CRADLE: Embracing Variability in Cross-Embodiment Transfer

Anonymous Author(s)
Affiliation
Address
email

Abstract: Reusing large datasets is crucial to scale vision-based robotic manipulators to everyday scenarios due to the high cost of collecting robotic datasets. However, robotic platforms possess varying control schemes, camera viewpoints, kinematic configurations, and end-effector morphologies, posing significant challenges when transferring manipulation skills from one platform to another. To tackle this problem, we propose a set of key design decisions to train a single policy for deployment on multiple robotic platforms. Our framework first aligns the observation and action spaces of our policy across embodiments via utilizing wrist cameras and a unified, but modular codebase. To bridge the remaining domain shift, we align our policy's internal representations across embodiments through contrastive learning. We evaluate our method on a dataset collected over 60 hours spanning 6 tasks and 3 robots with varying joint configurations and sizes: the WidowX 250S, the Franka Emika Panda, and the Sawyer. Our results demonstrate significant improvements in success rate and sample efficiency for our policy when using new task data collected on a different robot, validating our proposed design decisions. More details and videos can be found on our anonymized project website: https://sites.google.com/view/cradle-multirobot

Keywords: vision-based manipulation, multi-robot generalization

1 Introduction

Leveraging large datasets is essential to learning widely generalizable models in computer vision [1] and natural language processing [2]. In robotic manipulation, a promising avenue of research lies in the collection of similar extensive, in-domain datasets with the aspiration that they bestow comparable benefits. However, while past datasets have demonstrated good generalizability within the same hardware setup, applying them to different robotic configurations has proven difficult [3]. This challenge stems from four sources of variation: control scheme, camera viewpoint, kinematic configuration, and end-effector morphology. Each of these factors can vary significantly across robotic setups, leading to a large domain shift when transferring data collected on one robot platform to another. In this paper, we study the problem of how to mitigate this issue and effectively leverage robotic data across different platforms, making a stride toward learning widely-applicable policies.

In an effort to bridge the aforementioned domain gap, prior works have made advancements in enabling transfer across a subset of factors of variation. Early works have studied cross-embodiment transfer across kinematic configurations from low-dimensional observations [4, 5]. More recently, efforts have shifted towards utilizing high-dimensional image observations, enabling transfer across robotic hardware with a fixed camera angle for 3-DoF tasks [6] and across end-effectors with a fixed embodiment [7]. Unlike these works, we do not constrain the camera viewpoint, embodiment, or low-level controller to be fixed. Instead, we propose several new design choices to align the observation and action spaces across robots, such as using wrist-mounted cameras and a shared inverse

Submitted to the 7th Conference on Robot Learning (CoRL 2023). Do not distribute.

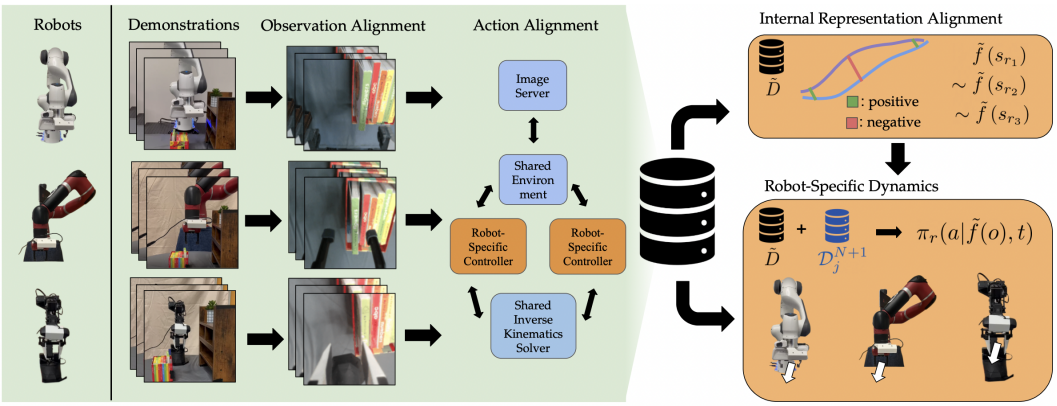

Figure 1: **Our framework for generalization across multiple robots.** We first standardize our observation space using front-mounted wrist cameras and our action space using a shared higher-level control environment. We then align our policy's internal representations using contrastive learning then finetune these representations to learn robot-specific dynamics.

kinematics solver. Each of these choices greatly mitigate the domain shift across embodiments without compromising the generality of the robotic setup.

We integrate these design choices into a single framework that aligns the *input*, *output*, and *internal representation* spaces of our policy across embodiments. Our choice of utilizing front-mounted wrist cameras aligns the input (or observation) space by naturally reducing the visual variation between embodiments. This allows us to remove assumptions about collecting data only under a specific fixed angle. In order to align the output space of our policy, we employ a shared inverse kinematics solver, while allowing the lower-level controller to vary. Although we would ideally completely unify the output space by using a single abstract action space across robots, this is infeasible due to disparities in action interpretation arising from hardware discrepancies. We instead learn a multiheaded policy with robot-specific heads that capture separate dynamics. Finally, we exploit a consistent, low-dimensional proprioceptive state signal to align our policy's internal representations. This facilitates generalization across the remaining factors that may cause a domain shift.

