Extended Abstract

Motivation Generalizing control policies across diverse robot embodiments remains a central challenge in robot learning. While imitation learning (IL) is effective at acquiring skills from expert demonstrations, it often produces policies that are tightly coupled to the specific embodiment of the training robot. A policy trained on a fixed-base arm, for example, may struggle to transfer to a mobile manipulator or a robot with different degrees of freedom. This lack of flexibility limits the development of generalist robots that can operate across platforms and environments.

To address this, we propose a data-efficient framework that combines the sample efficiency of IL with the adaptability of reinforcement learning (RL). Our goal is to learn a general policy that can be quickly adapted to novel robot embodiments without retraining from scratch.

Method We adopt a two-phase training strategy. In Phase I, we train a behavior cloning (BC) policy using demonstrations from multiple robot embodiments performing the same manipulation task. Our architecture includes a shared encoder that maps the robot's state and identity into a latent space, a latent policy that outputs task-level actions, and a decoder that maps these latent actions to robot-specific actuation. This modular design enables shared learning while preserving embodiment-specific control.

In Phase II, we fine-tune the latent policy, as well as the encoder and decoder using Soft Actor-Critic (SAC) on a single robot embodiment. This selective adaptation allows the policy to adjust to embodiment-specific dynamics while retaining task-level priors learned from demonstrations.

Implementation We conduct experiments in the Robosuite simulation platform using manipulation tasks Lift and Door. Expert demonstrations are collected from three robot embodiments, and the BC model is trained using a masked input/output representation to align diverse robot state and action spaces. SAC fine-tuning is performed with dense rewards using the robot to assess adaptability.

Results SAC fine-tuning consistently improves average reward over the BC baseline. In the Lift task, IIWA's average trajectory reward improves from 1.18 (BC) to 15.10 (SAC), and in the Door task, Sawyer improves from 373.82 (BC) to 609.88 (SAC). These results demonstrate the effectiveness of SAC in adapting shared policies to new embodiments.

However, none of the policies achieved consistent task success, highlighting limitations in generalization. Contributing factors include limited demonstration data (10 per robot), short fine-tuning horizons, and potential optimization difficulty from masking high-dimensional inputs.

Conclusion Our results show that combining behavior cloning with SAC fine-tuning in a shared latent policy architecture improves reward-based performance across robot embodiments. While we could not achieve full task success under these constrained settings, the proposed framework represents a promising step toward scalable, embodiment-agnostic robot learning. Future work should explore richer data, longer adaptation, and more expressive models to enhance both transferability and robustness.

Using RL to Generalize Robot Policies for Multiple Embodiments

Ariel Bachman
Department of Mechanical Engineering
Stanford University
arielbac@stanford.edu

Raúl Molina Gómez
Department of Mechanical Engineering
Stanford University
rmolinag@stanford.edu

Daniel Voxlin
Department of Electrical Engineering
Stanford University
dvoxlin@stanford.edu

Abstract

Generalizing control policies across robot embodiments is a core challenge in robotics. While imitation learning (IL) efficiently acquires skills from expert demonstrations, resulting policies often overfit to the morphology of the training robot. We propose a two-phase framework that combines the sample efficiency of behavior cloning (BC) with the adaptability of reinforcement learning (RL) to develop embodiment-aware, generalizable policies. In Phase I, we train a BC policy using demonstrations from multiple robot embodiments performing a shared task. A shared encoder, latent policy, and decoder architecture enables transfer by mapping observations to a common latent space and decoding them into robot-specific actions. In Phase II, we fine-tune the latent policy, encoder and decoder using Soft Actor-Critic (SAC) on a single embodiment, allowing adaptation to novel dynamics without retraining the task-level policy. We evaluate our approach on the Lift and Door tasks in Robosuite using three robot embodiments. SAC fine-tuning consistently improves rewards over BC-only policies, although task success remains limited due to data and compute constraints. Our results highlight the potential of modular IL and RL frameworks for scalable cross-embodiment robot learning and point to future improvements in robustness and generalization.

1 Introduction

The primary objective of our project is to develop an efficient adaptation framework that leverages online reinforcement learning (RL) to enable robot control policies, which are initially acquired through imitation learning (IL), to generalize across a broad range of robotic embodiments. While imitation learning has proven highly effective for acquiring complex skills from expert demonstrations, its generalization capability is often constrained by the embodiment on which it was trained. A policy trained on one specific robot (e.g. a fixed-base arm) typically struggles to transfer to another with different kinematics, dynamics, or degrees of freedom, such as a mobile manipulator or a humanoid platform. This lack of flexibility represents a major bottleneck in realizing the vision of generalist robots that can seamlessly operate in unstructured and dynamic real-world settings.

