Sim-to-Real Transfer of Co-Optimized Soft Robot Crawlers

Charles Schaff[2†], Audrey Sedal[1*†], Shiyao Ni[1] and Matthew R. Walter[2]

[1]Department of Mechanical Engineering, McGill University, 817 Sherbrooke St W, Montreal, H3A 0C3, QC, Canada .
[2]TTI-Chicago, 6045 S Kenwood Ave, Chicago, 60637, IL, USA .

*Corresponding author(s). E-mail(s): audrey.sedal@mcgill.ca;
Contributing authors: cbschaff@ttic.edu; shiyao.ni@mail.mcgill.ca; mwalter@ttic.edu;
†These authors contributed equally to this work.

This work provides a complete framework for the simulation, co-optimization, and sim-to-real transfer of the design and control of soft legged robots. Soft robots have "mechanical intelligence": the ability to passively exhibit behaviors that would otherwise be difficult to program. Exploiting this capacity requires consideration of the coupling between design and control. Co-optimization provides a way to reason over this coupling. Yet, it is difficult to achieve simulations that are both sufficiently accurate to allow for sim-to-real transfer and fast enough for contemporary co-optimization algorithms. We describe a modularized model order reduction algorithm that improves simulation efficiency, while preserving the accuracy required to learn effective soft robot design and control. We propose a reinforcement learning-based co-optimization framework that identifies several soft crawling robots that outperform an expert baseline with zero-shot sim-to-real transfer. We study generalization of the framework to new terrains, and the efficacy of domain randomization as a means to improve sim-to-real transfer.

1 Introduction

The deformable nature of soft robots enables designs that respond to contact or control inputs in sophisticated ways, with behaviors that have proven effective across a variety of domains. Design of these robots is tightly coupled with the policy that controls their motion, giving rise to a form of "mechanical intelligence" (Rus and Tolley, 2015) in which materials and mechanisms respond to their environment in useful ways that augment functionality, e.g., conforming to an object to create a better grasp or storing elastic energy to improve the power of a walking gait. Therefore, methods that jointly optimize both the robot's physical design and its control policy provide a promising approach to realizing mechanically intelligent soft robots. While there is extensive work on joint design-control optimization in the context of rigid robots (Sims, 1994; Park and Asada, 1994; Paul and Bongard, 2001; Paul et al, 2006; Spielberg et al, 2017; Seo et al, 2019; Digumarti et al, 2014; Ha et al, 2017; Zhao et al, 2020; Schaff et al, 2019; Ha, 2019; Chen et al, 2020; Pathak et al, 2019), less exists for soft robotics.

This work provides a complete framework for the simulation, co-optimization, and sim-to-real transfer of the design and control of modular soft robots for locomotion tasks. Integral to this framework, we propose a co-optimization algorithm that utilizes multi-task deep reinforcement learning to generate a design-aware policy capable of generalizing across the space of designs. The algorithm exploits this policy to quickly focus its search on high-performing designs. To encourage "mechanical intelligence", we learn an open-loop controller,

forcing complex behavior to be expressed through the resulting soft body.

An important prerequisite for contemporary co-optimization algorithms is a simulator that is both fast enough to explore a large set of design and control strategies and accurate enough to ensure that the learned robots are physically realizable and capable of sim-to-real transfer. However, modelling soft bodies is both challenging and computationally intensive. The best way to simulate soft bodies for robotics is an open question, and the few existing co-optimization approaches for soft robotics suggest different simulation strategies (Hu et al, 2019; Spielberg et al, 2021; Hiller and Lipson, 2014). These simulators have varying degrees of realism and their ability to produce soft robots that cross the reality gap is unclear.

With the focus of sim-to-real transfer in mind, we choose to employ finite element analysis (FEA), which is the defacto standard for simulating deformable materials with a high degree of accuracy. In this work, we show that FEA simulations, with enough computation, allow for the direct transfer of co-optimized soft robots. Yet, high-fidelity FEA simulations can be hundreds-of-times slower than real-time, rendering learning-based methods infeasible. In order to improve the computational complexity of FEA simulation while preserving its accuracy, we extend the recent work of Goury and Duriez (2018) that proposes a model order reduction technique for soft robotics in the open-source FEA simulation framework SOFA (Faure et al, 2012; Coevoet et al, 2017). While their method improves the computational efficiency of simulation, it incurs a large initial cost that prohibits learning over different designs. We propose a reconfigurable reduction framework that reduces a set of composable parts that can then be combined to create reduced order models of soft robots with varying morphologies. The proposed model-order reduction method can be tuned to higher or lower fidelity through reduction tolerances, resulting in a trade-off between the speed of simulation and the fidelity of the model.

Given the high sample complexity of modern data-driven optimization methods, the speed-fidelity trade-off becomes increasingly important. Our work compares the sim-to-real transfer of robot designs and policies learned at varying levels of reduced model fidelity.

We experimentally assess the effectiveness of domain randomization (Tobin et al, 2017) as an alternative to improving the realism (and, in turn computational cost) of the simulator, investigating whether it improves sim-to-real transfer in the case of a lower-fidelity model and whether it produces learned gaits that adapt to new terrains. Domain randomization has been shown to encourage the sim-to-real transfer of reinforcement learning frameworks in rigid robotics (Tan et al, 2018; Kleeberger et al, 2020; Ju et al, 2022). Yet, the physics of soft robots differs from rigid robots, making it unclear how well these strategies generalize. Soft robots' deformable bodies give them infinite degrees-of-freedom (DoF) (Wang and Chortos, 2022). While, differences between the simulator and environment would eventually be reflected as errors in a finite number of joint torques for a rigid robot, they may manifest in the form of a much wider number and variety of soft robot states. It has not yet been empirically demonstrated whether domain randomization can improve sim-to-real transfer, or add overall robustness, to soft robots.

While our approach is general, we focus our study on the easily manufacturable PneuNet actuator (Mosadegh et al, 2014), which has previously been used to create robots capable of walking and crawling gaits (Gamus et al, 2020; Shepherd et al, 2011). We experimentally validate our proposed approach by learning combinations of PneuNets and their controllers that together lead to faster gaits. Critically, we demonstrate the ability of our framework to successfully transfer optimized design-control pairs to reality on a variety of flat terrains that have high and low friction with the robot legs. Building off of our recent work (Schaff et al, 2022), these contributions are:

1. a model-free algorithm for optimizing the blended design and control spaces of soft robots;
2. a framework for creating reconfigurable reduced-order soft robot models that improve computational efficiency and enable the use of learning techniques;
3. a comparison study of reduced-order models at varying accuracy levels using domain randomization as an commonly employed means to encourage sim-to-real transfer;

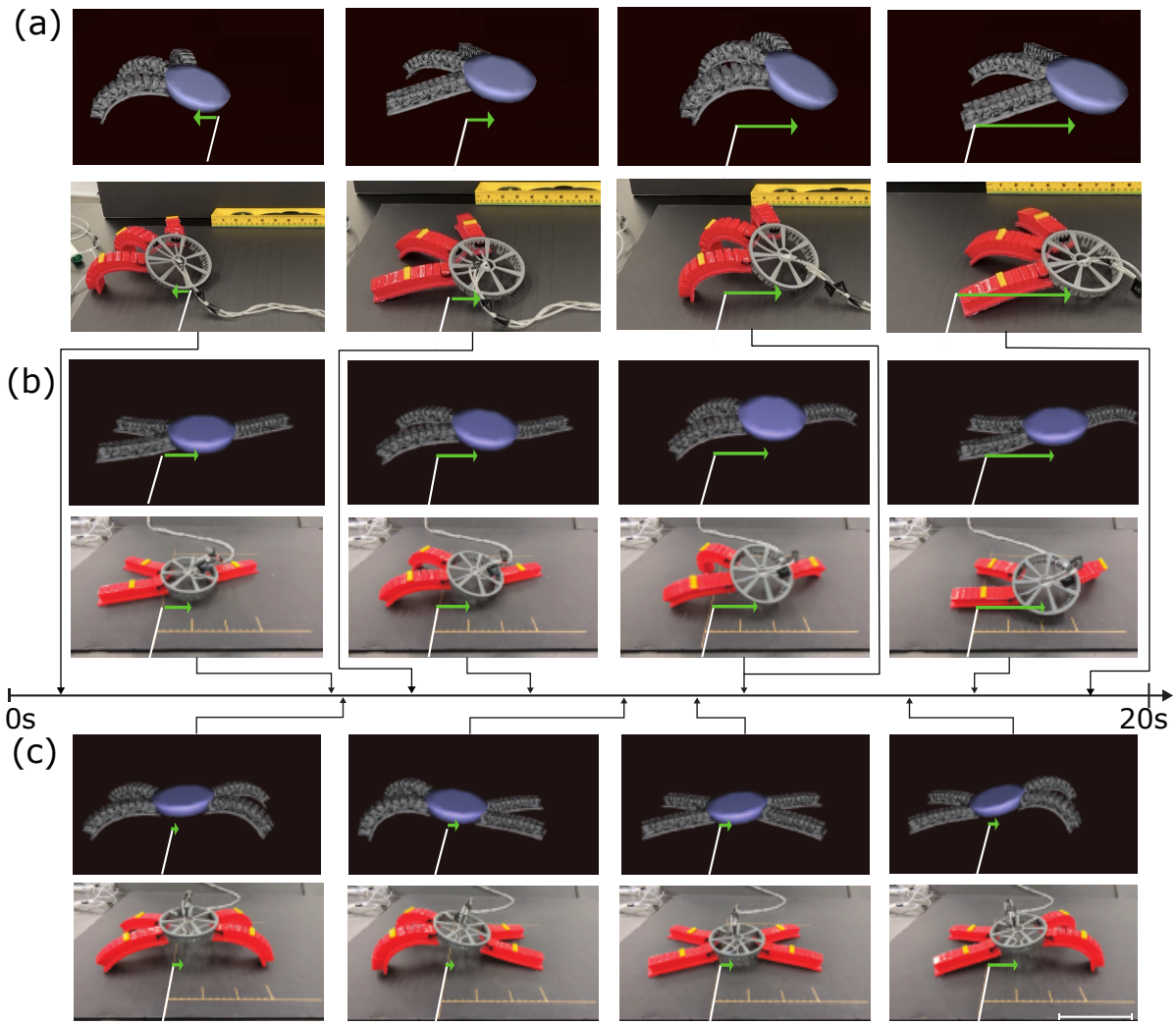


Fig. 1 Our framework jointly learns the design and control of crawling soft robots (top) that outperform an expert-designed baseline (bottom). While trained exclusively in simulation, our learned robots are capable of zero-shot sim-to-real transfer, with the optimal design moving more than 2× faster than the baseline in the real world. (a) Robot trained with τ_{hi}, μ_{rand}, Seed 0. (b) trained with τ_{low}, μ_{fix}, Seed 0. (c) Baseline: design and control developed by expert.

4. the discovery of pneumatically actuated soft robots that outperform a standard expert-designed crawling robot in both simulation and reality.

2 Related Work

The problem of jointly optimizing a rigid robot's physical structure along with its control has a long history in robotics research. Early work employs evolutionary methods to optimize the robot's design along with its (often neural at the time) controller (Lipson and Pollack, 2000; Paul and Bongard, 2001; Murata and Kurokawa, 2007; Bongard, 2011). Another common approach is to assume access to a parameterized model of the robot's dynamics and to then optimize these parameters together with those of control (or motion) (Paul et al, 2006; Villarreal-Cervantes et al, 2013; Ha et al, 2017; Spielberg et al, 2017; Geilinger et al, 2018; Taylor and Rodriguez, 2019;

Bravo-Palacios et al, 2020). Bolstered by the availability of efficient high-fidelity physical simulators, joint optimization methods based on reinforcement learning are able to learn capable rigid-body design-controller pairs without prior knowledge of the dynamics (Schaff et al, 2019; Pathak et al, 2019; Ha, 2019; Whitman et al, 2021). Because these methods are trained in simulation, the sim-to-real transfer of the learned solutions is unclear. Our work focuses on the use of high-fidelity simulators, which have been shown to facilitate transfer for rigid robots (Tan et al, 2018). Unlike rigid-body domains, however, achieving the level of fidelity necessary for soft-bodied robots typically requires computationally demanding simulators that prohibit learning-based co-optimization, a challenge that we address here.

Compared to rigid robotics, jointly optimizing the design and control of soft robots is less explored. Of the work that exists, the large majority focus exclusively on simulation. Many approaches reason over design and control spaces that include a mix of discrete and continuous parameters (e.g., voxel-based soft robots (VSRs) (Talamini et al, 2019) are composed of discrete voxels, but the input frequency to each voxel is considered to be continuous). Spielberg et al (2021) propose an autoencoder-based method that is able to optimize the placement of a large number of such voxels for simulated locomotion tasks with fewer iterations than other approaches. Cheney et al (2013) use an evolutionary neural strategy to develop designs for VSRs that locomote in simulation. Kriegman et al (2019) describe an approach to deforming the structure of VSRs subject to damage such that the original control policy remains valid. Ma et al (2021) use a material point method-based simulation and gradient-based optimization methods to co-optimize the shape and control of simulated swimming robots. Deimel et al (2017) use particle filter-based optimization to co-optimize finger angles and the grasp strategy of a soft gripper. The success of these methods in simulation is encouraging for soft roboticists, and recent simulation-based benchmarks allow for a rigorous comparison of co-optimization methods (Collins et al, 2021; Bhatia et al, 2021). However, existing work provides a limited evaluation of the physical design-control pairs, and so little is known about their ability to transfer to the real world. Indeed, experiments on voxel-based soft robots reveal that their behavior in simulation can differ significantly from reality (Kriegman et al, 2020). Difficulties in modelling friction and stick-slip behaviour (Majidi et al, 2013; Gamus et al, 2020) have been discovered to be a core reason that transfer is challenging.

Recent research in soft robotics has suggested techniques for closing the sim-to-real gap. Zhang et al (2022) describe a novel system identification technique for differentiable soft robot simulators. Dubied et al (2022) proposed a differentiable finite-element model for soft robotics that they verify experimentally on canonical mechanical problems (e.g., a cantilevered beam). Unlike our work, however, these approaches each deal with a fixed design rather than a space of designs. One notable exception, Morzadec et al (2019) experimentally verify an optimized soft robotic joint, showing how shape optimized using a finite element analysis-based simulator (Coevoet et al, 2017) translates to improvements in a real-world soft robotic leg. However, unlike our work, they do not optimize the controller. Another exception is recent work that integrates a pneumatic-based passive controller into the robot's design to achieve a forward walking gait (Drotman et al, 2021), providing an example of how soft robots can have unclear boundaries between design and control. In rigid robotics, Tobin et al (2017) demonstrate that applying domain randomization to a policy's visual inputs in simulation enables the policy to transfer directly to the real world in a zero-shot fashion. Tan et al (2018) employ domain randomization over the parameters of the dynamics model to transfer the control policy for a quadruped robot trained in simulation to the real world. Similar approaches have been used to facilitate sim-to-real transfer for a variety of robot learning domains (Kleeberger et al, 2020; Ju et al, 2022). Comparatively, again, the applicability of domain randomization to sim-to-real transfer in soft robotics (with its higher-DoF systems and the wider variety of relevant physical phenomena), is not as well explored. Centurelli et al (2022) show that a policy trained on randomly generated trajectories controls a dynamic soft robot arm. Li et al (2022) performs domain randomization by adding noise to both actions and observations, demonstrating control of a soft arm in path-following. Tiboni et al (2023) randomize the soft robot body material's Young's modulus

and Poisson Ratio, demonstrating (in simulation) control of a soft arm and a crawling robot similar to the expert baseline here.

Meanwhile, individual design and control methods continue to be key areas of research in soft robotics (Rus and Tolley, 2015). There exist a wide variety of design concepts for soft robots (Chen and Wang, 2020) such as fluidically pressurizeable soft devices (Mosadegh et al, 2014; Shepherd et al, 2011), metamaterial-based designs (Rafsanjani et al, 2018; Lipton et al, 2018), and cable-driven devices (Bern et al, 2020). Soft roboticists note that existing design optimization methods for compliant, nonlinear mechanisms, such as topology optimization, are challenging to use in soft robotics due to complicated soft material behavior (Chen and Wang, 2020). The diversity of the design space for soft robots further exacerbates the challenge of automating the search for optimal designs (Pinskier and Howard, 2021).

Model- (Bern et al, 2019; Bruder et al, 2019) and learning-based (Lee et al, 2020; Culha et al, 2020; Kim et al, 2021) controllers have also proven successful, as well as hybrid policy designs (Vitanov et al, 2020; Bern et al, 2020; Howison et al, 2020). Zhu et al (2019) consider an origami-like robot with various design configurations that all inform policy optimization, and Morimoto et al (2021) employ the soft actor-critic algorithm (Haarnoja et al, 2018) for reaching tasks. Related, Vikas et al (2016) present a modular approach to designing 3D-printed motor-tendon soft robots that can be readily fabricated, and a model-free algorithm for learning the corresponding control policy. Unlike our framework, however, they do not jointly reason over design and control.

3 Co-Optimization Algorithm

We first describe the general approach to jointly (co-)optimizing robot design and control, and then discuss a specific application to crawling soft robots. Algorithm 1 and Figure 2 give an overview of this approach.[1]

[1]A publicly available implementation of our joint optimization framework is available at https://github.com/cbschaff/evolving-soft-robots.

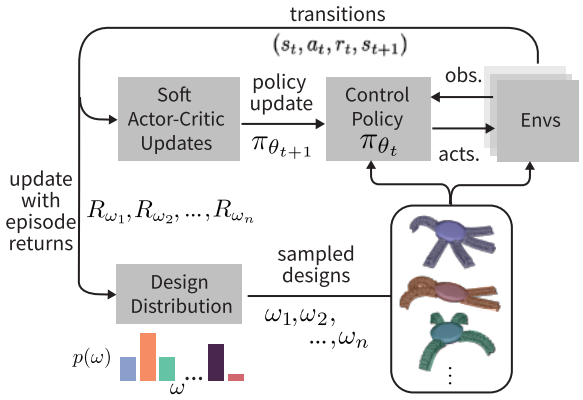


Fig. 2 Our approach maintains a distribution over designs $p(\omega)$. At each iteration, the method samples a set of designs $\omega_1, \ldots, \omega_n$ and controls each using a shared, design-conditioned, policy π_θ. We train the policy using soft actor-critic on a mixture of data from different designs, and update the design distribution based on the episode returns of the sampled designs.

3.1 General Approach via Multi-task Reinforcement Learning

We model the control problem as a Markov decision process (MDP) $\mathcal{M} = \text{MDP}(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R})$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ is the transition dynamics, and $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ is the reward function. When co-optimizing design and control, we additionally define the design space Ω. Assuming we are optimizing for a single task specified by its reward $\mathcal{R}$, we define the design-specific MDP $\mathcal{M}_\omega = \text{MDP}(\mathcal{S}_\omega, \mathcal{A}_\omega, \mathcal{P}_\omega, \mathcal{R})$ for each design $\omega \in \Omega$. In most co-optimization settings with a single task, the state and action spaces will change between designs only when those designs have different morphologies.

Let $\pi^*_\omega : \mathcal{S}_\omega \times \mathcal{A}_\omega \to [0, 1]$ be the optimal policy for MDP $\mathcal{M}_\omega$. The goal of co-optimization is to find the optimal design and controller pair $(\omega^*, \pi^*_{\omega^*})$ such that:

$$\omega^*, \pi^*_{\omega^*} = \arg\max_{\omega, \pi_\omega} \mathbb{E}_{\pi_\omega}\left[\sum_t \gamma^t \mathcal{R}_t\right] \quad (1)$$

In this setting, we are faced with many MDPs that share common structure. Solving each MDP independently is intractable and ignores similarities in morphology between designs. We draw

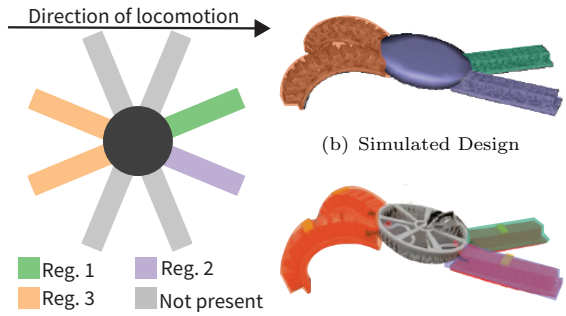
Algorithm 1 Joint Optimization of Design and Control

1: Initialize $\pi_\theta(a|s,\omega)$, $p(\omega)$, $T = 0$
2: **while** $T <$ BUDGET **do**
3: Sample designs $\omega_1, \omega_2, \ldots, \omega_n \sim p_\phi$
4: Control $\omega_1, \omega_2, \ldots, \omega_n$ with π_θ for t timesteps. Add transitions to replay buffer.
5: Update θ using soft actor-critic.
6: Update $R_{\omega_1}, R_{\omega_2}, \ldots, R_{\omega_n}$ with their obtained returns.
7: Set timestep $T = T + nt$
8: Set β_T to match entropy target $\mathcal{H}_T$.
9: Set $p(\omega) = \frac{e^{\beta_T R_\omega}}{\sum_\Omega e^{\beta_T R_\omega}}$
10: **end while**

on insights from multi-task reinforcement learning (Varghese and Mahmoud, 2020) to more efficiently solve for the optimal design-control pairs (Eqn. 1) by exploiting this common structure. Similar to goal-conditioned policies, our approach learns a single *design-conditioned* policy $\pi_\theta : \mathcal{S} \times \mathcal{A} \times \Omega \to [0, 1]$ to control all the designs in Ω for the specified task. We proposed this idea in the context of co-optimization (Schaff et al, 2019), and it has also been used for the sub-problem of controlling a set of designs with different morphologies (Huang et al, 2020; Kurin et al, 2021). This policy can be trained using any RL algorithm on a mixture of data collected with designs in Ω.

In order to search over designs, we maintain a distribution $p(\omega)$ over the design space Ω. This distribution generates designs for training the controller and models the belief about which designs are optimal given the current design-conditioned control policy. At the start of training, $p(\omega)$ should provide a large diversity of designs and then, once the controller has been sufficiently trained, slowly concentrate probability mass around high-performing designs. The controller can then specialize to an increasingly promising subset of designs until the algorithm converges on a single design and a controller that is then fine-tuned for that design.

The design distribution can be modelled in a number of different ways depending on the nature of the design space. For example, Schaff et al (2019) use a mixture of Gaussians for a continuous design space and shift the distribution towards high-performing designs in a manner analogous to



(a) Design Space (b) Simulated Design (c) Real-world Design

Fig. 3 A visualization of (a) our design space that consists of a disk with $N = 8$ candidate locations for pneumatic actuators, each of which can be connected to one of $M = 3$ pressure regulators. On the right are examples of a (b) simulated and (c) real-world design, where colors denote the pressure regulator for each actuator. The forward direction is to the right.

a policy gradient update. In this work, we assume that the design space is discrete and that the number of designs is practically enumerable, and thus employ a categorical distribution. Following the principle of maximum entropy, we model $p(\omega)$ as a Gibbs distribution:

$$p(\omega) = \frac{e^{\beta R_\omega}}{\sum_{\omega \in \Omega} e^{\beta R_\omega}}, \tag{2}$$

where R_ω is the most recent reward obtained by design ω, and β is an inverse temperature parameter used to control entropy. At each point in training, we set β to maintain a decaying entropy target. Specifically, we set $\beta = 0$ to specify a uniform distribution for an initial training period, and then decay entropy according to a linear schedule. This schedule is akin to removing a constant fraction of designs from the search space at each step during training.

3.2 Application to Soft, Legged Robots

The design space that we study here (Fig. 3) consists of a disk with N equally spaced positions where soft, pneumatic actuators can be positioned radially outward. Each actuator is connected to one of M different pressure regulators. Designing the robot then amounts to choosing whether (or not) to place an actuator at each of the N locations and, for each placed actuator, connecting

it to one of the M pressure regulators. Our specific implementation considers $N = 8$ candidate locations and $M = 3$ regulators, and restricts the design to having between three and six actuators. This results in a total of 41202 unique designs, which can be reduced to 6972 by exploiting symmetry in the regulator assignments (e.g., before training the control policy, designs that involve different assignments of three regulators to the same set of three legs can be considered to be the same). Each actuator is a PneuNet (Mosadegh et al, 2014) which, like similar soft actuators, has been combined to achieve crawling gaits (Goury and Duriez, 2018; Shepherd et al, 2011; Gamus et al, 2020; Vikas et al, 2016), providing a well-studied baseline.

While our approach is compatible with any RL algorithm, we use the standard soft actor-critic (SAC) algorithm because it offers stable and data-efficient learning dynamics. We train an open-loop controller modeled as a feed-forward neural network for the task. This simplification of the controller forces the design to perform "morphological computation" (Rus and Tolley, 2015) to enable intelligent behavior. The policy takes as input the design parameters along with the four most-recent actions and outputs pressure targets for each regulator.

4 Reduced-Order Model

Integral to our approach, we propose a new modular form of model order reduction (MOR) suited to co-optimization that significantly improves the efficiency of FEA-based simulators while preserving their accuracy, supporting sim-to-real transfer. Fidelity of the reduced-order model (ROMs) is set by selecting tolerances during the reduction procedure, enabling a variety of candidate ROMs that trade between compute speed and accuracy. This feature enables a comparison of learned robots in both high- and low-fidelity simulators.

4.0.1 Reduction through Snapshot POD and Hyperreduction

The finite element method provides an approximate numerical solution to partial differential equations (PDEs) by discretizing space into a mesh consisting of a set of finite elements. Often,

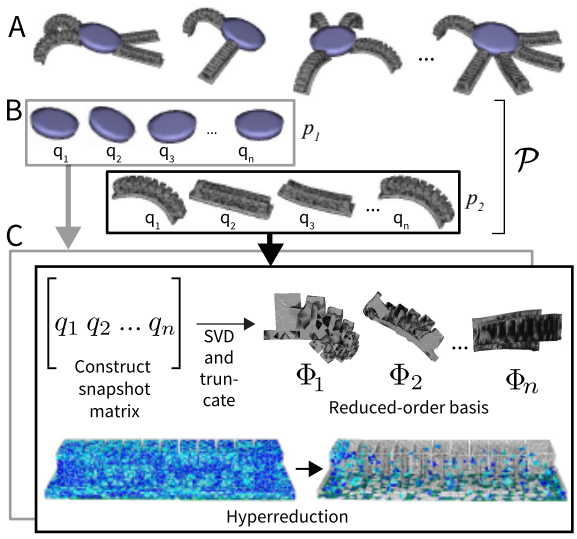


Fig. 4 Our proposed technique for model order reduction that is compatible with co-optimization. **A**: Sample candidate designs and record their animations. **B**: Collect parts across designs and transform them into a common reference frame. **C**: For each part, perform a snapshot POD reduction and hyperreduction to obtain a reduced-order basis for motion, and reduced integration domain. We create reductions for new designs by combining reductions of their parts.

dense meshes (and subsequently, large amounts of computation) are needed.

The soft actuator mesh contains nodes with position q_{t_n} and velocity v_{t_n} at discrete time step t_n. At each t_n, simulation requires solving a discrete form of Newton's second law (Goury and Duriez, 2018):

$$A(q_{t_n}, v_{t_n})dv = b(q_{t_n}, v_{t_n}) + H^\top \lambda, \qquad (3)$$

where $dv = v_{t_{n+1}} - v_{t_n}$, $A \in \mathbb{R}^{d\times d}$ collects inertial and internal forces, $b \in \mathbb{R}^d$ contains terms from internal and external forces, and $H^\top\lambda \in \mathbb{R}^d$ collects constraints (e.g., associated with contact with the floor), with d being the number of degrees-of-freedom in the mesh. When using dense meshes for accurate simulation, constructing the matrix A and solving this system of equations are often the main bottleneck in FEA simulations.

We first reduce the system dimension through snapshot proper orthogonal decomposition (POD) Weiss (2019), a common method for models in fluid mechanics and other nonlinear physical environments. Using the methods of Goury and Duriez (2018), we find a low-dimensional

subspace Φ that well-approximates the space of possible motions and deformations while reducing the order through a Galerkin projection onto Equation 3:

$$\Phi^\top A(q_{t_n}, v_{t_n})\Phi d\alpha = \Phi^\top b(q_{t_n}, v_{t_n}) + \Phi^\top H^\top \lambda. \quad (4)$$

We achieve this by recording 'snapshots' of the position q_t of the mesh throughout a series of predefined motions that try to cover the space of common deformations.

The simulation then uses lower-dimensional coordinates α, with $q_{t_n} = q_0 + \Phi\alpha_{t_n}$. This formulation has a truncation error ν which is a function of p (the order of α), n_q (the order of the initial finite-element model) and the singular values σ_i:

$$\nu = \frac{\sum_{i=p+1}^{n_q} \sigma_i^2}{\sum_{i=1}^{n+q} \sigma_i^2}. \quad (5)$$

We set the maximum error ν as a tolerance in the snapshot POD algorithm. Though snapshot POD reduces the time to solve Equation 3, it still requires computing the high-dimensional matrix A at every time step. We therefore perform a hyperreduction to further approximate A by predicting its entries from the contributions of a 'reduced integration domain' consisting of a small number of elements. We use the hyperreduction method of energy conservation sampling and weighting (ECSW) (Farhat et al, 2014). This method iteratively solves a non-negative least squares problem until a desired tolerance τ, reflecting error in force between the full and the reduced integration domains, is reached. For further details regarding this two-part method and a demonstration in soft robotics, we refer the reader to Goury and Duriez (2018).

4.0.2 Modularized Reduction for Design-Reconfigurability

The MOR method described in the previous section uses a single fixed mesh. For high-dimensional soft robot design spaces, separately reducing each design embodiment is computationally intractable. Instead, we define a modularized design space: each design $\omega \in \Omega$ is defined as a combination of a small set of fixed parts $\mathcal{P}$.

We then reduce each part in $\mathcal{P}$ using the method of Section 4.0.1 independently and combine the parts in arbitrary ways to form new designs. The number of times we perform MOR is then of the same size as $\mathcal{P}$ rather than the size of Ω.

MOR on the modularized design space only well-approximates the full-order model when the computed subspace Φ^p for each part $p \in \mathcal{P}$ is close to all frequently achieved deformations. Because designs will deform in different ways, it is necessary to include 'snapshots' of motions from a large set of designs to achieve high-quality reduced-order models.

Therefore, careful snapshot selection on each module in $\mathcal{P}$ is crucial and inaccuracies may be exploited during optimization, resulting in invalid design-control pairs.

We achieve high-quality reduced-order models for each part by collecting snapshots from a heuristically chosen subset of Ω and animating those designs by cycling through the pressure extremes of each actuator.

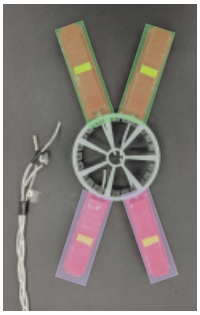
When constructing the reduced basis for new designs, we transform the basis Φ_p to match the initial pose (t_i, R_i) of each part p_i by rigidly rotating the node positions that make up each basis vector:

$$\Phi_j^{p_i} = \begin{bmatrix} R_i\Phi_j^p[0:3] & R_i\Phi_j^p[3:6] \\ & \cdots R_i\Phi_j^p[n-3:n] \end{bmatrix}, \quad (6)$$

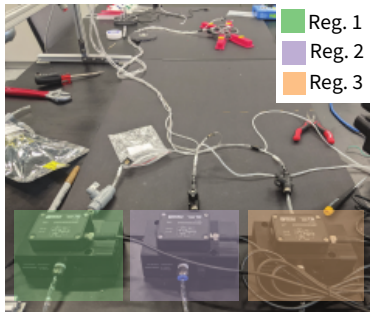
where $\Phi_j^p \in \mathbb{R}^n$ is the j^{th} column of Φ^p and $\Phi_j^p[k:l]$ is a slice of that vector from index k to index l. We ignore the initial translation t_i because translation basis vectors are included in Φ_p. Figure 4 gives an overview of this approach. This method is generalizeable to any modularly constructed soft robot.

4.1 Model-Order Reduction of Crawling Soft Robots

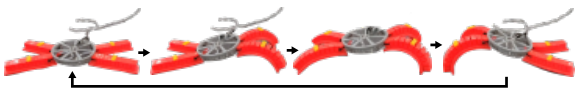
We apply this reduction technique to our design space of crawling soft robots. Our designs are composed of two parts: the central disk, and some number of identical PneuNets. Therefore, the above approach allows us to perform two reductions (one for each part) as opposed to reducing each of the 6972 designs in our design space. We found that a sparse disk mesh was sufficiently fast and accurate for simulation and we therefore only reduce the PneuNet.



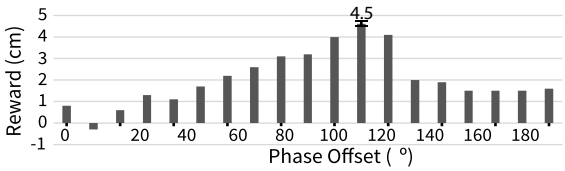
(a) Baseline Design



(b) Experimental Setup



(c) Baseline Gait



(d) Reward vs. Phase

Fig. 5 Our baseline consists of (a) an expert-designed soft robot with four legs, where the fore and hind legs are attached to pressure regulators one and two, respectively. The experimental setup consists of (b) three pressure regulators, a crawling surface, and 3.15 mm outer-diameter tubing connected to the robot (in the the distance). (c) Snapshots of the expert-designed gait. (d) Reward (distance travelled) obtained by an offset-sine gait with different phase shifts. A phase difference of 110° achieves the highest reward.

For the reduction of the PneuNet, we select a heuristic set of 256 designs for which we collect snapshots. Each design contains a unique subset of the eight potential PneuNet positions, and each PneuNet is controlled independently. Similar to Goury and Duriez (2018), we iterate through the extremes of each actuator and record snapshots at fixed time intervals. This can be seen as a walk through the vertices of an n-dimensional hypercube, where n is the number of PneuNets present. Details can be found in the Supplementary Material.

5 Experiments

We test our approach by attempting to find a design and open-loop controller that crawl as far as possible on a flat plane in a 20 second episode. We define reward as the distance traveled in the (forward) x-direction (in cm) as measured at the center of the disk. See Supplementary Material for more details on the experimental setup, including a full list of simulation and learning parameters.

5.1 Simulation Environment

After performing model order reduction, we carry out FEA simulation using the SOFA Finite Element framework (Faure et al, 2012) with the soft robotics (Coevoet et al, 2017) and model order reduction (Goury and Duriez, 2018) plugins. We model the PneuNet legs (including the inflatable and constraint material) and central disk as linear elastic materials. We estimated the Young's modulus of the PneuNet material (Smooth-On DragonSkin 30) based on the published Shore hardness together with the method of Qi et al (2003). We modeled the constraint layer of the PneuNet as being linear elastic with a Young's modulus twice the magnitude of the inflatable material. We tuned the Poisson's ratio in order to maintain numeric convergence and qualitative realism. We used Coulomb friction as the friction model, with $\mu_{fix} = 1.2$ for environments without domain randomization and $\mu_{rand} \in \{0.65, 0.7, 0.75..., 1.3\}$ for training with domain randomization. To account for additional possible unmodelled effects, we measure deformation of a single, real PneuNet under fixed pressures, find the corresponding pressures that results in the same deformation of the simulated PneuNet, and fit an affine function to this data. Pressures commanded by our learned policies are then mapped through this function to ensure a simulated response similar to that of the real PneuNets. We find that this step facilitates sim-to-real transfer. We select two reduced-order models to create separate training environments. One 'higher fidelity model' having $\nu = 0.0032$, $\tau = 0.001$ and a second 'lower fidelity model' having $\nu = 0.0032$, $\tau = 0.0032$. We avoid the lowest-accuracy/highest-tolerance models (e.g., $\tau \geq 0.01$) to maintain numeric stability. Each reduced model may be run for any robot design in the space, with wide variety of possible physical parameters. See Supplementary Material for details on the model-order reduction implementation.

5.2 Baseline

We designed a baseline design-controller pair similar to the robot used by Shepherd et al (2011). The baseline (Fig. 5) has two fore legs and two hind legs placed 45° apart with each pair controlled by a single regulator. Based on recent analysis of inching gaits (Gamus et al, 2020), we constrain each pressure regulator to produce a sine wave of equal amplitude and period. We achieve forward motion by imposing a phase shift between the sine waves for the fore and hind legs. We select a pressure range of 0 to 90 kPa to avoid both physical instabilities (i.e., aneurysms of the PneuNets) and numerical instabilities in the FEA simulator. We use the maximum amplitude allowed in this range of 45 kPa and choose a period of 4 seconds, which is the fastest period that led to stable motion. The optimal phase shift depends heavily on friction (Gamus et al, 2020; Majidi et al, 2013) so we conducted experiments on one surface with different phase shifts between 0° and 180° in increments of 10°, and chose the value that resulted in the highest reward. Figure 5 shows the effect of phase shift on the reward.

5.3 Training and Domain Randomization

We use 96 parallel environments for data collection. Each environment contains a design sampled from $p(\omega)$ (Algorithm 1, line 3) that is controlled with the current policy for one episode (Algorithm 1, line 4). The control policy is then updated using the soft actor-critic algorithm on data from a replay buffer (Algorithm 1, line 5). We repeat this process for 1M environment timesteps during which we fix the design distribution to be uniform for the first 200K timesteps, after which we linearly decay the entropy to zero at 1M timesteps.

In total, we perform training 14separate times. First, we train the low-tolerance ($\tau_{lo} = 0.001$) and high-tolerance ($\tau_{hi} = 3.2 \times \tau_{lo} = 0.0032$) models without domain randomization. We used two seeds ('Seed 0' and 'Seed 18' in dataset) for policy parameter initialization, giving 4 results. Then, we perform domain randomization using the coefficient of Coulomb friction μ modelled in the simulator. We sample the Coulomb friction parameter $\mu_{\text{rand}} \in \{0.65, 0.66, \ldots, 1.3\}$ as a uniform distribution for each training episode. We perform this domain randomization for both the high-accuracy model ($\tau = 0.001$) and the low-accuracy model ($\tau = 0.0032$). For each of these, we trained 3 times using 'Seed 0' and once using 'Seed 18' for policy parameters, each with three separate random seeds for the value of μ.

To find compare the effects of optimization over the joint design-control space with optimization of the controller only, we performed training runs using 'Seed 0' with the baseline design and the co-optimized design (originally trained with τ_{low}, μ_{fix}, and Seed 0).

5.4 Transfer Evaluation

After training, we manufacture several of the highest-performing designs and conduct a series of experiments to evaluate the sim-to-real transfer of the learned design-control pairs. For details about the fabrication of our robots, see the supplementary material.

We evaluate sim-to-real transfer error by measuring reward (i.e., the center of the robot's forward progress at the end of the episode) in simulation and in the physical experiment. For each of the four training settings, we test the final learned robot on various terrains. For the high-fidelity, low-tolerance ($\tau_{\text{lo}} = 0.001, \mu_{\text{fix}}$) training environment, we manufacture the top five unique robot designs obtaining the highest simulation reward, evaluating them on the foam board. For the other training environments, we manufacture only the final learned robot. We evaluate the ability of learned design-policy pairs to transfer to new conditions.

We evaluate transfer to new surface friction levels by testing each final learned robot from each training environment on a variety of surfaces: a silicone soldering mat ('Mat', Amazon.com), paper-faced foam board ('Card', Amazon.com), and a plastic laboratory table ('Table', Knoll Furniture). We run each design and its corresponding open-loop gait for an episode period of 20 sec, for five trials. The silicone soldering mat was short and resulted in the robot with trained $\tau = 0.001$, $\mu = 1.2$ going off of the mat. We estimate μ of each of these surfaces by measuring the horizontal force (using a spring scale) when dragging the robot across the surface with full contact and constant speed. Parameters are given in Table 5.4. In

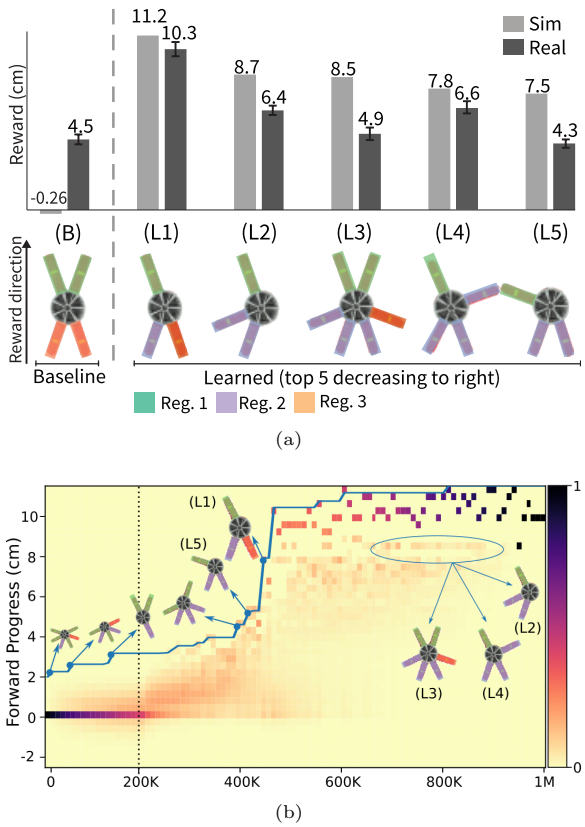


Fig. 6 (a) Reward comparison with expert baseline of top five designs L1–L5 learned with τ_{lo}, μ_{fix}.(b) Histogram of rewards throughout training, weighted by the design distribution. The blue line represents the highest reward achieved at each point throughout training. Labelled designs show important mode switches.

addition to transfer evaluation, we evaluate sim-to-sim transfer of the robots learned with τ_{hi} by running them in the τ_{low} environment.

Table 1 Measured Coulomb friction coefficients μ of each tested surface. Uncertainty arises due to resolution of the spring scale and fluctuations in the dragging speed.

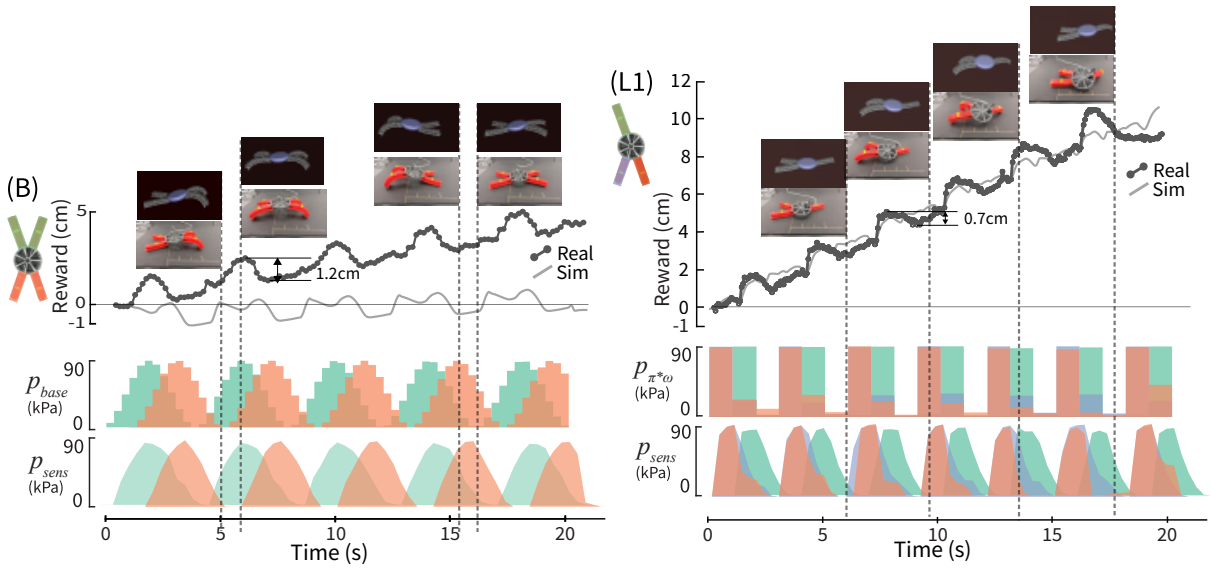
Surface	μ
Mat	0.60 ± 0.1
Board	1.44 ± 0.1
Table	2.64 ± 0.1

6 Results

First, on the foam board surface, we examined the performance and capacity for sim-to-real transfer of the top-five learned design-control pairs that our framework discovers along with our baseline design-control pair. We refer to the baseline as "B" and the learned pairs as "L1–L5" in order of decreasing reward.

Figure 6 visualizes these designs and compares their rewards to that of the baseline in simulation and in the real world, where we report the mean and standard deviation from five trials. The top four learned robots (L1–L4) outperform the baseline in both simulation and the real world, while the fifth robot (L5) performs comparably to the baseline in the real-world experiments. The top learned robot, L1 outperforms the baseline (in the real world) by a factor of 2.3. We emphasize that the baseline gait parameters were optimized through real-world experiments, whereas the learned design-control pairs were optimized in simulation (though, using parameters fit from real life).

In an effort to get a better understanding of the behavior of our approach to design-control optimization, we sample a set of designs from our design distribution during training and evaluate each sample with the corresponding design-conditioned policy at that point during training. Figure 6 provides a visualization of these training dynamics a set of histograms over the rewards achieved by the design distribution throughout training. The figure depicts the top-performing design at various points during training. In the beginning of training (0–200K timesteps), we constrain the design distribution to be uniform, and we see that nearly every design achieves zero reward. Starting at 200K timesteps, the algorithm constrains the distribution with a linearly decaying entropy, after which the algorithm specializes to high-performing designs (i.e., L5). Approximately halfway through training, the algorithm converges on design L1, which achieves a reward that travels more than 2 cm farther in the 20 second episode than the next-best design-control pair (L5). During the remainder of training, the algorithm refines the control policy for L1, which remains the best-performing design, while the histogram reveals three other designs (L2, L3, and L4) that make reasonable forward progress. Robots L1 and L4 achieve the lowest sim-to-real transfer error. Detailed analysis showing error accrual over time throughout various gait phases is shown in Figure 8.



(a) Baseline Policy

(b) Learned Policy with Highest Reward

Fig. 7 A comparison between the (a) baseline (B) and (b) highest-performing learned robot (L1) in terms of the reward (distance traveled in cm) achieved in simulation and reality for the duration of the 20 sec episode, along with the corresponding control policy in terms of commanded and sensed pressures.

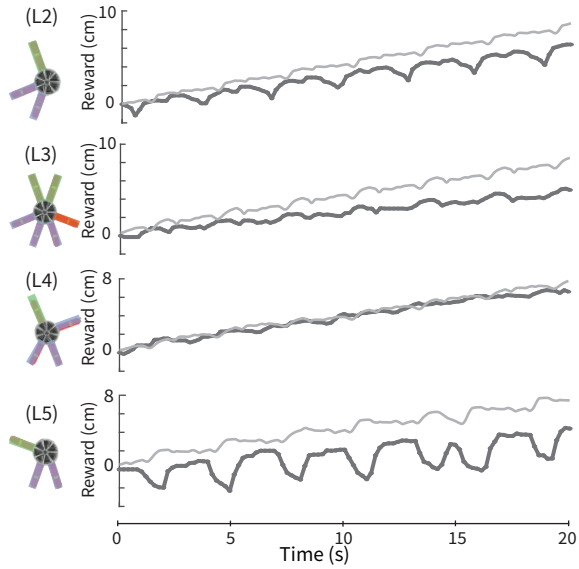


Fig. 8 Reward per timestep achieved in simulation and reality by the second- to fifth-ranking learned design-controller pairs L2–L5.

Observing the learned design-control pairs in simulation and through real-world experiments reveals that they exploit changes in frictional forces in clever ways to create forward motion. Figure 7 compares the open-loop (pressure) gaits

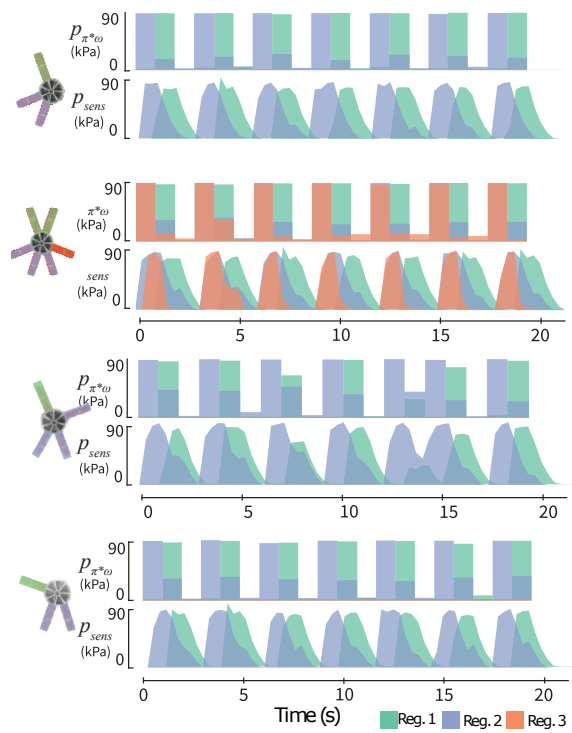


Fig. 9 From top: Commanded and sensed pressures $p_{\pi*\omega}$ and p_{sens} for the second- to fifth-ranking learned design-controller pairs L2–L5 trained with $\tau_{\text{lo}} = 0.001$ and $\mu_{\text{fix}} = 1.2$. All other gaits and pressure logs from each randomized trial are found in the dataset.

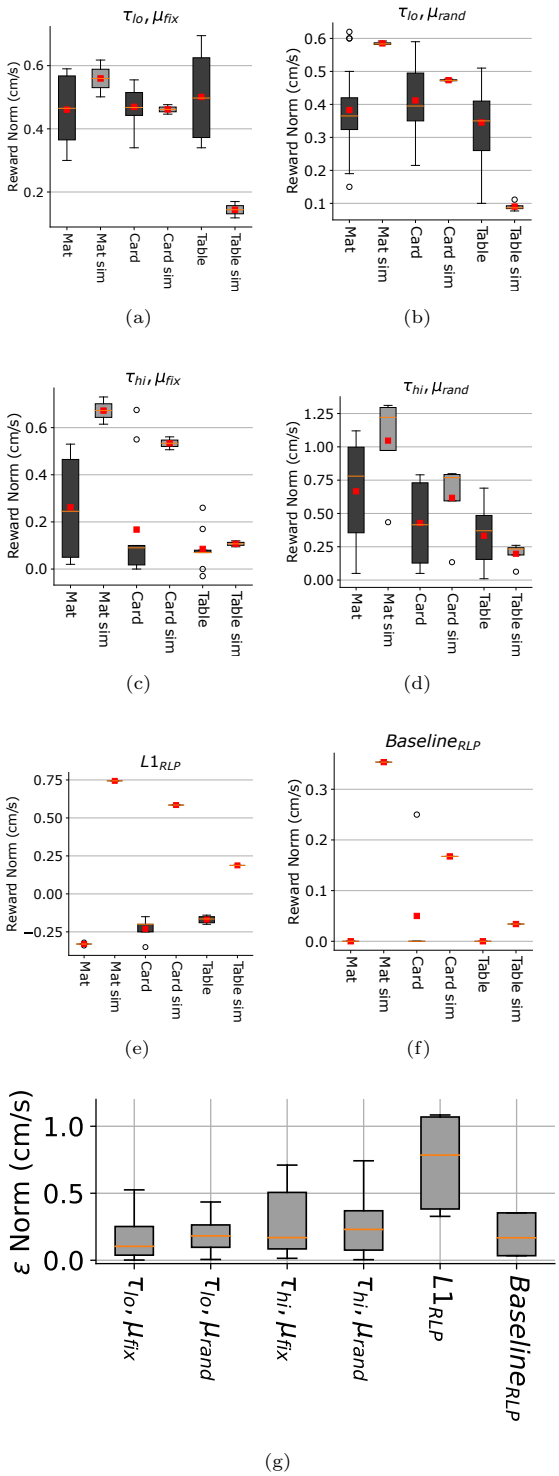


Fig. 10 (a)-(f) Sim and real reward for final trained robots in the four simulation settings $\tau_{lo/hi}$ and $\mu_{fix/rand}$; and the L1 robot design and the expert baseline design with trained policy only. Mean is red square; median orange line; IQR shaded box. (g) Mean normalized sim-to-real transfer error for same six robots.

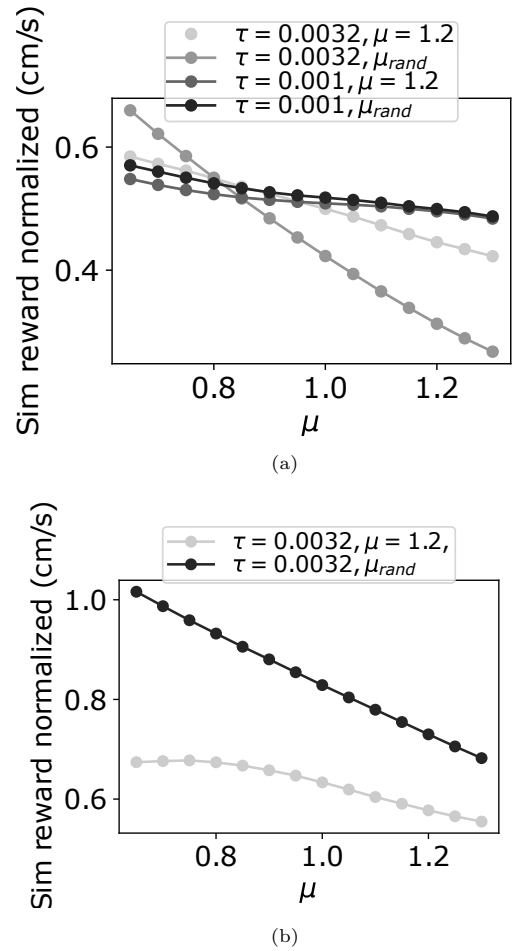


Fig. 11 (a) Mean normalized reward at $\mu = \{0.65, 0.7, ..., 1.3\}$ for all co-optimized robots in τ_{low} environment. (b) Same mean normalized reward for robots co-optimized with τ_{hi}, evaluated in τ_{hi} environment.

and reward trajectories in simulation and the real-world over the duration of the 20 sec episode for the baseline and top learned design-control pairs. The baseline robot (B) uses a symmetric design that moves forward through out-of-phase actuation of the fore and hind legs (Fig. 7(a)). In contrast, robot L1 uses an asymmetric design alongside out-of-phase actuation of the three attached pressure regulators. The result of this design and gait is a pivoting behavior visible that is visible in the photos within Figure 7(b) and the supplementary summary video. The robot trained with τ_{high}, μ_{rand}, Seed 0 also uncovered a novel gait. This three-legged gait, shown in 1(a), earns the highest reward by actuating the back legs in a fast,

out-of phase sequences that pushes the disk forward. Videos of all trained gaits in simulation and reality are available in the data set accompanying this paper.

Figure 7(a) compares the forward progress of the simulated ($\tau = 0.001$) baseline against its real-world counterpart. While the robot makes forward progress in both simulation and the real world, error accrues at the phase of the gait in which both the fore and hind legs are bent (e.g., between 5 sec and 6 sec in Figure 7(a)). Ending the simulation while the robot is backward-leaning results in the net distance travelled being negative. However, had the episode run for additional gait cycles, one would see forward progress in simulation.

The front PneuNets exhibit little slip in simulation, but they maintain less grip in the real world. The resulting error in the backsliding phase of each of these gaits then accumulates with every step, resulting in less forward motion than the robots achieve in simulation.

Figure 9 shows the policy output and measured internal pressure for the L2–L5 robots with τ_{lo}, μ_{fix} (top four rows) and the final learned robots trained with τ_{lo}, μ_{rand}, τ_{hi}, μ_{fix}, and τ_{hi}, μ_{rand} (fifth, sixth and seventh rows of Fig. 9 respectively). Learned gaits all operate in similar stages to L1: the hind legs inflate first to anchor the robot and subsequently inflate the fore leg. One exception is the τ_{hi}, μ_{rand} robot which takes five steps without inflating the front leg at all. Interestingly, each of the learned gaits inflate the hind legs before the fore legs, while the baseline does the opposite. Further, all robots leverage asymmetry. While the morphology of robot L3 is symmetric, the use of regulator three on only one leg adds asymmetry to its gait.

The gait trained with τ_{hi}, μ_{rand} shows irregular actuation of the front leg. We hypothesize that a strategy of taking different types of steps across the episode might result in improved robustness to contact friction, as some subset of the steps might be successful for any given surface. For gaits L1–L5, we measured pressure at a higher frequency during experiments compared to the other experiments for which we measured pressure once per policy action. This difference accounts for the change of the resolution of p_{sens} curves in Figure 9.

Because frictional error accrues over time, we normalize the reward by episode length (20 sec) for the following transfer error analyses. We evaluate the effect of MOR tolerance τ and friction domain randomization on task performance and transfer error. Figure 10(a-d) show real and simulation reward normalized by episode length for each robot. In reality, error accrues over five separate tests in which robots may reach differing rewards. Precise values of normalized reward and transfer error can be found in the Supplementary Material.

Robot L1 trained with τ_{low}, μ_{fix} and the final robot trained with τ_{low}, μ_{rand} achieve similar reward in simulation. Yet, the former has better transfer performance to reality than the latter. The strategy adopted in the latter case (τ_{lo}, μ_{rand}) takes fewer, longer steps. This reduces the influence of friction but likely introduces sensitivity to inertial perturbations (e.g., external moments caused by the tubes).

The highest overall reward in simulation is achieved under training with τ_{hi}, μ_{rand}. However, there is a gap of greater than 0.25cm/s betweent he average sim and real rewards. In only one case (robot co-optimized with $\tau_{high}, \mu_{rand}, S0$) does the reality match the simulation within 1.5cm for the 20s episode.

Figure 10(e-f) show simulated and real reward for policies trained on fixed designs (the L1 design and the four-legged expert baseline design). Since only one random seed was used in these training sets, less variation in reward is evident. Similarly to the training environment with τ_{high}, μ_{fix}, reward for these gaits is most often higher (i.e., outside of 1.5x the interquartile range) in simulation. Videos and policies, each given in the supplementary data, show that an aperiodic gait is learned on the baseline and that while the gait learned on L1 is periodic, it does not leverage asymmetry of the front leg nor discover stick-slip behaviour.

Figure 10(g) presents the mean transfer error calculated across all trials for each setting under which robots were trained. Differences in transfer error are caused by differences in real reward achieved across the five trials for each real robot experiment and variations in simulation between the reward achieved by gaits trained with different random seeds.

In addition to reality transfer, we analyze transfer from the higher MOR tolerance (τ_{hi}) simulation environment to the higher-fidelity τ_{lo}

environment. Figure 10 shows the reward normalized by episode length for robots trained in each environment. In spite of the small differences in position error present between the two tolerances on the heuristic animations, the final trained gaits for τ_{hi} exhibit high transfer error to the τ_{low} simulation. The robot trained withτ_{hi}, μ_{rand} has a peak error exceeding 0.4cm/s in the low-friction setting.) Figure 10 also shows monotonicity of the reward with μ except for the case of τ_{hi}, μ_{fix} evaluated in the τ_{hi} environment.

7 Discussion

7.1 Expert Baseline Comparison

Robots trained in the lower-tolerance setting ($\tau_{\text{lo}} = 0.001$) outperform the baseline on cardboard in simulation and reality. Robot L1, which our algorithm learns without domain randomization, doubles the performances of the expert baseline on the foam board surface. The robot learned with τ_{lo} and μ_{rand} also outperforms the expert baseline on the foam board. Four out of five the five evaluated designs trained with τ_{lo}, μ_{fix}, Seed 0 (L1–L4) outperform the baseline in real-world physical experiments; L5 performs comparably to the baseline. One of the robots trained with τ_{hi}, μ_{rand}, Seed 0 even further exceeds the expert baseline (Fig. 1).

Learned robots that beat the baseline employ a 'pivoting' behaviour that allows for large bursts of forward sliding. Due to asymmetry, these robots roll sideways and the contact area of the front leg with the floor is reduced (leading to reduced frictional forces on that leg). This reduction in contact area enables the front PneuNet 'leg' to slip forward instead of pushing backward. We hypothesize that the asymmetry of the learned designs makes this motion possible.

We further show that training in the joint space of design and control can be advantageous over training a controller on the expert-developed design or the L1 design; here, policies trained through co-optimization achieve higher reward and stronger transfer in most cases (however, it is possible that longer training or other parameter tuning might improve this). In general, gaits trained through co-optimization are qualitatively different than those trained of fixed designs. Compared to voxel-based design spaces (Cheney et al, 2013), this design space is relatively small. While it is not clear what effects might arise by considering a finer-grained space (e.g., FEA element-level rather than actuator level, we note that modularization at the actuator level seems useful for practitioners with access to standard fabrication tools.)

A limitation of this approach is that it is not guaranteed to find global maxima across the joint space. The choice here to measure reward on all designs in the sample at each step may cut certain designs that have higher potential reward but require more controller training steps to reach this potential. These limitations might become more important as we add complexity to the design space and the controller.

7.2 Transfer

We see qualitatively successful sim-to-real for several designs. While the gaits learned in the τ_{hi} environment differ significantly from those learned in τ_{lo}, they have similar mean sim-to-real transfer error due to high variance in error for different surfaces. On the aggregate, no particular surface emerges as more error-prone than any other (Fig. 10 a-d). Further, sim-to-sim error of the τ_{hi} model (evaluated in the τ_{lo} environment) is comparable to its sim-to-real error (Figs. 10e-f).

The robots L1 and L4 have strong agreement with simulation—with the standard deviation across trials taken into account, real-world reward is within 0.9 cm (0.045cm/s normalized) of the reward achieved in simulation. Our baseline gait achieves poor reward in simulation. In key segments of the gait (depicted in the images in Figure 7(a)), the simulation records less progress, or backward progress, compared to what it achieves in the real world. In the τ_{lo} simulation environment, we measured performance of each robot across the range of μ used in training.

We further investigate policy transfer between training environments with different MOR tolerances; significant transfer error is found. Both simulated and real-world reward exhibit some non-monotonicity with respect to the parameter μ, showing the complexity of the joint design-control space. For τ_{low}, worse transfer performance is noted on the Table surface, whose friction parameter $\mu_{table} = 2.68$ is far out of the range of $\mu_{rand} \in \{0.65, ...1.3\}$. The robot trained with

τ_{hi}, μ_{rand} has the most consistent performance across all surfaces but comparable mean error. The learned gait uses two step types, likely each successful on different surfaces. Overall, the lowest transfer error occurs in robots trained with the lower-tolerance model reduction and evaluated in an environment with similar friction to some known μ_{fix}.

Though it is difficult to collapse multiple phenomena into a one-dimensional error measure, we identify stick-slip transitions as the main source of disagreement. Shown in the supplementary video and the gait traces of Fig. 7, stick-slip transition timing has a large effect on the transfer performance of our designs. Robots that do not transition at the correct time from stick to slip do not travel forward, and robots that stick with too high a force may flip over. Stick-slip transitions are known to be difficult to model (Luo and Hauser, 2015). Further, it is worth noting that the tested surfaces fall near or outside of the range of μ used in training. Custom design of surfaces for all friction increments used in training is left to future work. The material parameter fitting and action warping steps outlined in the appendix could arguably be identified as an early 'real-to-sim' step; one interesting direction could include fine-tuning using experimental data.

7.3 Domain Randomization Effects

Having identified stick-slip transitions as the main source of both performance and error, we perform domain randomization on the Coulomb friction parameter in an effort to further improve sim-to-real transfer. We evaluate how randomization of friction coefficient μ can affects transfer to surfaces with higher or lower friction and to reality. Prior work (Majidi et al, 2013) theoretically analyzes and empirically demonstrates the baseline sinusoidal gait's highly nonlinear sensitivity to frictional parameters in Coulomb friction. We instead evaluate the learned design-controller pairs for surfaces within the range of μ_{rand} (silicone mat, foam board) and outside it (table).

In the particular joint space studied here, domain randomization across a range of friction coefficients does not appear to support transfer from simulation to the real world. All training settings achieved comparable transfer error across all surfaces. Analyzing the learned gaits offers a potential reason. The robot with for τ_{lo}, μ_{rand} learns a gait which is less sensitive to friction in the simulator, but which uses longer, precarious steps that fail more easily in reality. The robot trained with $\tau_{\text{hi}}, \mu_{\text{rand}}$ offers the least variance in reward across the different surfaces by attempting a variety of step types. The model trained with τ_{hi} accrues significant error in transfer to the τ_{lo} environment.

Unlike in rigid robotics, where the effects of physical perturbations most often manifest as changes in a discrete set of values (e.g., joint torques), perturbations of a soft robot could affect motion anywhere along its continuous body. We hypothesize that soft systems' larger state spaces may make sim-to-real transfer challenging when using domain randomization over a single parameter. Selecting parameters to randomize is difficult; reducing sensitivity to one parameter could increase sensitivity to another. Our simulation environment offers 11 physical parameters that could be randomized (though the models themselves have more DoF–tens of thousands for the full-order model and tens to hundred for the ROM.) Future work will explore randomization of several parameters at once such as material parameters, encouraged by the result of Tiboni et al (2023), under constraints related to tractability and stability. It will perform ablation studies to understand parameters' significance.

8 Conclusion

This work describes a framework for the simulation and co-optimization of the design and control of modular, PneuNet-based crawling soft robots capable of sim-to-real transfer. We present a model-free algorithm for co-optimization together with a method for creating reconfigurable, high-fidelity reduced-order models, allowing our algorithm to efficiently optimize over designs with different topologies while preserving realism. We conduct a series of experiments that demonstrate that our framework learns design-policy pairs that outperform an expert-designed baseline in a soft robot locomotion task. We demonstrate and evaluate the successful qualitative and quantitative transfer of these learned pairs from simulation to reality. Further, we explore the robustness of the learned robots to new terrains and the utility of

this method in conjunction with domain randomization, a method shown to improve sim-to-real transfer of learned policies in rigid robotics.

It is difficult if not impossible to explore these design-control spaces through prototyping or analysis alone—simulation is needed to build and evaluate designs and controllers in a tractable manner. Deformability and nonlinear mechanics of soft robots introduce additional challenges in simulation. Soft roboticists experience a trade-off between simulation fidelity and speed. In addition to an expert comparison, this work investigates sim-to-real and sim-to-(higher-fidelity)sim transfer of policies to new terrains, and the effect of domain randomization of friction on performance and transfer. Designs trained with the higher-fidelity reduced model are discovered that have comparable performance (within uncertainty bounds) on all three surfaces. We find that the training environment tolerance (in the chosen range) and the presence of friction parametr randomization have a negligible effect on sim-to-real transfer error counted across all surfaces. We plan to explore adaptations of this framework with additional sim-to-real techniques (e.g., domain randomization of multiple variables) and to different design and control spaces.

Acknowledgments

We thank Olivier Goury and Hugo Talbot for assistance with the SOFA implementation, and Arthur MacKeith.

Declarations

Ethical Approval. Not applicable. **Competing Interests.** Not applicable. **Authors' Contributions.** A.S. and C.S. contributed equally to this manuscript. A.S., C.S. and M.W. wrote the text. A.S, M.W. and C.S. developed the framework and performed physical experiments. C.S. and S.N. performed computations. S.N. generated the video. All authors reviewed the manuscript. **Funding.** Partially funded by the National Sciences and Engineering Research Council of Canada (NSERC), Discovery Grants program (RGPIN 1512). **Availability of data and materials.** Code is available at the framework's Github Page: https://github.com/macrobotics-lab/evolving-soft-robots/tree/main. Data is available at https://github.com/macrobotics-lab/AuRo_evolving_sobot_data.

References

Bern JM, Banzet P, Poranne R, et al (2019) Trajectory optimization for cable-driven soft robot locomotion. In: Proceedings of Robotics: Science and Systems (RSS)

Bern JM, Schnider Y, Banzet P, et al (2020) Soft robot control with a learned differentiable model. In: Proceedings of the IEEE International Conference on Soft Robotics (RoboSoft)

Bhatia J, Jackson H, Tian Y, et al (2021) Evolution gym: A large-scale benchmark for evolving soft robots. In: Advances in Neural Information Processing Systems (NeurIPS)

Bongard J (2011) Morphological change in machines accelerates the evolution of robust behavior. Proceedings of the National Academy of Sciences 108(4):1234–1239

Bravo-Palacios G, Del Prete A, Wensing PM (2020) One robot for many tasks: Versatile co-design through stochastic programming. IEEE Robotics and Automation Letters 5(2):1680–1687

Bruder D, Gillespie B, Remy CD, et al (2019) Modeling and control of soft robots using the Koopman operator and model predictive control. In: Proceedings of Robotics: Science and Systems (RSS)

Centurelli A, Arleo L, Rizzo A, et al (2022) Closed-loop dynamic control of a soft manipulator using deep reinforcement learning. IEEE Robotics and Automation Letters 7(2):4741–4748

Chen F, Wang MY (2020) Design optimization of soft robots: A review of the state of the art. IEEE Robotics & Automation Magazine

Chen T, He Z, Ciocarlie M (2020) Hardware as policy: Mechanical and computational co-optimization using deep reinforcement learning. arXiv:200804460

Cheney N, MacCurdy R, Clune J, et al (2013) Unshackling evolution: evolving soft robots with multiple materials and a powerful generative encoding. In: Proceedings of the Annual Conference on Genetic and Evolutionary Computation (GECCO)

Coevoet E, Morales-Bieze T, Largilliere F, et al (2017) Software toolkit for modeling, simulation, and control of soft robots. Advanced Robotics 31

Collins J, Chand S, Vanderkop A, et al (2021) A review of physics simulators for robotic applications. IEEE Access

Culha U, Demir SO, Trimpe S, et al (2020) Learning of sub-optimal gait controllers for magnetic walking soft millirobots. In: Proceedings of Robotics: Science and Systems (RSS)

Deimel R, Irmisch P, Wall V, et al (2017) Automated co-design of soft hand morphology and control strategy for grasping. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)

Digumarti KM, Gehring C, Coros S, et al (2014) Concurrent optimization of mechanical design and locomotion control of a legged robot. In: Mobile Service Robotics. p 315–323

Drotman D, Jadhav S, Sharp D, et al (2021) Electronics-free pneumatic circuits for controlling soft-legged robots. Science Robotics 6(51)

Dubied M, Michelis MY, Spielberg A, et al (2022) Sim-to-real for soft robots using differentiable fem: Recipes for meshing, damping, and actuation. IEEE Robotics and Automation Letters 7(2):5015–5022

Farhat C, Avery P, Chapman T, et al (2014) Dimensional reduction of nonlinear finite element dynamic models with finite rotations and energy-based mesh sampling and weighting for computational efficiency. International Journal for Numerical Methods in Engineering 98(9):625–662

Faure F, Duriez C, Delingette H, et al (2012) SOFA: A multi-model framework for interactive physical simulation. In: Soft Tissue Biomechanical Modeling for Computer Assisted Surgery. Studies in Mechanobiology, Tissue Engineering and Biomaterials, p 283–321

Gamus B, Salem L, Gat AD, et al (2020) Understanding inchworm crawling for soft-robotics. IEEE Robotics and Automation Letters 5(2):1397–1404

Geilinger M, Poranne R, Desai R, et al (2018) Skaterbots: Optimization-based design and motion synthesis for robotic creatures with legs and wheels. ACM Transactions on Graphics (TOG) 37(4)

Goury O, Duriez C (2018) Fast, generic, and reliable control and simulation of soft robots using model order reduction. Transactions on Robotics 34(6):1565–1576

Ha D (2019) Reinforcement learning for improving agent design. Artificial Life 25(4):352–365

Ha S, Coros S, Alspach A, et al (2017) Joint optimization of robot design and motion parameters using the implicit function theorem. In: Proceedings of Robotics: Science and Systems (RSS)

Haarnoja T, Zhou A, Abbeel P, et al (2018) Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In: Proceedings of the International Conference on Machine Learning (ICML)

Hiller J, Lipson H (2014) Dynamic simulation of soft multimaterial 3D-printed objects. Soft Robotics 1:88–101

Howison T, Hauser S, Hughes J, et al (2020) Reality-assisted evolution of soft robots through large-scale physical experimentation: A review. arXiv:200913960

Hu Y, Liu J, Spielberg A, et al (2019) ChainQueen: A real-time differentiable physical simulator for soft robotics. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)

Huang W, Mordatch I, Pathak D (2020) One policy to control them all: Shared modular policies for agent-agnostic control. arXiv:200704976

Ju H, Juan R, Gomez R, et al (2022) Transferring policy of deep reinforcement learning from simulation to reality for robotics. Nature Machine Intelligence pp 1–11

Kim D, Kim SH, Kim T, et al (2021) Review of machine learning methods in soft robotics. PLOS ONE 16(2)

Kleeberger K, Bormann R, Kraus W, et al (2020) A survey on learning-based robotic grasping. Current Robotics Reports 1:239–249

Kriegman S, Walker S, Shah D, et al (2019) Automated shapeshifting for function recovery in damaged robots. In: Proceedings of Robotics: Science and Systems (RSS)

Kriegman S, Nasab AM, Shah D, et al (2020) Scalable sim-to-real transfer of soft robot designs. In: Proceedings of the IEEE International Conference on Soft Robotics (RoboSoft)

Kurin V, Igl M, Rocktäschel T, et al (2021) My body is a cage: The role of morphology in graph-based incompatible control. arXiv:201001856

Lee K, Kim S, Lim S, et al (2020) Generalized Tsallis entropy reinforcement learning and its application to soft mobile robots. In: Proceedings of Robotics: Science and Systems (RSS)

Li Y, Wang X, Kwok KW (2022) Towards adaptive continuous control of soft robotic manipulator using reinforcement learning. In: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, pp 7074–7081

Lipson H, Pollack JB (2000) Automatic design and manufacture of robotic lifeforms. Nature 406:974–978

Lipton JI, MacCurdy R, Manchester Z, et al (2018) Handedness in shearing auxetics creates rigid and compliant structures. Science 360(6389):632–635

Luo J, Hauser K (2015) Robust trajectory optimization under frictional contact with iterative learning. In: Proceedings of Robotics: Science and Systems (RSS)

Ma P, Du T, Zhang JZ, et al (2021) DiffAqua: A differentiable computational design pipeline for soft underwater swimmers with shape interpolation. arXiv preprint arXiv:210400837

Majidi C, Shepherd RF, Kramer RK, et al (2013) Influence of surface traction on soft robot undulation. The International Journal of Robotics Research 32(13):1577–1584

Morimoto R, Nishikawa S, Niiyama R, et al (2021) Model-free reinforcement learning with ensemble for a soft continuum robot arm. In: Proceedings of the IEEE International Conference on Soft Robotics (RoboSoft)

Morzadec T, Marcha D, Duriez C (2019) Toward shape optimization of soft robots. In: Proceedings of the IEEE International Conference on Soft Robotics (RoboSoft)

Mosadegh B, Polygerinos P, Keplinger C, et al (2014) Pneumatic networks for soft robotics that actuate rapidly. Advanced Functional Materials 24(15):2163–2170

Murata S, Kurokawa H (2007) Self-reconfigurable robots. IEEE Robotics & Automation Magazine 14(1):71–78

Park JH, Asada H (1994) Concurrent design optimization of mechanical structure and control for high speed robots. Journal of Dynamic Systems, Measurement, and Control 116(3):344–356

Pathak D, Lu C, Darrell T, et al (2019) Learning to control self-assembling morphologies: A study of generalization via modularity. arXiv:190205546

Paul C, Bongard JC (2001) The road less travelled: Morphology in the optimization of biped robot locomotion. In: Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)

Paul C, Valero-Cuevas FJ, Lipson H (2006) Design and control of tensegrity robots for locomotion. Transactions on Robotics 22(5)

Pinskier J, Howard D (2021) From bioinspiration to computer generation: Developments in autonomous soft robot design. Advanced Intelligent Systems

Qi H, Joyce K, Boyce M (2003) Durometer hardness and the stress-strain behavior of elastomeric materials. Rubber Chemistry and Technology 76(2):419–435

Rafsanjani A, Zhang Y, Liu B, et al (2018) Kirigami skins make a simple soft actuator crawl. Science Robotics 3(15)

Rus D, Tolley MT (2015) Design, fabrication and control of soft robots. Nature 521:467–475

Schaff C, Yunis D, Chakrabarti A, et al (2019) Jointly learning to construct and control agents using deep reinforcement learning. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)

Schaff C, Sedal A, Walter MR (2022) Soft robots learn to crawl: Jointly optimizing design and control with sim-to-real transfer. In: Proceedings of Robotics: Science and Systems (RSS)

Seo J, Paik J, Yim M (2019) Modular reconfigurable robotics. Annual Review of Control, Robotics, and Autonomous Systems 2:63–88

Shepherd RF, Ilievski F, Choi W, et al (2011) Multigait soft robot. Proceedings of the National Academy of Sciences 108(51):20,400–20,403

Sims K (1994) Evolving virtual creatures. In: Proceeding of the International Conference on Computer Graphics and Interactive Techniques (SIGGRAPH)

Spielberg A, Araki B, Sung C, et al (2017) Functional co-optimization of articulated robots. In: Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)

Spielberg A, Amini A, Chin L, et al (2021) Co-learning of task and sensor placement for soft robotics. IEEE Robotics and Automation Letters 6(2):1208–1215

Talamini J, Medvet E, Bartoli A, et al (2019) Evolutionary synthesis of sensing controllers for voxel-based soft robots. In: Proceedings of the Artificial Life Conference (ALIFE)

Tan J, Zhang T, Coumans E, et al (2018) Sim-to-real: Learning agile locomotion for quadruped robots. In: Proceedings of Robotics: Science and Systems (RSS)

Taylor O, Rodriguez A (2019) Optimal shape and motion planning for dynamic planar manipulation. Autonomous Robots 43(2):327–344

Tiboni G, Protopapa A, Tommasi T, et al (2023) Domain randomization for robust, affordable and effective closed-loop control of soft robots. arXiv preprint arXiv:230304136

Tobin J, Fong R, Ray A, et al (2017) Domain randomization for transferring deep neural networks from simulation to the real world. In: 2017 IEEE/RSJ international conference on intelligent robots and systems (IROS), IEEE, pp 23–30

Varghese NV, Mahmoud Q (2020) A survey of multi-task deep reinforcement learning. Electronics

Vikas V, Cohen E, Grassi R, et al (2016) Design and locomotion control of a soft robot using friction manipulation and motor–tendon actuation. IEEE Transactions on Robotics 32(4):949–959

Villarreal-Cervantes MG, Cruz-Villar CA, Alvarez-Gallegos J, et al (2013) Robust structure-control design approach for mechatronic systems. Transactions on Mechatronics 18(5):1592–1601

Vitanov I, Rizqi A, Althoefer K (2020) Shape reconstruction of soft-body manipulator: A learning-based approach. In: Proceedings of the Annual Conference Towards Autonomous Robotic Systems (TAROS)

Wang J, Chortos A (2022) Control strategies for soft robot systems. Advanced Intelligent Systems 4(5):2100,165

Weiss J (2019) A tutorial on the proper orthogonal decomposition. In: AIAA aviation 2019 forum, p 3333

Whitman J, Travers M, Choset H (2021) Learning modular robot control policies. arXiv:210510049

Zhang JZ, Zhang Y, Ma P, et al (2022) Sim2real for soft robotic fish via differentiable simulation. In: 2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), IEEE, pp 12,598–12,605

Zhao A, Xu J, Konaković-Luković M, et al (2020) Robogrammar: graph grammar for terrain-optimized robot design. ACM Transactions on Graphics (TOG) 39(6)

Zhu Y, Rossiter J, Hauser H (2019) Learning in growing robots: Knowledge transfer from tadpole to frog robot. In: Proceedings of the International Conference on Biomimetic and Biohybrid Systems