

Developing the PSAC-CMA algorithm presented significant challenges due to its innovative approach compared to traditional methods. Integrating a robust safety layer into the reinforcement learning framework was crucial but complex, requiring a careful design to protect the

Algorithm 2 PSAC-CMA

Initialize: $\pi(\cdot|\cdot; \theta^\pi), V(\cdot; \theta^V), V'(\cdot; \theta^{V'})$ and $Q(\cdot, \cdot; \theta^Q)$

Input: Learning rates $\beta_Q, \beta_V, \beta_\pi$ and τ

for number of episodes = 1, 2, ... **do**

 Receive initial observation s_1

for $t = 1, T$ **do**

 Receive action $a_{t(ag)} = \pi(a_t | s_t; \theta^\pi)$

 Receive action $a_{t(PID)}$ from a PID controller

 Fuse both actions using weighted average method to receive

$a_{t(f)}$

 Pass $a_{t(f)}$ through the safety layer to get $a_c = f(a_c, s_t)$

 Observe reward r_t and new status s_{t+1}

 Store transition tuple (s_t, a_c, r_t, s_{t+1}) in replay buffer B

 Sample a random minibatch of N_{tr} transitions (s_i, a_i, r_i, s_{i+1})

from B

 Update value function $\theta^V \leftarrow \theta^V + \beta_V \nabla J_V(\theta^V)$

 Update policy using CMA

 Update critic function $\theta^Q \leftarrow \theta^Q + \beta_Q \nabla J_Q(\theta^Q)$

 Target network update $\theta^{V'} = (1 - \tau)\theta^{V'} + \tau\theta^V$

if perform auxiliary training **then**

 Update policy function $\theta^\pi \leftarrow \theta^\pi + \beta_\pi \nabla L^{aux}$

end if

end for

end for

Table 3

PSAC-CMA hyper-parameters used for training using 10K time-step benchmark.

Hyper-parameter	Value
Discount (γ)	0.99
Tau (τ)	0.005
Actor learning rate (β_π)	0.001
Critic learning rate (β_Q)	0.002
V network learning rate (β_V)	0.002
Update iteration	5
Temperature (α)	0.2

robot without hindering learning. Adapting the covariance matrix for continuous state and action spaces of SRs demanded advanced computational techniques to balance exploration and exploitation. Managing the computational intensity of training simulations and addressing the nonlinear dynamics and high DOFs of SRs further underscored the algorithm's advanced nature and the rigorous innovation needed for its development.

4. Simulation results

The action space of the robots can be either continuous or discrete, depending on how they are modeled within the simulation environment. To evaluate the performance of PSAC-CMA, we benchmarked it against SOTA RL algorithms, including DDPG (Polydoros and Nalpan-tidis, 2017), SAC (Haarnoja et al., 2018; Li et al., 2021; Gao et al., 2024; Hu et al., 2023), TD3 (Fujimoto et al., 2018) and SoftQ (Haarnoja et al., 2017). During off-policy training, all robots were tasked with reaching a designated target position within their respective workspaces. After each episode, the robots were reset to their initial positions. Details of the simulation environment are summarized in Table 3.

To ensure optimal performance, the hyperparameters of PSAC-CMA were carefully fine-tuned through an iterative process. The discount factor γ was set to emphasize long-term rewards, ensuring the policy prioritized long-term goal attainment. The soft update parameter $\tau = 0.005$ enabled smooth updates to the target networks, promoting stable learning. The learning rate was chosen to balance speed and stability, preventing divergence from the optimal solution. The update iteration parameter was set to 5, regulating the number of parameter updates per

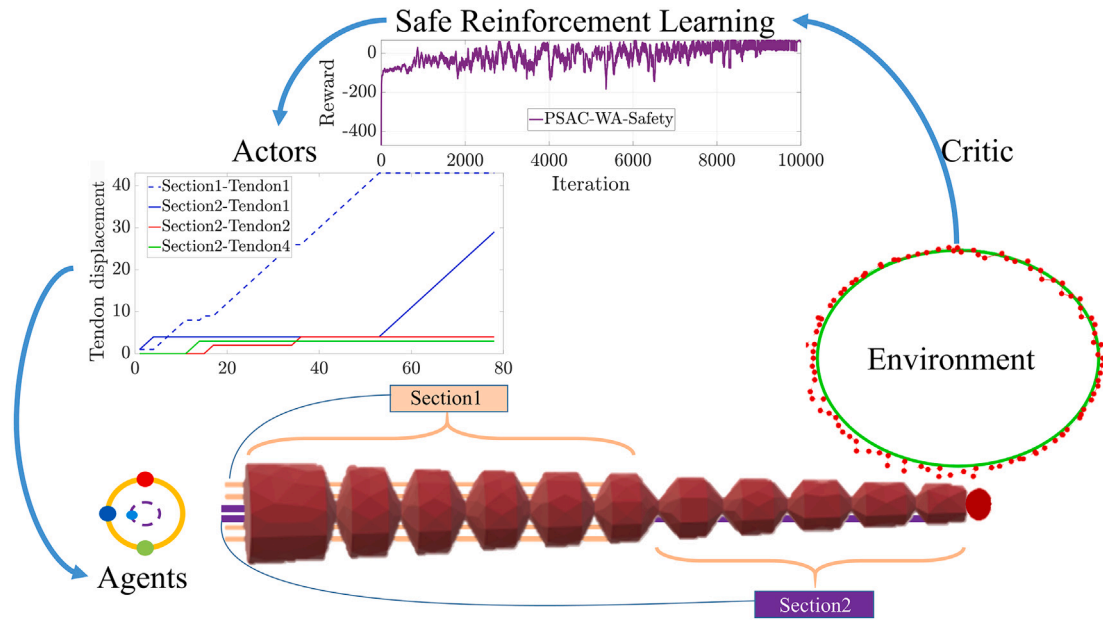


Fig. 4. The control process via SRL for a soft trunk robot.

Table 4

Rewards after sampling 10,000 steps.

Robot	PSAC-CMA	TD3	SAC	DDPG	SoftQ
MGR	95.33	4.23	-64.34	-24.12	-16.9
SGR	62.12	-76.94	-32.17	-69.63	-79.31
SRT	73.47	-42.88	-64.34	-69.7	-84.21

training step and striking a balance between computational efficiency and convergence speed. The temperature parameter controlled the exploration–exploitation trade-off, ensuring sufficient exploration of the action space while avoiding premature convergence to suboptimal policies.

All algorithms were trained for 10,000 episodes, as most exhibited asymptotic performance within this range. To assess their effectiveness, maximum reward (MR), mean square error (MSE), and mean absolute error (MAE) were used as evaluation metrics. MR represented the highest reward achieved during training, reflecting the efficiency of the learning process in achieving the desired goal. Higher MR values indicated better performance. MSE and MAE provided insights into trajectory tracking accuracy, where lower values signified more precise control. These key performance indicators (KPIs) collectively measured effectiveness, safety, efficiency, and precision, aligning with the objective of developing a robust and reliable control system for SRs.

The maximum reward received during the training is listed in Table 4 and represented in Fig. 5.

The results, as illustrated in Fig. 5, demonstrated that PSAC-CMA consistently outperformed the baseline algorithms, achieving significantly higher rewards after 10,000 training steps. The trained policy enabled the robots to reach their target positions with greater accuracy than alternative methods. Specifically, PSAC-CMA achieved reward values that were 1.5 times higher than TD3, 1.6 times higher than SAC, and 1.7 times higher than both DDPG and SoftQ. This highlighted the superior learning efficiency and policy effectiveness of the proposed approach.

The reward trajectories over 10,000 steps, depicted in Fig. 6, further confirmed that PSAC-CMA outperformed the baseline algorithms in terms of reward optimization. On average, PSAC-CMA reached optimal behavior within 1000 episodes, whereas DDPG, SAC, TD3, and SoftQ required approximately 5000 episodes to achieve similar performance levels. This efficiency in learning translates to faster and more reliable

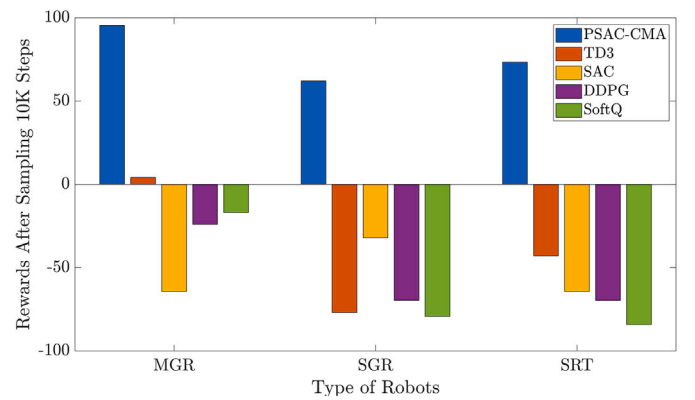


Fig. 5. Comparison of rewards achieved by PSAC-CMA, TD3, SAC, DDPG and SoftQ algorithm after sampling 10K steps.

deployment in real-world applications. A key advantage of PSAC-CMA was its integration of a safety layer, which significantly enhanced the safety of the learning process. During training, PSAC-CMA satisfied task constraints 71% of the time, compared to only 45% for SAC. This aspect is particularly crucial in applications where strict adherence to safety constraints is necessary, such as in medical robotics.

To test the impact of the safety layer on the algorithm, we made the trunk robot move along a circular trajectory and square trajectory of appropriate dimensions. It consists of 100 waypoints and the trained policy is tasked with controlling the robotic trunk to each waypoint, starting from the previous one. The error values of the PASC-CMA with and without safety layer were recorded and are displayed in Fig. 7. Fig. 7 displays the trajectories represented by a green line, while the path of the robotic trunk when controlled using the safety layer is illustrated by red dots and without the safety layer using hollow blue dots (see Tables 5 and 6).

The addition of the safety layer in PSAC-CMA resulted in substantial reductions in tracking errors. For instance, in tests with circular and square trajectories each of which was performed 5 times, the MSE and MAE were significantly lower with the safety layer than without it. The confidence achieved with the safety layer was also higher by 3%. Specifically, the MSE was 8 times less for the circular trajectory

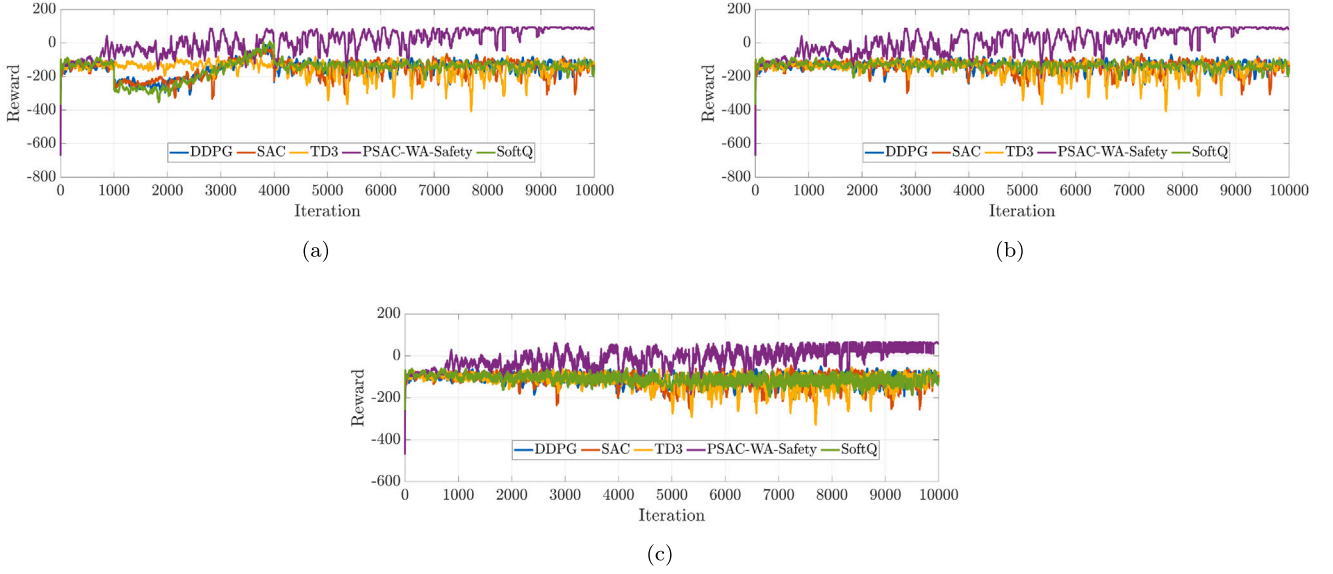


Fig. 6. The results of different RL algorithms (DDPG, SAC, SoftQ, and TD3) against PSAC-CMA with a safety layer, tested on (a) MGR, (b) SGR, and (c) STR.

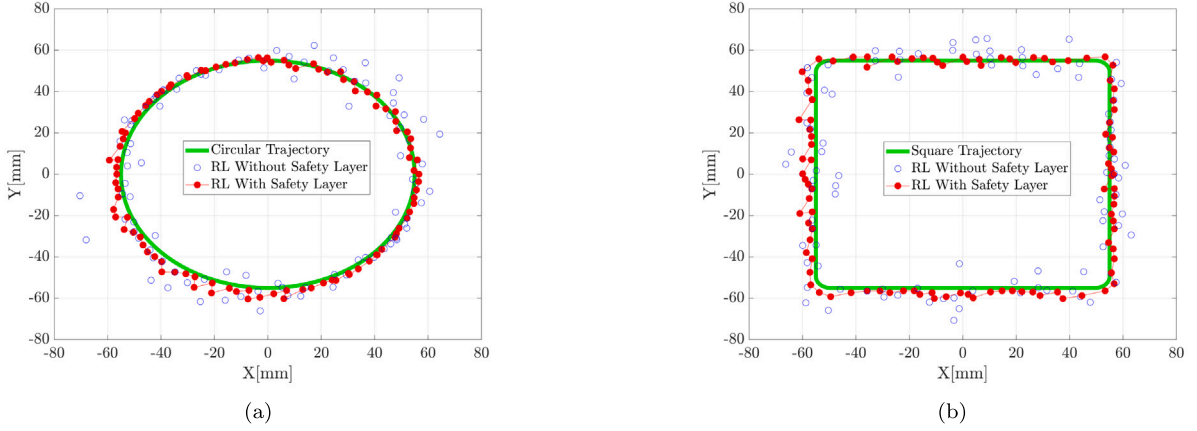


Fig. 7. Error values of PSAC-CMA with and without a safety layer tested on SRT tracking: (a) circular trajectory, and (b) square trajectory. The trajectory by RL policy without safety layer is seen in hollow blue bubbles and the trajectory by RL with a safety layer is depicted by red filled bubbles. As can be observed, the policy without a safety layer is more scattered and has more errors while following the circular and square trajectory.

Table 5
Error of PSAC-CMA for the circular trajectory.

Algorithm	MSE (mm)	MAE (mm)	Confidence
With SL	3.48	1.27	94%
Without SL	24.54	3.62	90%

Table 6
Error of PSAC-CMA for the square trajectory.

Algorithm	MSE (mm)	MAE (mm)	Confidence
With SL	2.18	1.05	95%
Without SL	35.18	4.606	92%

and 17 times less for the square trajectory. As can be observed, we achieve minimum error when the RL algorithm is trained with the safety layer. The control effort of SRT tendons while moving along circular trajectory is shown in Fig. 8.

Beyond comparisons with SOTA RL algorithms, PSAC-CMA was evaluated against another recent learning-based algorithm (INNC) (Garg et al., 2022). To assess the comparative performance, both approaches were tasked with following complex trajectories, as depicted

in Fig. 9. The results demonstrated that PSAC-CMA not only exhibited superior tracking accuracy but also maintained greater stability across varying conditions. Specifically, PSAC-CMA achieved a MSE of 0.5 mm over the two trajectories tested, while INNC had an MSE of 1 mm. This further validated the robustness and adaptability of the proposed method, proving its potential for broader applications in robotic control and RL.

5. Conclusion

This paper presented a safe model-free reinforcement learning algorithm for a SR trained in simulation using SOFA software. We proposed a constraint-based control scheme that ensures the robot's safety based on the constraints defined for the problem. Also, to facilitate learning of the continuous state and action space, we integrated a PID control during the initial stages of training. The reinforcement learning algorithm (PSAC-CMA) was tested against three SR models, namely (i) multigait robot, (ii) soft gripper robot, and (iii) soft robotic trunk. A detailed study of the three simulation models was presented along with the simulation settings. In the simulation, the average error between any randomly selected target position and starting trunk position with the proposed algorithm was 30% less compared to SAC. Since

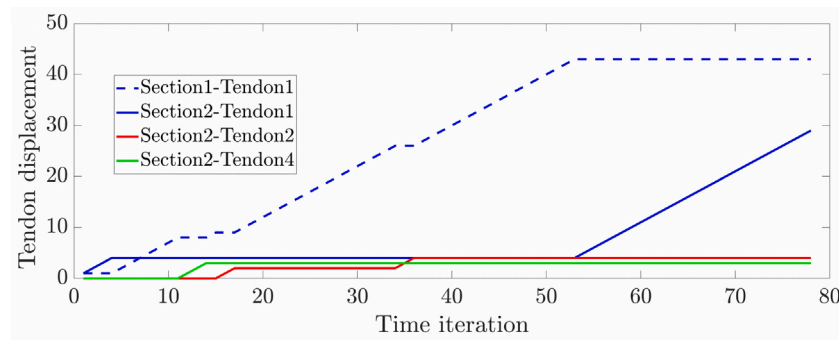


Fig. 8. Control effort of the active tendons of SRT tracking the circular trajectory.

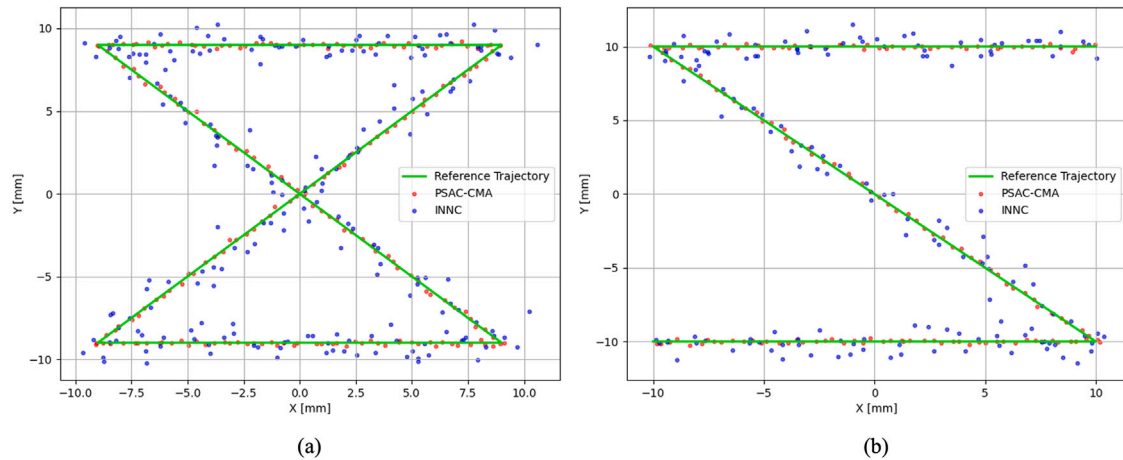


Fig. 9. Error analysis of PSAC-CMA and INNC for SRT tracking: (a) a trajectory composed of two triangles sharing a common vertex and (b) a trajectory representing the letter “Z”. The reference trajectory (green) serves as the ideal path, while PSAC-CMA (red dots) represents the proposed algorithm, and INNC (blue dots) shows the comparison algorithm. The scattered points indicate deviations from the reference trajectory.

SOFA models work on FEM modeling and have been validated with real-life experiments, the proposed RL algorithm can be used for real-life operations. Future work will focus on real-world implementation and testing on physical SRs to validate its effectiveness, exploring advanced constraint-based control schemes, and integrating with other control techniques such as MPC for enhanced robustness. Additionally, extending the algorithm to diverse SR architectures, optimizing PID parameters, and ensuring long-term stability are crucial. Research into collaborative multi-robot systems and human–robot interaction can further broaden its applications in various fields such as search and rescue, medical robotics, and industrial automation.

CRedit authorship contribution statement

Shaswat Garg: Writing – original draft, Validation, Software, Methodology, Conceptualization. **Masoud Goharimanesh:** Writing – original draft, Validation. **Sina Sajjadi:** Writing – review & editing. **Farrokh Janabi-Sharifi:** Writing – review & editing, Supervision, Project administration, Funding acquisition.

Funding statement

This work was supported by MITACS Globalink Research Award and Natural Sciences and Engineering Research Council of Canada under grant # 2017-06930.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing

interests: Shaswat Garg and Farrokh Janabi-Sharifi declare that they have received financial support from Mitacs and the Natural Sciences and Engineering Research Council of Canada (NSERC), respectively. The other authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors thank Dr. Amir Jalali (TMU) for providing valuable insights regarding the modeling of soft robots.

Data availability

No data was used for the research described in the article.

References

- AboZaid, Y.A., Aboelrayat, M.T., Fahim, I.S., Radwan, A.G., 2024. Soft robotic grippers: A review on technologies, materials, and applications. *Sens. Actuators A: Phys.* 115380.
- Almanzor, E., Ye, F., Shi, J., Thuruthel, T.G., Wurdemann, H.A., Iida, F., 2023. Static shape control of soft continuum robots using deep visual inverse kinematic models. *IEEE Trans. Robot.* 39 (4), 2973–2988.
- Armanini, C., Boyer, F., Mathew, A.T., Duriez, C., Renda, F., 2023. Soft robots modeling: A structured overview. *IEEE Trans. Robot.* 39 (3), 1728–1748.
- Boyer, F., Lebastard, V., Candelier, F., Renda, F., 2020. Dynamics of continuum and soft robots: A strain parameterization based approach. *IEEE Trans. Robot.* 37 (3), 847–863. <http://dx.doi.org/10.1109/TRO.2020.3036618>.

- Centurelli, A., Arleo, L., Rizzo, A., Tolu, S., Laschi, C., Falotico, E., 2022. Closed-loop dynamic control of a soft manipulator using deep reinforcement learning. *IEEE Robot. Autom. Lett. (RA-L)* 7 (2), 4741–4748. <http://dx.doi.org/10.1109/LRA.2022.3146903>.
- Cobbe, K.W., Hilton, J., Klimov, O., Schulman, J., 2021. Phasic policy gradient. In: Meila, M., Zhang, T. (Eds.), *Int. Conf. Mach. Learn., ICML*, In: *Proceedings of Machine Learning Research*, vol. 139, PMLR, pp. 2020–2027, URL <https://proceedings.mlr.press/v139/cobbe21a.html>.
- Dai, J., Zhu, M., Feng, Y., 2021. Stiffness control for a soft robotic finger based on reinforcement learning for robust grasping. In: *Int. Conf. Mechatron. Mach. Vis. Pract. M2VIP*, IEEE, pp. 540–545. <http://dx.doi.org/10.1109/M2VIP49856.2021.9665056>.
- Dalal, G., Vijitham, K., Vecerik, M., Hester, T., Paduraru, C., Tassa, Y., 2018. Safe exploration in continuous action spaces. <http://dx.doi.org/10.48550/arXiv.1801.08757>, arXiv preprint [arXiv:1801.08757](https://arxiv.org/abs/1801.08757).
- Dermatas, E., Nearchou, A., Aspragathost, N., 1996. Error-back-propagation solution to the inverse kinematic problem of redundant manipulators. *Robot. Comput. Integr. Manuf.* 12 (4), 303–310. [http://dx.doi.org/10.1016/S0736-5845\(96\)00008-7](http://dx.doi.org/10.1016/S0736-5845(96)00008-7).
- Duanmu, Z., Stommel, M., Cheng, L.K., Xu, W., 2021. Simulation of solid meal digestion in a soft gastric robot using SOFA. In: 2021 27th International Conference on Mechatronics and Machine Vision in Practice. M2VIP, IEEE, pp. 357–362.
- Duriez, C., 2013. Control of elastic soft robots based on real-time finite element method. In: *IEEE Int. Conf. Robot. Automat., ICRA*, IEEE, pp. 3982–3987. <http://dx.doi.org/10.1109/ICRA.2013.6631138>.
- Faure, F., Duriez, C., Delingette, H., Allard, J., Gilles, B., Marchesseau, S., Talbot, H., Courtecuisse, H., Bousquet, G., Peterlik, I., et al., 2012. Sofa: A multi-model framework for interactive physical simulation. *Soft Tissue Biomech. Model. Comput. Assist. Surg.* 283–321. http://dx.doi.org/10.1007/8415_2012_125.
- Ferrentino, P., Roels, E., Brancart, J., Terryn, S., Van Assche, G., Vanderborcht, B., 2023. Finite element analysis-based soft robotic modeling: Simulating a soft actuator in sofa. *IEEE Robot. Autom. Mag.*.
- Fujimoto, S., Hoof, H., Meger, D., 2018. Addressing function approximation error in actor-critic methods. In: *Int. Conf. Mach. Learn., ICML*, PMLR, pp. 1587–1596, URL <https://proceedings.mlr.press/v80/fujimoto18a.html>.
- Gao, J., Li, Y., Chen, Y., He, Y., Guo, J., 2024. An improved SAC-based deep reinforcement learning framework for collaborative pushing and grasping in underwater environments. *IEEE Trans. Instrum. Meas.* 73, 1–14.
- Garcia, J., Fernández, F., 2015. A comprehensive survey on safe reinforcement learning. *J. Mach. Learn. Res.* 16 (1), 1437–1480.
- García, J., Shafie, D., 2020. Teaching a humanoid robot to walk faster through safe reinforcement learning. *Eng. Appl. Artif. Intell.* 88, 103360. <http://dx.doi.org/10.1016/j.engappai.2019.103360>.
- Garg, S., Dudeja, S., Rastogi, V., 2022. Inverse kinematics of tendon driven continuum robots using invertible neural network. In: 2022 2nd International Conference on Computers and Automation. CompAuto, IEEE, pp. 82–86.
- George Thuruthel, T., Ansari, Y., Falotico, E., Laschi, C., 2018. Control strategies for soft robotic manipulators: A survey. *Soft Robot. (SoRo)* 5 (2), 149–163. <http://dx.doi.org/10.1089/soro.2017.000>.
- Goharimaneh, M., Mehrkish, A., Janabi-Sharifi, F., 2020. A fuzzy reinforcement learning approach for continuum robot control. *J. Int. Robot. Syst.* 100, 809–826. <http://dx.doi.org/10.1007/s10846-020-01237-6>.
- Goury, O., Carrez, B., Duriez, C., 2021. Real-time simulation for control of soft robots with self-collisions using model order reduction for contact forces. *IEEE Robot. Autom. Lett.* 6 (2), 3752–3759.
- Goury, O., Duriez, C., 2018. Fast, generic, and reliable control and simulation of soft robots using model order reduction. *IEEE Trans. Robot.* 34 (6), 1565–1576. <http://dx.doi.org/10.1109/TRO.2018.2861900>.
- Haarnoja, T., Tang, H., Abbeel, P., Levine, S., 2017. Reinforcement learning with deep energy-based policies. In: *Int. Conf. Mach. Learn., ICML*, PMLR, pp. 1352–1361.
- Haarnoja, T., Zhou, A., Abbeel, P., Levine, S., 2018. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In: *Int. Conf. Mach. Learn., ICML*, PMLR, pp. 1861–1870, URL <https://proceedings.mlr.press/v80/haarnoja18b.html>.
- Hannan, M.W., Walker, I.D., 2001. Analysis and experiments with an elephant's trunk robot. *Adv. Robot.* 15 (8), 847–858. <http://dx.doi.org/10.1163/156855301317198160>.
- Hansen, N., Ostermeier, A., 1996. Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In: *IEEE Int. Conf. Evolut. Comput.*, IEEE, pp. 312–317. <http://dx.doi.org/10.1109/ICEC.1996.542381>.
- Hassan, T., Manti, M., Passetti, G., d'Elia, N., Cianchetti, M., Laschi, C., 2015. Design and development of a bio-inspired, under-actuated soft gripper. In: *Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. EMBC*, IEEE, pp. 3619–3622. <http://dx.doi.org/10.1109/EMBC.2015.7319176>.
- He, B., Wang, Z., Li, Q., Xie, H., Shen, R., 2013. An analytic method for the kinematics and dynamics of a multiple-backbone continuum robot. *Int. J. Adv. Robot. Syst.* 10 (1), 84. <http://dx.doi.org/10.5772/54051>.
- Hu, Y., Fu, J., Wen, G., 2023. Graph soft actor-critic reinforcement learning for large-scale distributed multirobot coordination. *IEEE Trans. Neural Netw. Learn. Syst.*.
- Igel, C., Hansen, N., Roth, S., 2007. Covariance matrix adaptation for multi-objective optimization. *Evol. Comput.* 15 (1), 1–28. <http://dx.doi.org/10.1162/evco.2007.15.1.1>.
- Isele, D., Nakhaei, A., Fujimura, K., 2018. Safe reinforcement learning on autonomous vehicles. In: *IEEE Int. Conf. Intell. Robots. Syst., IROS*, IEEE, pp. 1–6. <http://dx.doi.org/10.1109/IROS.2018.8593420>.
- Ji, G., Yan, J., Du, J., Yan, W., Chen, J., Lu, Y., Rojas, J., Cheng, S.S., 2021. Towards safe control of continuum manipulator using shielded multiagent reinforcement learning. *IEEE Robot. Autom. Lett. (RA-L)* 6 (4), 7461–7468. <http://dx.doi.org/10.1109/LRA.2021.3097660>.
- Jones, B.A., Walker, I.D., 2006. Kinematics for multisection continuum robots. *IEEE Trans. Robot.* 22 (1), 43–55. <http://dx.doi.org/10.1109/TRO.2005.861458>.
- Kargin, T.C., Kolota, J., 2023. A reinforcement learning approach for continuum robot control. *J. Intell. Robot. Syst.* 109 (4), 77.
- Kim, D., Orron, D.E., Skillman, J.J., Kent, K.C., Porter, D.H., Schlam, B.W., Carrozza, J., Reis, G.J., Baime, D.S., 1992. Role of superficial femoral artery puncture in the development of pseudoaneurysm and arteriovenous fistula complicating percutaneous transfemoral cardiac catheterization. *Catheter. Cardiovasc. Diagn.* 25 (2), 91–97. <http://dx.doi.org/10.1002/ccd.1810250203>.
- Lai, J., Ren, T.-A., Yue, W., Su, S., Chan, J.Y., Ren, H., 2023. Sim-to-real transfer of soft robotic navigation strategies that learns from the virtual eye-in-hand vision. *IEEE Trans. Ind. Inform.*.
- Largilliere, F., Verona, V., Coevoet, E., Sanz-Lopez, M., Dequidt, J., Duriez, C., 2015. Real-time control of soft-robots using asynchronous finite element modeling. In: *IEEE Int. Conf. Robot. Automat., ICRA*, IEEE, pp. 2550–2555. <http://dx.doi.org/10.1109/ICRA.2015.7139541>.
- Li, M., Kang, R., Geng, S., Guglielmino, E., 2018. Design and control of a tendon-driven continuum robot. *Trans. Inst. Meas. Control.* 40 (11), 3263–3272. <http://dx.doi.org/10.1177/0142331216685607>.
- Li, G., Shintake, J., Hayashibe, M., 2021. Deep reinforcement learning framework for underwater locomotion of soft robot. In: 2021 IEEE International Conference on Robotics and Automation. ICRA, IEEE, pp. 12033–12039.
- Liu, W., Jing, Z., D'Eleuterio, G., Chen, W., Yang, T., Pan, H., 2019. Shape memory alloy driven soft robot design and position control using continuous reinforcement learning. In: *Int. Conf. Int. Auton. Syst., ICoIAS*, IEEE, pp. 124–130. <http://dx.doi.org/10.1109/ICoIAS.2019.00028>.
- Liu, W., Jing, Z., Pan, H., Qiao, L., Leung, H., Chen, W., 2020. Distance-directed target searching for a deep visual servo sma driven soft robot using reinforcement learning. *J. Bionic. Eng.* 17, 1126–1138. http://dx.doi.org/10.1007/978-3-319-65289-4_17.
- Manti, M., Hassan, T., Passetti, G., D'Elia, N., Laschi, C., Cianchetti, M., 2015. A bioinspired soft robotic gripper for adaptable and effective grasping. *Soft Robot. (SoRo)* 2 (3), 107–116. <http://dx.doi.org/10.1089/soro.2015.0009>.
- Mazumder, A., 2023. Reinforcement learning based controller for a soft continuum robot. In: 2023 International Conference on Big Data, Knowledge and Control Systems Engineering. BDKCSE, IEEE, pp. 1–6.
- Mo, H., Wei, R., Kong, X., Zhai, Y., Liu, Y., Sun, D., 2024. Data-efficient learning control of continuum robots in constrained environments. *IEEE Trans. Autom. Sci. Eng.*.
- Morimoto, R., Nishikawa, S., Niyama, R., Kuniyoshi, Y., 2021. Model-free reinforcement learning with ensemble for a soft continuum robot arm. In: *IEEE Int. Conf. Soft Robot., RoboSoft*, IEEE, pp. 141–148. <http://dx.doi.org/10.1109/RoboSoft51838.2021.9479340>.
- Polydoros, A.S., Nalpanitidis, L., 2017. Survey of model-based reinforcement learning: Applications on robotics. *J. Int. Robot. Syst.* 86 (2), 153–173. <http://dx.doi.org/10.1007/s10846-017-0468-y>.
- Qin, L., Peng, H., Huang, X., Liu, M., Huang, W., 2024. Modeling and simulation of dynamics in soft robotics: A review of numerical approaches. *Curr. Robot. Rep.* 5 (1), 1–13.
- Robinson, G., Davies, J.B.C., 1999. Continuum robots-a state of the art. In: *IEEE Robot. Autom. Lett., RA-L*, Vol. 4, IEEE, pp. 2849–2854. <http://dx.doi.org/10.1109/ROBOT.1999.774029>.
- Rus, D., Tolley, M.T., 2015. Design, fabrication and control of soft robots. *Nature* 521 (7553), 467–475. <http://dx.doi.org/10.1038/nature14543>.
- Shepherd, R.F., Ilievski, F., Choi, W., Morin, S.A., Stokes, A.A., Mazzeo, A.D., Chen, X., Wang, M., Whitesides, G.M., 2011. Multigait soft robot. *Proc. Natl. Acad. Sci.* 108 (51), 20400–20403. <http://dx.doi.org/10.1073/pnas.1116564108>.
- Thieffry, M., Kruszewski, A., Duriez, C., Guerra, T.-M., 2018. Control design for soft robots based on reduced-order model. *IEEE Robot. Autom. Lett. (RA-L)* 4 (1), 25–32. <http://dx.doi.org/10.1109/LRA.2018.2876734>.
- Wei, D., Zhou, J., Zhu, Y., Ma, J., Ma, S., 2023. Axis-space framework for cable-driven soft continuum robot control via reinforcement learning. *Commun. Eng.* 2 (1), 61.
- Wu, K., Zheng, G., 2021. Fem-based gain-scheduling control of a soft trunk robot. *IEEE Robot. Autom. Lett. (RA-L)* 6 (2), 3081–3088. <http://dx.doi.org/10.1109/LRA.2021.3061311>.
- Wu, K., Zheng, G., Zhang, J., 2022. FEM-based trajectory tracking control of a soft trunk robot. *Robot. Auton. Syst.* 150, 103961.
- Xiang, P., Zhang, J., Sun, D., Qiu, K., Fang, Q., Mi, X., Wang, Y., Xiong, R., Lu, H., 2023. Learning-based high-precision force estimation and compliant control for small-scale continuum robot. *IEEE Trans. Autom. Sci. Eng.*.
- Zhang, H., Cao, R., Zilberstein, S., Wu, F., Chen, X., 2017. Toward effective soft robot control via reinforcement learning. In: *Int. Conf. Int. Robot. Appl., ICIRA*, Springer, pp. 173–184.