To address this limitation, we propose a framework that combines the sample efficiency and task priors of imitation learning with the online adaptability and exploration capabilities of reinforcement learning. Our approach begins with training a behavior cloning (BC) policy across multiple robot embodiments, allowing the policy to learn a shared representation of task-relevant features. However,

instead of relying solely on offline demonstrations, we further improve this policy using online RL to enable adaptation to novel or underrepresented embodiments. This two-stage learning process allows the policy to retain core task knowledge from demonstrations while refining its control strategy in response to embodiment-specific variations encountered during deployment.

Concretely, our goal is to train a general policy that can control at least three distinct robot embodiments performing fundamental manipulation tasks such as lifting objects and opening doors. We initially implemented and evaluated these tasks in the RoboCasa simulation environment, which provides a rich testbed for task objectives that supports domain variation. However, due to limitations in the availability of robot embodiments within RoboCasa, we then transitioned to the Robosuite environment, which also supports simulation and testing of robot tasks.

Our architecture is designed to balance shared learning and embodiment-specific adaptation. We explore a modular design where a shared encoder learns a common latent representation of proprioceptive and task information, and a decoder conditions on robot-specific parameters, such as kinematic structure or action dimensions, to produce valid actions. The online RL phase fine-tunes this architecture, improving both task performance and generalization across embodiments through continuous interaction with the environment.

The ability to train general policies that adapt online is a critical step toward scalable and reusable robot learning systems. By minimizing the need for per-robot retraining, such policies dramatically reduce data and compute requirements for real-world deployment. Moreover, our approach offers a pathway toward lifelong learning for robots, where a single policy can incrementally adapt to new embodiments, tasks, and environments over time. Ultimately, this project contributes to the broader goal of building embodiment-agnostic control frameworks for generalist robots.

2 Related Work

Recent efforts to address the cross-embodiment challenge in reinforcement learning (RL) have explored a range of strategies to enable agents to generalize across different robotic platforms. One paper focuses on latent space alignment as a mechanism for policy transfer. Wang et al. Wang et al. (2024) propose a method that maps the state and action spaces of source and target robots into a shared latent representation using encoders and decoders trained jointly with a latent-space policy. This framework employs generative adversarial training with cycle consistency and does not require access to reward signals or task-specific tuning in the target domain. The disadvantage of this approach from our point of view is the extra training and implementation required for the decoding and encoding of the latent space. A significant amount of computation is required, and for the objective described above, it may not be the best approach.

In contrast to representation learning approaches, Doshi, et al. Doshi et al. (2024) introduce CrossFormer, a scalable, transformer-based policy architecture designed to learn from large-scale multi-embodiment data. CrossFormer is trained on a dataset comprising 900k trajectories collected from 20 different robot embodiments, including manipulators, mobile bases, quadcopters, and quadrupeds. The model does not require manual alignment of observation or action spaces and is capable of controlling all robot types using the same set of network weights. The authors show that CrossFormer not only matches the performance of expert-tuned policies for individual robots but also outperforms prior cross-embodiment models in generalization tasks, highlighting the benefits of data scale and flexible architectures. However, collecting 900k trajectories is not always achievable and requires extensive computation. So, our method attempts to achieve the same goal but with much more efficient data usage of the expert demonstrations in RoboCasa and Robosuite.

Addressing the problem of multi-source transfer across heterogeneous domains, Heng et al. Heng et al. (2022) present the Cross-domain Adaptive Transfer (CAT) framework. Unlike previous work limited to single-source transfer and shared state-action spaces, CAT learns task-specific state-action correspondences from multiple source policies and adaptively integrates them to guide learning on a new target task. Although effective, we think that CAT does not generalize easily to settings involving continuous or unsupervised embodiment adaptation.

3 Method

We adopt a two-phase training strategy to develop robotic control policies that generalize across multiple embodiments and are capable of adaptation to specific robot hardware. The method consists of an offline training via Behavior Cloning (BC) from expert demonstrations and an online fine-tuning using Soft Actor-Critic (SAC). This framework enables us to share skills across robot embodiments while retaining the capacity to specialize when necessary.