Our main contribution is a pipeline for learning end-to-end closed-loop policies that can reuse and benefit from data collected on other robotic platforms. We empirically demonstrate that our method, **CRADLE: Cross-Robot ADaption and aLignmEnt**, can effectively transfer new scene and task knowledge. We first show that given a shared dataset of tasks, data from other robots for a new similar task can be transferred in a zero-shot fashion. In addition, providing as few as 5 demonstrations allows our system to achieve an average of $> 70\%$ success rate on difficult tasks which cannot be learned without data from other robots. We then show that aligning internal representations across robots can help transfer across remaining domain shifts, with our method having an average of 20.8% higher success rate over an ablation without representation alignment. Finally, we show that our multiheaded training method achieves high success on 6-DoF tasks outperforming naive blocking controllers that achieve no success at all. To our knowledge, our system demonstrates the first instance of visual cross-embodiment transfer for 6-DoF tasks.

2 Related Work

Learning from large datasets. The study of utilizing diverse datasets to scale robotic learning is currently gaining momentum in the research community. Examples of large-scale real-world robot datasets include single-embodiment datasets [8, 9, 3, 10, 11, 12] and simulation benchmarks [13, 14, 15, 16]. However, these methods have typically focused on transfer with a single embodiment. To maximally reuse data from various sources, there have additionally been efforts to learn generalizable representations from sources other than robot data. One line of work is learning from human videos, including unstructured egocentric videos [17, 18, 19, 20, 21, 22, 23], egocentric videos collected with a parallel gripper device [10], and in-the-wild videos [24, 25, 26].

Apart from human videos, there are works that use representations from large-scale image datasets [27, 28, 29, 30], Youtube videos [31], and multi-source data from the Internet [32]. Although these methods can improve sample-efficiency and generalizability, they are often focused on learning visual representations that can be fine-tuned for learning robot policies. Instead, we focus directly on learning visuomotor policies from robot data.

Learning across embodiments. A number of works have studied learning policies that transfer across embodiments. Some works in the past have focused on smaller aspects of robotic transfer, including across 2-dimensional kinematic configurations [4], robotic models in simulation [5], end-effector morphologies [7], dynamics [33, 34, 35], and camera viewpoints [36]. A few works exist that focus on the task of transferring between robotic hardware. General navigation models (GNM) demonstrates cross-embodiment transfer in navigation for wheeled vehicles, quadrupeds, and drones [37]. Hu et al. study transfer to new manipulators via a robot-aware visual foresight model trained on observations from fixed exterior camera by computing a mask [6]. In contrast to these works, we focus on transfer across a wide range of high-dimensional observations, diverse robotic control schemes, and more complex 6-DoF tasks.

Transfer learning. Cross-robot transfer is closely related to the general area of transfer learning, where there is a large body of work on task generalization [38, 39, 40, 41, 42, 43, 44, 45, 32] and sim2real generalization [36, 46, 47, 48, 49, 50]. These methods either only use a single embodiment or largely ignore cross-embodiment differences. In addition, our work is closely related to the area of domain adaptation. Previous works include using GANs [51, 52, 50, 25], domain-adversarial learning [53, 54, 55], contrastive learning [17, 20], and learning system-invariant feature representations [56, 57, 58] to transfer between two related domains. Our method focuses on the domain adaptation problem across robots by exploiting a shared low-dimensional proprioceptive signal.

3 Multi-Robot Generalization Through Domain Alignment

Our goal is to maximally reuse robotic datasets collected from one setup when deploying policies on another. Let $D_r^i = \{\{(p_0, o_0, a_0), (p_1, o_1, a_1), \ldots, (p_T, o_T, a_T)\}\}$ be a dataset containing demonstrations of robot r completing task n. $p \in \mathcal{P}_r$ denotes the robot's proprioceptive end-effector pose, $o \in \mathcal{O}_r$ denotes the the image observation, and $a \in \mathcal{A}_r$ denotes the action. We assume that there exists a shared dataset $\tilde{D} := \bigcup_{n \leq N, r \leq R} D_r^n$ of experience for N different tasks and R different robots. Then, given a dataset for a new task on a given robotic platform D_j^{N+1}, $1 \leq j \leq R$, we would like to learn a policy $\pi_{k,N+2}(a|o)$ that completes the same task on robot k.

In order to reduce the domain gap between robots, we align the observation space, action space, and internal representations of our policy. In an ideal world, this would allow us to train a joint multitask policy $\tilde{\pi}(\tilde{a}|o, t)$ on $D_j^{N+1} \cup \tilde{D}$ where $\tilde{a} \in \tilde{\mathcal{A}}$ is a shared action with similar interpretation across robots. Prior work [37] demonstrates that training cross-embodiment policies with a unified abstract action space can facilitate learning cross-robot features by coupling similar states with similar signals. However, discrepancies in action interpretation across manipulators make this approach infeasible, so we instead train a task-conditioned multiheaded policy with R heads: $\pi_r(a|\tilde{f}(o), t),\ r \leq R,\ a \in A_r$ where $\tilde{f}$ is a shared encoder. Fig. 4 and Fig. 5 in the Appendix depict this architecture.

3.1 Aligning the Observation Space

Ideally, an aligned observation space $\tilde{\mathcal{O}}$ would have the property that $o_1 \in D_j^n$ is similar to $o_2 \in D_k^n$ if s_1 is similar to s_2, where s_1 and s_2 denote the ground truth states of o_1 and o_2. This corresponds to $\tilde{f}(o_1) \sim \tilde{f}(o_2)$ if $s_1 \sim s_2$. To align this space as much as possible while maintaining generality of our robotic setup, we employ 3D-printed wrist camera mounts. The cameras are mounted directly in front of the robot and positioned to capture the end-effector within their field of view. However, because of variability across robots, the wrist camera mounts are not the same across robots, nor do we standardize the camera angle. Appendix A.1 shows the locations of the wrist and exterior cameras. Utilizing a wrist camera greatly simplifies the range of variation of camera positioning to two dimensions: the height of the camera with respect to the tip of the end-effector, and the angle of

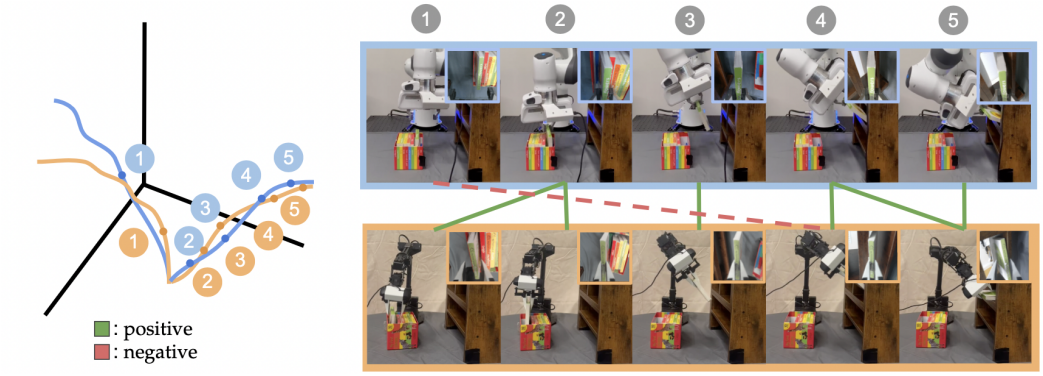

Figure 2: **Internal Representation Alignment.** This figure depicts two trajectories across different robots for the same task. Our contrastive pretraining approach maps observations with similar proprioceptive state with respect to the grasped book and cabinet together. The orange lines represent example pairs of observations mapped together, while the red line represents an example pair of observations whose embeddings are pushed apart.

the camera. This is in contrast with exterior cameras, which require six dimensions to fully specify. Secondly, since only the end-effector is in view, as seen in the *Observation Alignment* section of Fig. 1, the observation is invariant to visual differences in the robot arm.

3.2 Aligning the Action Space

To ensure a consistent action interpretation across robots, we use a shared upper-level environment. Fig. 1 depicts our control stack under the *Action Alignment* section. For an action $a_t \in \mathcal{A}_r$, we use a shared upper-level environment, responsible for processing a_t and converting it into a commanded delta pose target Δp_t^c. Then the sum of the robot's current pose and desired target pose, $p_t + \Delta p_t^c$, is transmitted to a robot-specific controller. For modularization, each robot-specific controller c_r, $1 \leq r \leq R$, exposes a small, shared API to set pose targets and query end-effector poses, which is called by the upper-level environment. Further details for these controllers are provided in Appendix A.4. In addition, we utilize a shared inverse kinematics solver to minimize inconsistencies in interpretation of pose commands. This solver also naturally aligns the coordinate frame of the robot's actions.

While the above design decisions successfully aligns the poses of the robot $p_t \in \mathcal{P}_{ri}$ and ensures consistent targets $p_t + \Delta p_t^c$ are transmitted to each robot controller, it should be noted that the resulting achieved pose at the next timestep p_{t+1} is not necessarily aligned. This discrepancy arises from the fact that the trajectory a end-effector follows to reach its target can differ based on the specific robot controller c_{r_i} and the characteristics of the robot's hardware, even when employing a shared inverse kinematics solver. In the case of continuous or non-blocking controllers, this trajectory exhibits high nonlinearity due to movement limitations imposed by the robot's kinematic configuration. As the controller interrupts the robot with a new command at regular intervals of Δt, the resulting achieved pose p_{t+1} becomes highly unpredictable given the commanded pose p_t^c and the current state. Consequently, even with a standardized control stack, the displacement Δp_t^c proves insufficient for learning a shared action space with a consistent learning signal because its interpretation differs per robot.

One attempt to circumvent this issue would be to use a blocking controller. Firstly, we would relabel the actions in our replay buffer as the change in achieved poses p and train a blocking controller to reach these poses. Since teleoperation with blocking controllers is difficult, we would instead use continuous control to collect the trajectory $\{(p_0, o_0, a_0), (p_1, o_1, a_1), \ldots, (p_T, o_T, a_T)\}$. We would then construct a new dataset $\{(p_0, o_0, \Delta p_0), (p_n, o_n, \Delta p_n), \ldots, (p_{kn}, o_{kn}, \Delta p_{kn})\}$, where $\Delta p_{kn} = p_{(k+1)n} - p_{kn}$. However, we find that even with blocking controllers, $p_t + \Delta p_t^c$ can have significant error from p_{t+1}. Due to limited degrees of freedom of the hardware and inaccuracies with the controller, not all Δp_{t+1} actions are easily reachable from the current state. As a result, one robot may have significant difficulty following trajectories from other robots for tasks that require

more complex 6-DoF motion. Due to these challenges, we instead use separate heads $\pi_r(a|\tilde{f}(o), t)$ to learn each robot's individual dynamics. By doing so, our policy can benefit from shared visual characteristics and overall movement directions, while enabling each head to learn the specific means to reach the desired goals.

3.3 Aligning Internal Representations

Aligning the internal representations of our policy can allow it to harness the advantages of training with a consistent signal, even in the absence of a unified action space. Our use of a shared kinematics solver provides a consistent proprioceptive signal p_t with respect the robot's base. As a result, two (theoretical) trajectories with same exact same motion of the end effector, $T_1 \in \mathcal{D}^n_{r_1}, T_2 \in \mathcal{D}^n_{r_2}$ will differ only by some translation (i.e. for all timesteps t and $p_{r_1,t}, p_{r_2,t} \in T_1, T_2$, $(p_{r_1,t} - p_{r_2,t}) = k$ for some constant k). Even if we do not directly act on this signal when rolling out our policy, we can still exploit it to learn better features from our observations. With this in mind, we propose to pretrain robot-agnostic features, then fine-tune to learn robot-specific controls.

In order to pretrain using the shared pose signal, we use a contrastive method that maps together similar states across trajectories and robots. We define the notion of "state similarity" by computing the changes in pose between a state and a predefined set of "fixed states" in the trajectory. Fixed states are defined as states in a trajectory that have a similar notion of task completion. For example, in quasi-static environments, these "fixed states" can be defined as states where a subtask is completed, since all successful demonstrations must contain this state. More specifically, consider a trajectory $\tau := \{(p_0, o_0, a_0), (p_1, o_1, a_1), \ldots, (p_T, o_T, a_T)\}$ and a set of fixed poses $p_f := \{p^{t_f}\}$. We define the difference between a trajectory state and fixed state by.

$$d(p_i, p_{t_f})^{xyz} = p_i^{xyz} - p_{t_f}^{xyz},\ d(p_i, p_{t_f})^{quat} = p_{t_f}^{quat}(p_i^{quat})^{-1}),$$

where p_i^{xyz} is the proprioceptive Cartesian position of the state and p_i^{quat} is the orientation of the state. Since we may have more than one fixed state for trajectory, we additionally define a closest fixed state difference $d(p_i) := d(p_i, p_{t_f})$, where t_f is the first timestep greater than or equal to i which corresponds to a fixed states. After randomly sampling an anchor batch A, for each state, observation pair $(p_a, o_a) \in A$, we uniformly sample corresponding positive and negative states $(p_+, o_+) \sim U(P_{p_a})$, $(p_-, o_-) \sim U(N_{p_a})$ by thresholding the distances between closest fixed state differences:

$$P_{p_a} := \{(p_+, o_+);\ ||d(p_a)^{xyz} - d(p_+)^{xyz}||_2^2 < \epsilon^{xyz}, \cos^{-1}(\langle d(p_a)^{quat}, d(p_+)^{quat}\rangle^2 - 1) < \epsilon^{quat}\}$$
$$N_{p_a} := \{(p_+, o_+);\ ||d(p_a)^{xyz} - d(p_-)^{xyz}||_2^2 < \epsilon^{xyz}, \cos^{-1}(\langle d(p_a)^{quat}, d(p_-)^{quat}\rangle^2 - 1) < \epsilon^{quat}\}$$

Note that $(\cos^{-1}(\langle p, q\rangle^2 - 1)$ is the geodesic distance between the two quaternions p and q. We finally use a triplet contrastive loss to pretrain our policy: $L(o_a, o_+, o_-) = \max(0, m + ||\tilde{f}_\theta(o_a) - \tilde{f}_\theta(o_+)||_2^2 - ||\tilde{f}_\theta(o_a) - \tilde{f}_\theta(o_-)||_2^2)$ where $\tilde{f}_\theta$ is the shared encoder parameterized by θ. These embeddings explicitly encourage mapping similar states together *across trajectories and robots*. This is achieved by sampling states from other rollouts in the positive buffer that correspond to similar poses with respect to a fixed pose. Fig. 2 depicts example positive and negative pair across two trajectories from different robots.

By finetuning our policy using these embeddings, we can learn robot-specific elements built upon robot-agnostic characteristics. To accomplish this, we train individualized dynamics modules $\pi_r(a|\tilde{f}(o), t)$ for each robot using a multi-headed training approach. Each "head" corresponds to a distinct action space that may vary among robots due to disparities in hardware and action interpretation. This is depicted in Fig. 1 by arrows denoting separate actions per robot. By adopting this technique, we circumvent potential challenges associated with trajectory matching involving blocking controllers, facilitating transferability across more intricate tasks that necessitate greater degrees of freedom.

4 Experiment Setup

We aim to test the following questions: (1) Does leveraging datasets from other robotic platforms enable zero-shot transfer and increase sample efficiency when learning a new task? (2) Can aligning

internal representations help bridge the domain gap across robotic platforms? (3) How does our choice of multiheaded training over robot-specific dynamics compare to using a shared action space through a blocking controller? (4) How does our decision of utilizing wrist cameras compare to the past approaches of collecting data with exterior cameras? Videos of experiments can be found on our project website: https://sites.google.com/view/cradle-multirobot

Robotic platforms. We evaluate our method with the WidowX 250S, Franka Emika Panda, and Sawyer robot arms. All three of the robots are controlled with 6-DoF delta joint position control. For each robot, we collect 64 by 64 image observations from two sources: an exterior camera pointing at the robot, and a wrist camera mounted on the arm itself.

Evaluation tasks. We assume access to a shared buffer containing experience for at least one task on all three robots. Then, we collect data for a target task on two robots that we want to transfer to a third. For zero-shot evaluation, we directly train on this data and evaluate on the third robot. We also test sample-efficiency of learning a new task with few-shot experiments, where we teleoperate 5 demonstrations for the new task on the third robot.

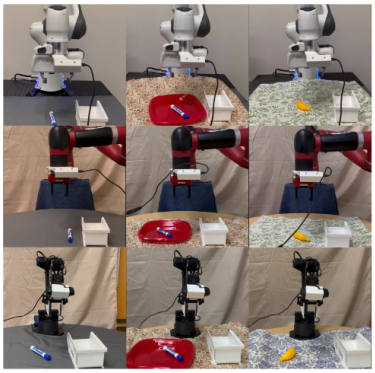

Figure 3: **Pick/Place Tasks.** The left column contains the shared pick/place task, while the other columns contain the new distractor and new object variants. For zero-shot evaluation, we include data for the shared task across all robots. For few-shot, we also include 5 demonstrations of a variant for one robot.

To effectively leverage data from prior tasks to solve new tasks, the prior tasks must share structural similarity with the new tasks. With this in mind, we evaluate the robot on variants for two types of tasks: standard pick/place, and shelf manipulation. In the pick/place task, the robot needs to pick up a marker and drop it into a white container from a variety of positions. In the shelf manipulation task, the robot rearranges a book from a container onto a shelf. The first set of tasks was chosen to evaluate multi-robot transfer in environments with simple dynamics and greater similarity in control from one robot to another. The second set evaluates transfer for environments that require greater degrees of motion. Each of the tasks is evaluated on 10 starting locations for the objects.

For each type of task, we test for both *scene generalization* and *task generalization*. We define scene generalization as generalization to a task for which there exists an mapping of observations to a previously seen task. More formally, given an optimal policy $\pi^*(a|o)$ for a previous task T_1, a new task T_2 only differs with T_1 by scene if there is a function $f : O \to O$ such that $\pi^*(a|f(o))$ solves T_2. Task generalization is more general than scene generalization, applying to new tasks which are similar to an old one, but may require different actions and controls to solve. The following are the variations that we evaluate on:

Scenario 1 (S1): *New Distractor Pick/Place*: Pick/place with a distractor object in the background
Scenario 2 (S2): *New Object Pick/Place*: Pick/place with a banana instead of a marker
Scenario 3 (S3): *New Orientation Shelf*: Shelf Manipulation with a reversed orientation of the original container that contains the books. The motion to grasp and place the book is the same.
Scenario 4 (S4): *New Compartment Shelf*: Shelf Manipulation putting a book into a new compartment on the bottom

Comparison methods. For few-shot transfer, we compare our main method (denoted are *Contrastive + Multiheaded* or *CRADLE* to two different baselines. The first is *Naive Multi-Robot Training*, which trains a task-conditioned multiheaded policy on exterior camera angles without contrastive pretraining. We also evaluate the *Single Robot Training* baseline, which only contains demonstrations for the target robot. We also consider two different ablations for our method: task-conditioned multiheaded training without constrastive pretraining denoted as *Ours w/o Contr.*, and constrastive pretraining with a blocking controller denoted as *Contr. + Blocking*.

Dataset collection. Details are provided in Appendix A.2.

Robot	Method	New Distr. (S1)	New Obj. (S2)	New Orient. (S3)	New Comp. (S4)
		Pick and place		Shelf	
Franka	CRADLE	**0.9**	**0.8**	**1.0**	**0.9**
	Naive Mult-Robot Training	0.4	0.3	0.0	0.0
	Single Robot Training	0.2	0.2	0.0	0.0
Sawyer	CRADLE	**0.9**	**0.9**	**0.9**	**0.7**
	Naive Mult-Robot Training	0.3	0.2	0.0	0.0
	Single Robot Training	0.2	0.1	0.0	0.0
WidowX	CRADLE	**0.9**	**1.0**	**0.8**	**0.7**
	Naive Mult-Robot Training	0.4	0.1	0.0	0.0
	Single Robot Training	0.3	0.2	0.0	0.0

Table 1: **Few-shot multi-robot transfer results.** Given 5 demonstrations on a new task, CRADLE performs significantly better than a baseline without data from other robots. In addition, our task-conditioned multiheaded policy enables the transfer of multi-robot data for shelf manipulation tasks, where a blocking controller fails.

5 Experimental Results

Utilizing data from other robots signficantly improves few-shot generalization performance on a new task. In Table 1, the success rate for CRADLE demonstrates significant improvement on all tasks and all robots over single robot training. On the Franka, the results show an average of 0.5 higher success rate on *Pick/Place* (0.9, 0.7 versus 0.4, 0.3) and 0.95 higher success rate on the Shelf Manipulation. (0.8, 0.9 versus 0.0, 0.0). This indicates that CRADLE can effectively utilize data from other robots to learn a new task with high sample efficiency. (talk about zero performance). Notably, single-robot training fails to get any success on Shelf Manipulation. Qualitatively, we observe that this policy quickly falls out-of-distribution and in unable to grasp the book. For tasks with high structural similarity to those in the shared buffer such as *New Distractor Pick/Place*, training without other robot data has nonzero performance, likely due to the variation in scenes we collect in our shared dataset. However, 5 demonstrations are not enough to cover the entire distribution of object positions for a new task, leading to suboptimal performance.

Robot	Method	S1	S2	S3	S4
Franka	CRADLE	**0.4**	**0.6**	**0.4**	0.0
	Contr. + Blocking	0.3	0.3	0.0	0.0
Sawyer	CRADLE	**0.6**	**0.5**	**0.4**	0.0
	Contr. + Blocking	**0.6**	0.3	0.0	0.0
WidowX	CRADLE	**0.4**	**0.4**	**0.5**	0.0
	Contr. + Blocking	**0.4**	0.3	0.0	0.0

Table 2: **Zero-shot results.** CRADLE can learn a new task with high structural similarity to tasks in a shared multi-robot buffer given only data from other robots.

CRADLE facilitates better transfer than naive task-conditioned multiheaded training on with exterior cameras. Table 1 shows significantly higher success rates for CRADLE over naive multi-robot training (0.9, 0.8, 1.0, 1.0 versus 0.4, 0.2, 0.0, 0.0 for the Franka). This suggests that task-conditioned multiheaded training on exterior camera struggles to utilize data from other robots, validating our hypothesis that a standardized observation space is crucial for this transfer to occur. For *Shelf Manipulation* scenarios, naive multi-robot training fails to achieve any success*Pick/Place* scenarios, this method does have some success, but it is likely due to the similarities between the new task and other tasks from the same robot.

Utilizing data from other robots allows for zero-shot generalization performance on a new task. For tasks that have higher structural similarity to that scene in previous data for the robot such as scene generalization tasks described in the experiment setup section, we see good success rate for zero-shot multi-robot transfer. Table 2 showcases this, with both CRADLE having success rates of 0.4, 0.6, 0.4 and the blocking controller having success rates of 0.3, 0.6, 0.4 for the New Distractor Pick/Place task. Without data from other robots, the success rate would be near 0 because the new task would not be present in the replay buffer. For Scenario 3, or *New Orientation Shelf*, CRADLE has nonzero performance while the blocking method doesn't. This is because while both methods are able to pick up the book from the reversed bookshelf, the blocking controller struggles with placing the book onto the shelf.

Robot	Method	S1	S2	S3	S4
Franka	CRADLE	**0.9**	**0.8**	**1.0**	**0.9**
	CRADLE w/o Contr.	0.8	0.5	0.6	0.7
Sawyer	CRADLE	**0.9**	**0.9**	**0.9**	**0.7**
	CRADLE w/o Contr.	0.7	0.5	0.6	0.7
WidowX	CRADLE	**0.9**	**1.0**	**0.8**	**0.7**
	CRADLE w/o Contr.	0.8	0.7	0.6	0.7

Table 3: **Ablation: ours vs ours without contrastive.** Our contrastive pretraining and multiheaded finetuning approach provides an average of 20.8% improvement on few-shot transfer for a new task over regular multiheaded training.

Method	S1	S2	S3	S4
CRADLE	**0.9**	**0.8**	**1.0**	**0.9**
Contr. + Blocking	0.8	0.6	0.0	0.0
CRADLE	**0.9**	**0.9**	**0.9**	**0.7**
Contr. + Blocking	**0.9**	**0.9**	0.0	0.0
CRADLE	**0.9**	**1.0**	**0.8**	**0.7**
Contr. + Blocking	0.8	0.9	0.0	0.0

Table 4: **Ablation: Ours vs contrastive + blocking** Although a blocking controller has similar few-shot performance to CRADLE on Pick/Place, it struggles in shelf manipulation scenarios

Aligning the internal representations of the observations between robots leads to learning more generalizable features. Table 3 compares the performances of CRADLE with an ablation of the contrastive pretraining phase. Aligning the internal representation of the policy causes an average of 20.8% increase in performance over. This suggests that aligning the internal representation of the policy assists in learning features that generalize across robots. We hypothesize that training on an aligned proprioceptive signal across robots. Notably, multiheaded training alone seems to have reasonable performance on all tasks, albiet lower than our method.

Multiheaded policies can better learn from 6-DoF, cross-robot demonstrations over blocking controllers. Table 4 shows that on the *Shelf Manipulation tasks*, CRADLE achieves high success rates $1.0, 0.9$ on the Franka compared to zero success rate on the blocking controller. This large discrepancy is explained by the inability of one robot to precisely imitate another robot. For instance, variations in the length of the wrist link can lead to large discrepancies in the radius of the rotation necessary to perform the *Shelf Manipulation* task. To verify this, we provide the error over timestep of imitating each task in the Appendix. Notably, the blocking controller has high performance on *Pick/Place* tasks due to lower discrepancy between robots on translational movement.

6 Discussion

Summary. We have developed a method, CRADLE, that efficiently learns new tasks using data collected on other robots. This is enabled through careful design decisions driven by a fundamental observation: transferring data across domains requires aligning the domains as much as possible without making assumptions that limit their applicability. For example, CRADLE uses wrist cameras, which can be mounted on a wide range of robots, while exhibiting significantly less variation than exterior cameras, even if the mounting position is not fixed. In addition, CRADLE uses a shared higher-level action representation and varying lower-level controller, which can align a policy's action interpretation while accommodating the diverse landscape of existing robotic setups. Finally, our choice of contrastive loss can align internal representations across robots. Despite the simplicity of each individual design decision, their combination sufficiently aligns our multi-embodiment dataset to enable cross-embodiment transfer. CRADLE achieves over a 70% success rate on all tasks, outperforming naive multi-robot training.

Limitations and Future Work. One limitation of our approach is that our method requires a shared dataset between robots to learn their correspondences. As a result, our method is not able to transfer policies to a new robot with no demonstrations. Our method also does not allow for zero-shot transfer on tasks with significantly different motion than seen in the shared dataset. In the future, we plan to scale the datasets for each robot, as we believe that can allow for more new tasks to be in-distribution.

References

[1] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009. doi:10.1109/CVPR.2009.5206848.

[2] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever. Improving language understanding by generative pre-training. 2018.

[3] S. Dasari, F. Ebert, S. Tian, S. Nair, B. Bucher, K. Schmeckpeper, S. Singh, S. Levine, and C. Finn. Robonet: Large-scale multi-robot learning. In *Conference on Robot Learning*, 2019.

[4] C. Devin, A. Gupta, T. Darrell, P. Abbeel, and S. Levine. Learning modular neural network policies for multi-task and multi-robot transfer. In *International Conference on Robotics and Automation*, 2016.

[5] T. Chen, A. Murali, and A. Gupta. Hardware conditioned policies for multi-robot transfer learning, 2019.

[6] E. S. Hu, K. Huang, O. Rybkin, and D. Jayaraman. Know thyself: Transferable visual control policies through robot-awareness. In *International Conference on Learning Representations*, 2022.

[7] G. Salhotra, xI Chun Arthur Liu, and G. Sukhatme. Bridging action space mismatch in learning from demonstrations. 2023.

[8] P. Sharma, L. Mohan, L. Pinto, and A. Gupta. Multiple interactions made easy (mime): Large scale demonstrations data for imitation, 2018.

[9] A. Mandlekar, Y. Zhu, A. Garg, J. Booher, M. Spero, A. Tung, J. Gao, J. Emmons, A. Gupta, E. Orbay, S. Savarese, and L. Fei-Fei. Roboturk: A crowdsourcing platform for robotic skill learning through imitation. 2018.

[10] S. Young, D. Gandhi, S. Tulsiani, A. Gupta, P. Abbeel, and L. Pinto. Visual imitation made easy. 2020.

[11] E. Jang, A. Irpan, M. Khansari, D. Kappler, F. Ebert, C. Lynch, S. Levine, and C. Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, 2021.

[12] F. Ebert, Y. Yang, K. Schmeckpeper, B. Bucher, G. Georgakis, K. Daniilidis, C. Finn, and S. Levine. Bridge data: Boosting generalization of robotic skills with cross-domain datasets. In *Robotics: Science and Systems*, 2022.

[13] T. Yu, D. Quillen, Z. He, R. Julian, A. Narayan, H. Shively, A. Bellathur, K. Hausman, C. Finn, and S. Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Conference on Robot Learning*, 2019.

[14] S. James, Z. Ma, D. R. Arrojo, and A. J. Davison. Rlbench: The robot learning benchmark & learning environment. 2019.

[15] Y. Zhu, J. Wong, A. Mandlekar, R. Martín-Martín, A. Joshi, S. Nasiriany, and Y. Zhu. robosuite: A modular simulation framework and benchmark for robot learning. In *arXiv preprint arXiv:2009.12293*, 2020.

[16] J. Fu, A. Kumar, O. Nachum, G. Tucker, and S. Levine. D4rl: Datasets for deep data-driven reinforcement learning. 2021.

[17] P. Sermanet, C. Lynch, Y. Chebotar, J. Hsu, E. Jang, S. Schaal, S. Levine, and G. Brain. Time-contrastive networks: Self-supervised learning from video. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1134–1141, 2018. doi:10.1109/ICRA.2018.8462891.

[18] I. Radosavovic, T. Xiao, S. James, P. Abbeel, J. Malik, and T. Darrell. Real-world robot learning with masked visual pre-training. 2022.

[19] K. Grauman et al. Ego4d: Around the world in 3,000 hours of egocentric video, 2022.

[20] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta. R3m: A universal visual representation for robot manipulation. In *Conference on Robot Learning*, 2022.

[21] Y. J. Ma, S. Sodhani, D. Jayaraman, O. Bastani, V. Kumar, and A. Zhang. Vip: Towards universal visual reward and representation via value-implicit pre-training. In *International Conference on Learning Representations*, 2023.

[22] A. Majumdar, K. Yadav, S. Arnaud, Y. J. Ma, C. Chen, S. Silwal, A. Jain, V.-P. Berges, P. Abbeel, J. Malik, D. Batra, Y. Lin, O. Maksymets, A. Rajeswaran, and F. Meier. Where are we in the search for an artificial visual cortex for embodied intelligence?, 2023.

[23] S. Karamcheti, S. Nair, A. S. Chen, T. Kollar, C. Finn, D. Sadigh, and P. Liang. Language-driven representation learning for robotics. 2023.

[24] A. S. Chen, S. Nair, and C. Finn. Learning generalizable robotic reward functions from "in-the-wild" human videos. 2021.

[25] S. Bahl, A. Gupta, and D. Pathak. Human-to-robot imitation in the wild. 2022.

[26] L. Shao, T. Migimatsu, Q. Zhang, K. Yang, and J. Bohg. Concept2robot: Learning manipulation concepts from instructions and human demonstrations. 2020.

[27] R. Shah and V. Kumar. Rrl: Resnet as representation for reinforcement learning. 2021.

[28] L. Yen-Chen, A. Zeng, S. Song, P. Isola, and T.-Y. Lin. Learning to see before learning to act: Visual pre-training for manipulation. 2021.

[29] M. Sharma, C. Fantacci, Y. Zhou, S. Koppula, N. Heess, J. Scholz, and Y. Aytar. Lossless adaptation of pretrained vision models for robotic manipulation. 2023.

[30] C. Wang, X. Luo, K. Ross, and D. Li. Vrl3: A data-driven framework for visual deep reinforcement learning. 2023.

[31] M. Chang, A. Gupta, and S. Gupta. Semantic visual navigation by watching youtube videos. 2020.

[32] S. Reed et al. A generalist agent. *Transactions on Machine Learning Research*, 2022.

[33] P. Christiano, Z. Shah, I. Mordatch, J. Schneider, T. Blackwell, J. Tobin, P. Abbeel, and W. Zaremba. Transfer from simulation to real world through learning deep inverse dynamics model. In *CoRR*, 2016.

[34] W. Yu, J. Tan, C. K. Liu, and G. Turk. Preparing for the unknown: Learning a universal policy with online system identification. In *CoRR*, 2017.

[35] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *International Conference on Robotics and Automation*. IEEE, 2018.

[36] F. Sadeghi, A. Toshev, E. Jang, and S. