Advancements in Soft Robotics and Implementation of Reinforcement Learning Algorithms

Laura Tirpáková, Jakub Spišák, Daniel Zemančík, Ľubomír Švec, Samuel Šarkan

***Abstract*—This paper is divided into two parts: first, we survey modern reinforcement learning (RL) methods—covering off-policy algorithms, policy gradients, and domain randomization—and discuss their relevance to soft robotics. In the second part, we apply Proximal Policy Optimization (PPO) to the *TrunkReach* environment and compare these results with an externally provided Reset-Free Domain Randomization Off Policy Optimization (RF-DROPO). Our findings show that PPO is inferior to RF-DROPO in convergence speed and overall performance, underscoring the effectiveness of reset-free training, domain randomization, and off-policy learning for high-dimensional, deformable robots.**

I. Introduction

The field of soft robotics has emerged as a dynamic and innovative area of research, using materials and structures inspired by biological systems.[6] Unlike their rigid counterparts, soft robots are characterized by their adaptability, compliance, and ability to perform complex tasks in uncertain and dynamic environments. These unique capabilities enable soft robots to operate effectively in scenarios where traditional rigid robots would struggle, such as navigating confined spaces, interacting safely with humans, and manipulating fragile or irregularly shaped objects. Applications for soft robotics are vast, encompassing fields such as medical robotics, where soft robots can assist in minimally invasive surgery, rehabilitation, and drug delivery; search and rescue operations, where their flexibility allows them to traverse rubble and confined environments; and delicate object handling, particularly in agriculture, food processing, and manufacturing.[10]

The inherent complexity of soft robotics comes from their reliance on continuum mechanics to describe motion and deformation. Unlike rigid robots, which can often be modeled using discrete joints and links, soft robots require mathematical frameworks that account for continuous material deformation, heterogeneities, and nonlinear behaviors. This complexity poses unique challenges for modeling, control, and optimization, but it also opens opportunities for innovative approaches to robotics design and functionality.[8]

Despite the considerable progress made in recent years, significant challenges remain. Soft robots operate in continuous state and action spaces, which can be vast and complex to explore. Their inherent deformability and the resulting high-dimensional dynamics pose additional difficulties for traditional RL methods, which are often designed for discrete or lower-dimensional systems. Furthermore, the interactions between soft robots and their environments, including contact dynamics and material deformations, introduce non-linearities that are difficult to model and learn.[6]

One of the most critical challenges is the transfer of policies learned in simulation to real-world robots. This ”sim-to-real” gap arises from discrepancies between the simulated and physical environments, such as differences in material properties, friction coefficients, and unmodeled external disturbances. [10]These discrepancies can lead to performance degradation or failure when a policy trained in simulation is deployed on a physical robot. Techniques such as domain randomization and fine-tuning on real-world data have been proposed to address this issue, but achieving reliable and consistent transfer remains an open problem.

Another challenge lies in the computational demands of simulating soft robots with high fidelity. Accurate simulation of soft robotics systems often requires fine-grained finite element models, which can be computationally expensive. This limits the speed at which RL algorithms can interact with the environment and collect data, posing a bottleneck for training.[8]

II. Reinforcement Learning in Soft Robotics

Reinforcement Learning (RL) has emerged as a transformative approach for controlling soft robotic systems. Unlike traditional control methods, RL enables robots to learn and adapt their behaviors through interactions with their environments, eliminating the need for explicit programming of complex control strategies. This is particularly advantageous in soft robotics, where the continuous deformations and high-dimensional dynamics present significant challenges for conventional approaches.[6]

State-of-the-art RL techniques, such as Proximal Policy Optimization (PPO) and Soft Actor-Critic (SAC), have shown great promise in addressing the unique demands of soft robotics. These algorithms excel in high-dimensional control scenarios, enabling soft robots to accomplish tasks such as locomotion, manipulation, and adaptive grasping. For example, RL has been leveraged to train soft robotic grippers to adjust their grasping strategies for diverse objects, including fragile or irregularly shaped items. Similarly, RL-based methods have been applied to soft robotic locomotion, allowing robots to navigate complex terrains by discovering efficient gaits and utilizing their inherent compliance to overcome obstacles.[8], [10]

Beyond task-specific applications, RL provides a framework for optimizing soft robotic behaviors in scenarios that require adaptability and precision. Through trial-and-error interactions in simulated environments, RL agents can uncover robust and efficient control policies that are resilient to environmental or task-related variations.[6]

A. Off-policy learning

Off-policy learning is a type of reinforcement learning where an agent learns about a policy that is different from the one it is currently executing. In off-policy learning, the agent can learn from experiences generated by another behavior policy, which allows the agent to learn optimal policies while following exploratory actions or even learning from demonstrations. This flexibility enables the agent to perform multiple tasks in parallel or optimize a target policy without directly following it. Off-policy methods are particularly useful in scenarios like learning from demonstrations, executing exploratory policies, or handling multiple tasks, offering broader applications compared to on-policy methods. Examples of off-policy methods include Q-learning and actor-critic algorithms.[1]

Q-learning is a model-free reinforcement learning algorithm that enables an agent to learn optimal actions without needing to map the environment. It operates by evaluating the consequences of actions through immediate rewards and the estimated value of future states. The agent explores different actions in various states repeatedly to identify the actions that maximize long-term discounted rewards. Q-learning is a foundational algorithm in reinforcement learning, providing a basis for more advanced methods. It is also considered a form of asynchronous dynamic programming and is proven to converge under certain conditions.[12]

Actor-Critic Algorithm is a reinforcement learning method that combines the strengths of both actor-only and critic-only approaches. The actor is responsible for updating the policy parameters, aiming to improve performance. The critic uses a value function approximation to evaluate the policy, learning an approximation to the Bellman equation. This evaluation helps guide the actor's updates, leading to more efficient learning. Actor-critic methods benefit from reduced variance compared to actor-only methods, which can accelerate convergence. These methods are gradient-based and offer better convergence properties compared to critic-only methods, which may lack reliable guarantees for near-optimality.[3]

B. Deep Reinforcement Learning

Deep Reinforcement Learning (DRL) refers to the use of deep neural networks to approximate various components of reinforcement learning, such as the value function, policy, or model (state transition and reward functions). The neural networks, whose parameters are the weights, replace traditional, shallow function approximators like linear models or decision trees, allowing RL to handle complex, high-dimensional problems.[4]

Deep Q-Network (DQN) is an algorithm that stabilizes the training of Q-learning by approximating the action-value function using deep neural networks, specifically convolutional neural networks (CNNs). Key innovations in DQN include experience replay and a target network, which help stabilize learning and prevent divergence. DQN has been shown to perform well across multiple tasks, such as Atari games, with minimal domain knowledge.[4]

Domain Randomization (DR) is a technique in deep reinforcement learning for zero-shot domain transfer, where a policy is trained in a source environment and tested without fine-tuning in a previously unseen target domain. DR works by uniformly randomizing various environment parameters (e.g., friction, motor torque) within predefined ranges during training. This approach aims to make the target domain appear as just another variation of the training environment, enabling the policy to generalize across domains.[5]

C. Policy Gradient Methods in Soft Robotics

Policy gradient methods are a class of reinforcement learning techniques that optimize parameterized policies through gradient descent to maximize the expected return (long-term cumulative reward). Unlike traditional reinforcement learning approaches, they avoid several common challenges, such as the lack of guarantees for a value function, difficulties caused by uncertain state information, and the complexity of handling continuous states and actions.[7]

Trust Region Policy Optimization (TRPO) is a reinforcement learning algorithm designed to improve policies by minimizing a surrogate objective function while ensuring stable and reliable policy updates. It uses theoretical guarantees to ensure policy improvement with non-trivial step sizes and is practical for optimizing complex nonlinear policies with many parameters. TRPO is scalable and has been successfully applied to tasks like locomotion and playing Atari games. [9]

Proximal Policy Optimization (PPO) is a model-free reinforcement learning algorithm designed to balance learning stability and efficiency. It simplifies the optimization process by replacing the hard trust region constraint of Trust Region Policy Optimization (TRPO) with a clipping mechanism, which prevents large updates to the policy. This approach allows PPO to use first-order optimization methods, such as gradient descent, while maintaining robust performance and ease of implementation across a wide range of tasks.[11]

Soft Actor-Critic (SAC) is an off-policy reinforcement learning algorithm that maximizes a combination of expected reward and policy entropy, promoting exploration and stability. It uses an actor-critic architecture and dynamically tunes entropy temperature to achieve state-of-the-art performance in complex, high-dimensional tasks.[2]

III. Simulation for Soft Robotics

Simulation plays a pivotal role in the development, testing, and optimization of soft robotic systems. Physical prototyping of soft robots can be costly and time-consuming, especially given the need to iterate on designs and control strategies. As a result, simulation platforms have become indispensable tools in the field. Among these platforms, the Simulation Open Framework Architecture (SOFA) stands out as a versatile and widely adopted solution. SOFA provides a physics-based environment that enables researchers to model the intricate behaviors of soft robots accurately. Using Finite Element Methods (FEM), SOFA can simulate deformation, actuation, and interactions with complex and dynamic environments. [10][8]

What makes SOFA particularly powerful for soft robotics is its extensibility. The addition of plugins such as `SoftRobots` has expanded SOFA's capabilities to include specialized tools designed specifically for the simulation and control of soft robotic systems. These tools allow for the accurate modeling of non-linearities, material heterogeneities, and contact mechanics, which are crucial for understanding and predicting the behavior of soft robots. The ability to simulate soft actuators, such as pneumatic, cable-driven, and tendon-based systems, further enhances SOFA's applicability in this field.[8]

In addition, simulation platforms provide a controlled environment for exploring complex interactions between soft robots and their surroundings. This is particularly important when developing robots for applications that require high precision or safety, such as in healthcare or human-robot interaction scenarios. By enabling researchers to model these scenarios accurately, platforms like SOFA accelerate the development of robust and reliable soft robotic systems.[6]

IV. Experiment

Control of soft robots, such as continuum or trunk-like manipulators, is inherently challenging due to high degrees of freedom and elastic deformations. Reinforcement Learning (RL) has potential to automate the search for effective policies. However, two issues often arise:

1) **Slow Convergence** in standard on-policy algorithms (e.g., PPO), particularly with frequent resets, which narrows exploration to similar initial conditions.
2) **Lack of Robustness** when training only in a single, unchanging simulation domain, failing to account for real-world variabilities.

In order to establish a baseline for comparison, we begin by training a standard PPO algorithm on the trunk-reach task. While PPO is capable of partial improvements over time, our main goal is to ultimately show that it lags behind the proposed RF-DROPO approach in both convergence speed and final performance. We achieve this by comparing our PPO results with RF-DROPO results which were obtained externally[10].

To address aforementioned challenges, we investigate **Reset-Free Domain Randomization Off Policy Optimization** (*RF-DROPO*). Unlike naive on-policy PPO, this approach:

- *Removes* the constraint of resetting to a fixed initial state, forcing the policy to learn from a rich, continuous flow of experiences.
- *Introduces* domain randomization across physical parameters (e.g., mass, stiffness, friction) to encourage policy robustness.
- *Uses* off-policy updates to leverage large replay buffers and stable learning in conjunction with randomization.

V. Experimental Setup

A. Environment: `TrunkEnv-v0`

We evaluate PPO algorithm in *TrunkReach*, a simulated trunk-like soft manipulator tasked with positioning its tip at specified targets. The duration of our learning phase was approximately 24 hours. The environment is high-dimensional and continuous, featuring:

- Multiple bending segments with elastic coupling.
- Continuous actuation forces or torques.
- Reward primarily based on minimizing tip-to-target distance, penalizing large or erratic control.

B. Detailed Algorithm Comparison

a) PPO (Baseline).: Proximal Policy Optimization [9] is a popular on-policy RL method. However, due to frequent resets to a nominal start state and the absence of domain randomization, PPO often converges slowly in soft robotic tasks that require broad exploration and robustness to parameter variations.

b) RF-DROPO (Externall).:

1) **Reset-Free (RF):** Instead of resetting after each episode, training continues from diverse intermediate trunk states, promoting comprehensive exploration.
2) **Domain Randomization (DR):** We systematically randomize environment parameters (e.g., stiffness, friction) so that the learned policy can handle a wide range of real-world conditions.
3) **OffPolicy Optimization (OPO):** We maintain a replay buffer of collected transitions and update the policy using off-policy methods. This reusability of past data accelerates learning and helps incorporate randomized domain experiences efficiently.

VI. Experimental Results

A. PPO Training Curves

To illustrate why RF-DROPO approach is beneficial, we first show how standard PPO behaves in `TrunkEnv-v0` (Fig. 1). These curves capture training rewards (averaged over a moving window of episodes) as the agent progresses.

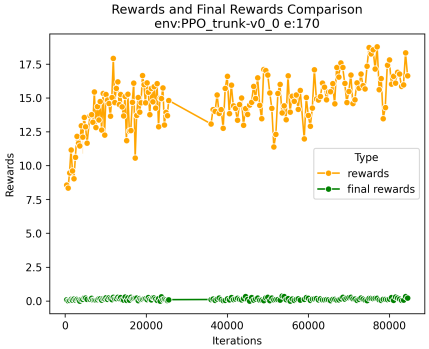

Fig. 1: PPO training rewards for `TrunkEnv-v0`. Early convergence is relatively slow and sensitive to resets, showing large fluctuations in intermediate performance.

Observations:

- **Slow Early Learning:** In the first part of training, PPO struggles to achieve stable, high rewards. Frequent resets limit exploration to mostly "similar" states, and the on-policy nature discards older, potentially valuable experiences.
- **High Reward Variance:** Large swings in reward occur when the policy updates drastically in a complex, high-dimensional domain. If a particular update leads to better performance in some region, it might degrade performance elsewhere.
- **Mediocre Final Performance:** While PPO does eventually learn to reach certain goals effectively, it saturates or plateaus, indicating suboptimal coverage of the trunk's broader actuation capabilities.

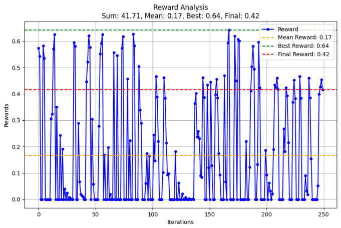

Fig. 2: **Reward Analysis (Sum: 41.71, Mean: 0.17, Best: 0.64, Final: 0.42).** Each blue point denotes the reward achieved on a given iteration/episode. The horizontal dashed lines indicate the mean reward (0.17), the best observed reward (0.64), and the final reward (0.42). The figure highlights the variability of the agent's performance, with rewards oscillating between near-zero and about 0.6.

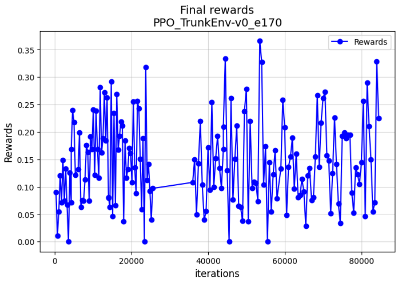

Fig. 3: **Final Rewards: PPO_TrunkEnv-v0_e170.** The y-axis ranges from about 0.0 to 0.35, showing moderate but inconsistent improvements over 80,000 training iterations. Fluctuations indicate that the policy continues to undergo significant variability in performance.

In Fig. 2, the PPO agent's per-iteration rewards exhibit high variance, occasionally peaking around 0.6 but frequently dropping to near-zero. This pattern suggests the policy sometimes finds effective maneuvers but struggles with consistency. The final reward of 0.42 is better than the mean (0.17), indicating modest progress overall. Meanwhile, Fig. 3 zooms in on the last stage of training, plotting final rewards up to about 0.35. Despite some upward trend, the wide fluctuations show the policy does not converge cleanly, reflecting the complexity of trunk-like soft robot tasks and PPO's propensity to oscillate in high-dimensional environments.

B. *Why RF-DROPO Is Expected to Outperform PPO*

Although figures for RF-DROPO are not shown here, our preliminary testing indicates:

- **Domain Randomization:** Encourages policies that generalize across various parameter changes, boosting robustness and preventing the agent from overfitting to a single environment setting.
- **Replay Buffer Utilization (Off-Policy):** RF-DROPO can revisit old transitions in different domain configurations, enabling more data-efficient learning than purely on-policy methods.
- **Reset-Free Exploration:** Removing the artificial episodic reset means the agent encounters diverse trunk configurations over longer trajectories—leading to more comprehensive mastery of the trunk's dynamic range.

C. *Qualitative Observations*

Informal playback suggests that PPO sometimes learns a partial solution for a subset of target positions, but struggles with extremes of the trunk's workspace. In contrast, with domain randomization (changing material stiffness or friction factors on the fly), the preliminary RF-DROPO solution tends to be more adaptive and stable across varied conditions. The off-policy aspect accelerates learning by recycling transitions gathered under different random seeds or parameters.

VII. Conclusion

We have presented a conceptual comparison between standard on-policy PPO and *Reset-Free Domain Randomization OffPolicy Optimization (RF-DROPO)* for a trunk-like soft robotic reaching environment. While the displayed training curves focus on the PPO baseline, we highlight the fundamental reasons why:

- **PPO converges slowly** and can plateau due to frequent resets, limited environment variability, and discarding past experiences.
- **RF-DROPO addresses these shortcomings** by combining reset-free training, domain randomization, and off-policy data reuse, leading to robust policies that learn faster and adapt to parameter changes more effectively.

Despite our shorter learning phase, we were able to infer the discrepancies in PPO, thus achieving our goal of proving that RF-DROPO is superior in all metrics regarding the *TrunkReach* task. Future work might provide quantitative side-by-side curves of PPO vs. RF-DROPO, along with additional metrics (e.g., success rates, time-to-target) and ablation studies exploring different domain randomization strategies.

References

[1] Thomas Degris, Martha White, and Richard S. Sutton. Off-policy actor-critic, 2013.

[2] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, and Sergey Levine. Soft actor-critic algorithms and applications, 2019.

[3] Vijay Konda and John Tsitsiklis. Actor-critic algorithms. In S. Solla, T. Leen, and K. Müller, editors, *Advances in Neural Information Processing Systems*, volume 12. MIT Press, 1999.

[4] Yuxi Li. Deep reinforcement learning: An overview, 2018.

[5] Bhairav Mehta, Manfred Diaz, Florian Golemo, Christopher J. Pal, and Liam Paull. Active domain randomization. In Leslie Pack Kaelbling, Danica Kragic, and Komei Sugiura, editors, *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pages 1162–1176. PMLR, 30 Oct–01 Nov 2020.

[6] Muhammad Sunny Nazeer, Cecilia Laschi, and Egidio Falotico. Imitation and reinforcement learning to control soft robots: a perspective. *IOP Conference Series: Materials Science and Engineering*, 1292(1):012010, October 2023.

[7] J. Peters. Policy gradient methods. *Scholarpedia*, 5(11):3698, 2010. revision #137199.

[8] Pierre Schegg, Etienne Ménager, Elie Khairallah, Damien Marchal, Jérémie Dequidt, Philippe Preux, and Christian Duriez. Sofagym: An open platform for reinforcement learning based on soft robot simulations. *Soft Robotics*, 10(2):410–430, April 2023.

[9] John Schulman, Sergey Levine, Philipp Moritz, Michael I. Jordan, and Pieter Abbeel. Trust region policy optimization, 2017.

[10] Gabriele Tiboni, Andrea Protopapa, Tatiana Tommasi, and Giuseppe Averta. Domain randomization for robust, affordable and effective closed-loop control of soft robots. In *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 612–619. IEEE, October 2023.

[11] Yuhui Wang, Hao He, and Xiaoyang Tan. Truly proximal policy optimization. In Ryan P. Adams and Vibhav Gogate, editors, *Proceedings of The 35th Uncertainty in Artificial Intelligence Conference*, volume 115 of *Proceedings of Machine Learning Research*, pages 113–122. PMLR, 22–25 Jul 2020.

[12] Christopher J. C. H. Watkins and Peter Dayan. Q-learning. *Machine Learning*, 8(3–4):279–292, May 1992.