3.1 Phase I: Behavior Cloning in Latent Space

In the first phase, we collect expert demonstrations for a the tasks (Lift and Door) across a set of three robot embodiments. These trajectories consist of timestamped observations and corresponding actions. Since each robot has different observation and action dimensions, we define:

- A set of shared observation keys that are common to all robots (e.g., end-effector position, object pose). However, they present different dimensions for each embodiment, so we need a mask to select the correct space for each robot.
- The maximum state and action dimensions observed among all robots, to define padded input/output spaces.
- A robot-specific binary mask over states and actions, indicating which dimensions are active, as described before.

Our BC model is composed of three modules:

1. **Encoder** E_ϕ: maps a padded and masked state $s \in \mathbb{R}^{d_s}$ and a robot identifier $r \in \mathbb{N}$ to a latent representation $z \in \mathbb{R}^{d_z}$:
$$z = E_\phi(s, r)$$
2. **Latent Policy** π_θ: a stochastic Gaussian policy over latent actions $a_z \sim \mathcal{N}(\mu(z), \sigma(z)^2)$.
3. **Decoder** D_ψ: maps the latent action $a_z \in \mathbb{R}^{d_{a_z}}$ and robot ID r back to the padded action space $\hat{a} \in \mathbb{R}^{d_a}$, applying the appropriate action mask:
$$\hat{a} = D_\psi(a_z, r)$$

The policy is trained to minimize the mean squared error between $\hat{a}$ and the expert action a_{expert}, across demonstrations from all robots:

$$\mathcal{L}_{\text{BC}} = \mathbb{E}_{(s,a,r)} \left[\| D_\psi(\pi_\theta(E_\phi(s, r)), r) - a \|^2 \right]$$

Robot-specific conditioning is incorporated through a one-hot embedding of the robot ID, which is input to both the encoder and decoder. This enables the model to share structure across robots while still accounting for embodiment-specific variations.

3.2 Phase II: Fine-Tuning with Soft Actor-Critic (SAC)

To improve task performance and embodiment-specific precision, we fine-tune the policy using reinforcement learning with the Soft Actor-Critic (SAC) algorithm. We chose SAC because it is off-policy, which allow us to take advantage of the online algorithm while having more exploration and robustness with the Replay Buffer.

Architecture and Initialization

We initialize the actor network with the BC-trained latent policy π_θ, and reuse the same encoder E_ϕ and decoder D_ψ. Two critic networks Q_{ω_1} and Q_{ω_2} are trained to estimate the Q-values of state-action pairs in latent space. Target networks Q'_{ω_1} and Q'_{ω_2} are maintained for stability via averaging.

Replay Buffer and Warmup

A replay buffer stores tuples $(s, a, r, s', d, r_{\text{id}})$. To address the cold-start problem, we use a warm-up phase to populate the buffer either by acting in the environment with the initial policy or by preloading expert demonstrations. In this case, we simply acted on the environment since we had the initial BC policy.

Critic Update

Given a batch of transitions, the target value is computed using the target networks and sampled latent actions:

$$a'_z \sim \pi_\theta(E_\phi(s'), r) \qquad \hat{a}' = D_\psi(a'_z, r)$$
$$y = r + \gamma(1-d) \min_{i=1,2} Q'_{\omega_i}(E_\phi(s'), \hat{a}')$$

The critic loss is:

$$\mathcal{L}_{\text{critic}} = \mathbb{E}\left[(Q_{\omega_1}(z, \hat{a}) - y)^2 + (Q_{\omega_2}(z, \hat{a}) - y)^2\right]$$

Actor and Encoder-Decoder Update

The actor loss encourages high Q-value actions while maximizing entropy:

$$\mathcal{L}_{\text{actor}} = \mathbb{E}_{z \sim E_\phi(s)}\left[\alpha \log \pi_\theta(a_z|z) - Q_{\omega_1}(z, D_\psi(a_z, r))\right]$$

We also update the encoder E_ϕ and decoder D_ψ jointly with the actor to refine the latent representation and embodiment-specific action decoding.

Target Network Updates

Critic target networks are updated with soft updates:

$$\omega'_i \leftarrow \tau\omega_i + (1-\tau)\omega'_i, \quad \text{with } \tau \ll 1$$

3.3 Evaluation Protocol

Both the BC and SAC policies are evaluated over 20 test episodes per robot embodiment. Each episode is capped at 5000 timesteps. The evaluation measures cumulative task reward per episode, allowing comparison between:

- The generalization capability of the shared BC policy.
- The improvement from SAC-based fine-tuning for each embodiment.

4 Experimental Setup

4.1 Environment and Data

All experiments are conducted in the **Robosuite** simulation framework Zhu et al. (2025), which we selected after initial trials in RoboCasa. While RoboCasa was used for early testing due to its high-fidelity manipulation environments Nasiriany et al. (2024), it currently supports only a single robot embodiment. As our study specifically targets cross-embodiment generalization, we migrated to Robosuite for its broader support of heterogeneous robots and consistent simulation structure.

Robosuite is particularly well-suited to our study for several reasons. First, it provides a diverse collection of robotic embodiments, including fixed-base arms (e.g., Sawyer, Panda), mobile manipulators (e.g., Stretch), and dual-arm systems (e.g., Baxter), all exposed through a consistent task interface. This makes it an ideal testbed for studying embodiment generalization.

Second, Robosuite supports a range of manipulation tasks with standardized reward functions and task success criteria, including Lift and Door. These tasks provide both dense and sparse rewards, enabling stable offline training and meaningful online improvement.

Third, Robosuite's deterministic physics engine ensures repeatable results and fine-grained control over environment parameters, which is essential for consistent evaluation across training and testing phases. The dataset of demonstrations used for behavior cloning is generated using scripted expert policies across three distinct robot embodiments per task. This enables the policy to learn from diverse embodiment configurations while maintaining a shared task objective.

(a) Example RoboCasa environment

(b) Example Robosuite environment for Stack task

Figure 1: RoboCasa and Robosuite environments

4.2 Experiments

Our experimental procedure is designed to evaluate the effectiveness of combining offline imitation learning with online reinforcement learning for embodiment-agnostic policy adaptation. We focus on a controlled set of manipulation tasks in the Robosuite environment, leveraging its standardized interface, physics simulation, and consistent task definitions across multiple robotic embodiments.

We begin by training a behavior cloning (BC) policy on a single manipulation task using demonstrations collected from three distinct robot embodiments (Sawyer, Panda, and IIWA). These demonstrations are generated using scripted expert policies, and the BC model is trained to minimize prediction error on the action sequences. The policy is trained using the shared encoder–decoder architecture described in Section 3, with shared latent representations and embodiment-aware decoding.

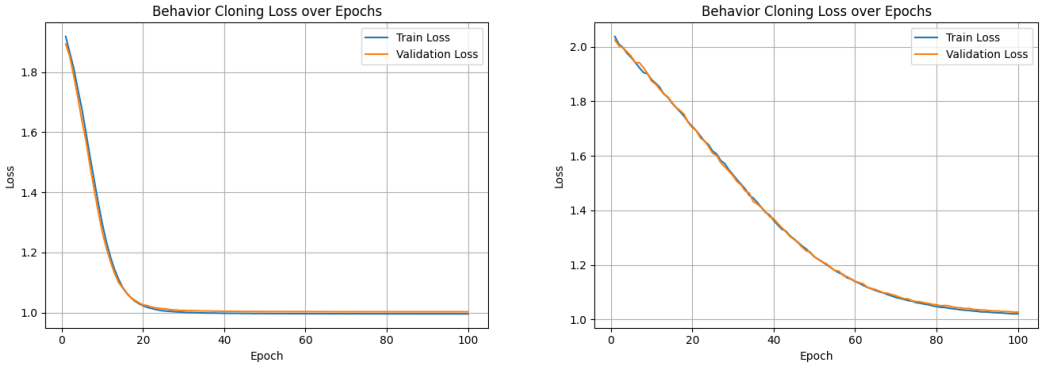
After BC training, we evaluate the generalization ability of the policy by testing it on a single robot embodiment on the same task. The initial evaluation provides a measure of how well the policy transfers across embodiments based solely on demonstration-based supervision. Following this zero-shot test, we fine-tune the policy using Soft Actor-Critic (SAC) with dense reward signals provided by the Robosuite environment. The online fine-tuning is conducted using the same single embodiment, allowing the policy to adapt to embodiment-specific dynamics and discrepancies not captured during offline training.

This process is repeated across a predefined set of manipulation tasks (Lift and Door), with each task run independently through the pipeline of BC pretraining, evaluation, SAC fine-tuning, and post-adaptation evaluation. For each configuration, we run evaluation episodes on the test embodiment both before and after fine-tuning. Each evaluation consists of 20 episodes, each with a maximum of 5000 time steps.

Performance is assessed using two key metrics: the average accumulated reward over each evaluation run, and the success rate, defined as the proportion of episodes in which the task-specific success condition is met (e.g., lifting an object above a height threshold or opening a door). These metrics provide complementary insights into the overall competence of the policy (via reward) and its robustness to embodiment variation (via success rate). Results are aggregated across tasks and embodiments to provide a comprehensive view of generalization and adaptability.

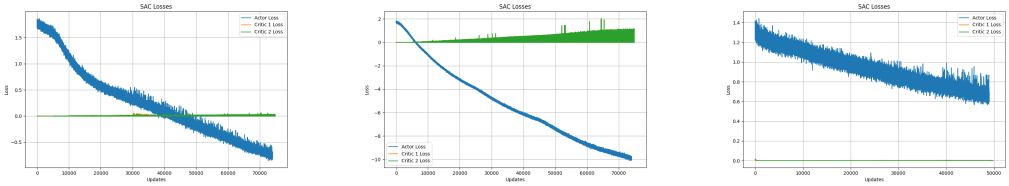
5 Results

5.1 Training



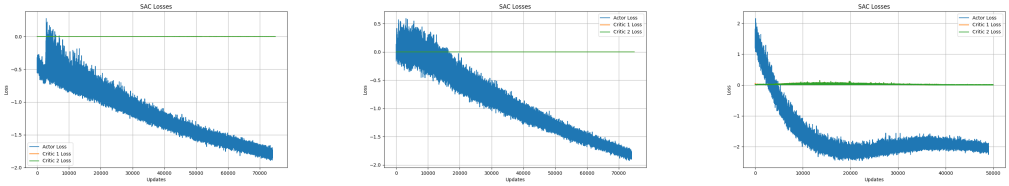
(a) Loss Curve for Door Task (b) Loss Curve for Lift Task

Figure 2: Behavior Cloning Training Loss Curves for Door and Lift Tasks in the Robosuite Environment



(a) IIWA (b) Sawyer (c) Panda

Figure 3: SAC Fine-Tuning Loss Curves for the Door Task in the Robosuite Environment



(a) IIWA (b) Sawyer (c) Panda

Figure 4: SAC Fine-Tuning Loss Curves for the Lift Task in the Robosuite Environment

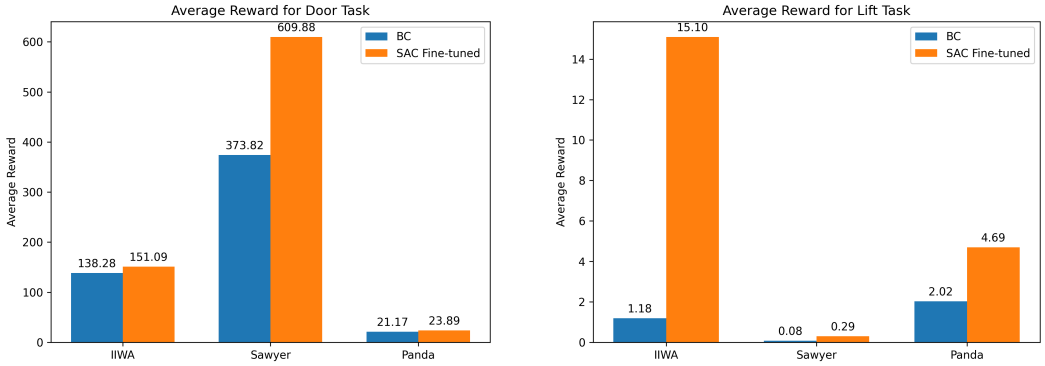
Figure 2 shows the training and validation loss curves for behavior cloning (BC) on the Door and Lift tasks. For the Door task (Figure 2a), the loss decreases rapidly in the first 20 epochs and converges to a low value around 1.0. The close alignment between training and validation losses suggests strong generalization and minimal overfitting. In contrast, the Lift task (Figure 2b) shows a slower convergence from an initial loss of about 2.0 to around 1.2, indicating a more challenging learning problem, though the training remains stable and well-aligned with validation.

Figures 3 and 4 present the training loss curves during SAC fine-tuning for the Door and Lift tasks, respectively, across three robot embodiments (IIWA, Sawyer and Panda). In all cases, we observe a consistent downward trend in loss, though with higher variance in the early stages, which is a common characteristic of reinforcement learning optimization. The Door task generally exhibits smoother

convergence, while the Lift task demonstrates more variability in some embodiments, suggesting differing levels of task difficulty and embodiment compatibility.

Overall, these results confirm that behavior cloning provides a reliable policy initialization, especially for the Door task. SAC fine-tuning further reduces the loss and helps adapt the latent policy to each specific robot embodiment, though the extent of improvement varies depending on the task and robot configuration.

5.2 Evaluation



(a) Average Reward for Door Task

(b) Average Reward for Lift Task

Figure 5: Evaluation results for IIWA, Sawyer, and Panda embodiments on Door and Lift tasks using Behavior Cloning (BC) and SAC fine-tuning.

Figure 5 shows the average reward achieved by policies trained using Behavior Cloning (BC) and those fine-tuned with Soft Actor-Critic (SAC) across three robot embodiments (IIWA, Sawyer, and Panda) for the Door and Lift tasks.

In the Door task (Figure 5a), SAC fine-tuning consistently improves performance over the BC baseline for all embodiments. The Sawyer robot exhibits the most significant improvement, with average reward increasing from 373.82 (BC) to 609.88 (SAC), suggesting strong alignment between the embodiment and the task. IIWA sees a modest increase from 138.28 to 151.09, indicating that the BC policy was already reasonably effective. The Panda robot performs poorly on this task under both BC and SAC (21.17 to 23.89), likely due to a mismatch between the embodiment and the demonstrations or task dynamics.

For the Lift task (Figure 5b), SAC also improves reward in all embodiments, though the degree of improvement varies. The IIWA robot shows the largest gain, increasing from 1.18 to 15.10, indicating that SAC is highly effective in enhancing an initially weak BC policy. Panda improves from 2.02 to 4.69, showing moderate benefit. The Sawyer robot, however, performs poorly in both cases (0.08 to 0.29), suggesting difficulties in adapting the policy to the Lift task for this specific embodiment.

In summary, SAC fine-tuning enhances BC-initialized policies across all embodiments and tasks, but the magnitude of improvement is highly dependent on the specific robot-task pairing. These results underscore the need for embodiment-aware policy training and evaluation when designing generalizable robotic control strategies.

Another metric used for evaluation is success rate. In our case, we achieved 0 success across the board for both the BC and the SAC fine-tuned policies.

Table 1: Total Reward Comparison for Door Task

Method	IIWA	Sawyer	Panda
BC	138.28	373.82	21.17
SAC Finetuned	151.09	609.88	23.89
Performance Improvement	1.09x	1.63x	1.12x

Table 2: Total Reward Comparison for Lift Task

Method	IIWA	Sawyer	Panda
BC	1.18	0.08	2.02
SAC Finetuned	15.10	0.29	4.69
Performance Improvement	12.80x	3.625x	2.32x

6 Discussion

We observe that fine-tuning with SAC consistently improves the average reward obtained by policies initially trained via behavior cloning (BC), supporting the hypothesis that reinforcement learning can effectively adapt a general latent policy to different robot embodiments. This improvement is especially evident in cases where the initial BC policy performed poorly, such as the Lift task with the IIWA robot, where SAC fine-tuning led to a more than tenfold increase in reward. These results highlight the potential of combining offline imitation learning with online reinforcement learning to bridge embodiment gaps in robotic control.

However, despite these improvements in reward, none of the evaluated robot embodiments achieved reliable task success under either BC or SAC policies. For example, while the Sawyer robot achieved relatively high reward on the Door task after SAC fine-tuning, its performance on the Lift task remained minimal, suggesting that reward improvement alone may not translate into task completion. This disconnect underscores the challenge of achieving both generality and robustness in multi-embodiment policy learning.

Several factors likely contributed to the limited performance observed. First, the dataset used for behavior cloning consisted of only 10 demonstration trajectories per robot embodiment. While our goal was to explore data-efficient policy learning, this small dataset was likely insufficient for capturing the full diversity of embodiment-specific state-action distributions needed for generalization. In addition, our BC policy operated in a shared latent space using a masked encoder-decoder architecture. Although this allowed for scalable multi-embodiment training, it may have constrained expressiveness by requiring all embodiments to conform to a common latent representation, which may not fully capture their kinematic or dynamic differences.

Second, the SAC fine-tuning phase was limited in both training duration and computational budget. Due to these constraints, we performed relatively few episodes of interaction per embodiment, which likely limited the policy's ability to meaningfully adapt. Furthermore, the reward signals in tasks like Lift and Door can be sparse and delayed, further slowing the learning process and requiring more samples for convergence. In practice, longer fine-tuning could help mitigate this issue.

Lastly, we note that the masking and padding approach used to align observations and actions across robots introduces challenges in optimization. The policy must learn to ignore irrelevant padded dimensions and focus only on the masked entries, which may introduce noise and instability, particularly in high-dimensional observation spaces.

In future work, we suggest exploring richer datasets with more demonstrations per embodiment, improved fine-tuning strategies such as prioritized experience replay, and alternative architectures for multi-robot generalization. For example, conditioning policies explicitly on robot-specific embeddings or using transformer-based architectures could help better leverage embodiment-specific priors while still enabling generalization. Additionally, using task success as an auxiliary signal during training could better align policy optimization with the final performance metric of interest.

7 Conclusion

In this work, we presented a two-phase learning framework that combines behavior cloning (BC) with Soft Actor-Critic (SAC) fine-tuning to develop generalizable policies across multiple robot embodiments performing shared tasks. By leveraging a shared encoder-decoder architecture and a masked latent policy representation, we enabled scalable training across diverse embodiments in a common latent space.

Our results show that SAC fine-tuning consistently improves the average reward over policies initialized via BC, particularly in cases where the BC policy is weak. This confirms the effectiveness of reinforcement learning for embodiment-specific adaptation. However, despite these gains, none of the tested embodiments achieved reliable task success, highlighting limitations in generalization and robustness under data-scarce and time-constrained conditions.

These findings suggest that while shared latent architectures offer a promising path toward scalable multi-embodiment learning, additional data, extended fine-tuning, and more expressive policy representations are necessary to achieve reliable generalization. Future work should focus on addressing these limitations through richer demonstration datasets, improved optimization strategies, and architectures better suited to capturing embodiment-specific dynamics while retaining the benefits of shared representation learning.

8 Team Contributions

- **Ariel Bachman**: Handled the evaluation pipeline, including defining metrics, running generalization tests and analyzing the results.
- **Raúl Molina Gómez**: Focused on the implementation of the SAC algorithm and integrating cross-embodiment data from the Robosuite environment. He also led the training and tuning of the policy.
- **Daniel Voxlin**: Designed and conducted the imitation learning baseline experiments, including policy transfer tests and data collection across different robot types.

Changes from Proposal The roles of Ariel Bachman and Daniel Voxlin were switched to accommodate scheduling constraints that arose during the quarter. Nonetheless, all team members collaborated closely throughout the project, and contributions often extended beyond the originally assigned responsibilities.

References

Ria Doshi, Homer Walke, Oier Mees, Sudeep Dasari, and Sergey Levine. 2024. Scaling Cross-Embodied Learning: One Policy for Manipulation, Navigation, Locomotion and Aviation. arXiv:2408.11812 [cs.RO] https://arxiv.org/abs/2408.11812

You Heng, Tianpei Yang, YAN ZHENG, Jianye HAO, and Matthew E. Taylor. 2022. Cross-domain Adaptive Transfer Reinforcement Learning Based on State-Action Correspondence. In *The 38th Conference on Uncertainty in Artificial Intelligence*. https://openreview.net/forum?id=ShN3hPUsce5

Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. 2024. RoboCasa: Large-Scale Simulation of Everyday Tasks for Generalist Robots. In *Robotics: Science and Systems (RSS)*.

Tianyu Wang, Dwait Bhatt, Xiaolong Wang, and Nikolay Atanasov. 2024. Cross-Embodiment Robot Manipulation Skill Transfer using Latent Space Alignment. arXiv:2406.01968 [cs.RO] https://arxiv.org/abs/2406.01968

Yuke Zhu, Josiah Wong, Ajay Mandlekar, Roberto Martín-Martín, Abhishek Joshi, Kevin Lin, Abhiram Maddukuri, Soroush Nasiriany, and Yifeng Zhu. 2025. robosuite: A Modular Simulation Framework and Benchmark for Robot Learning. arXiv:2009.12293 [cs.RO] https://arxiv.org/abs/2009.12293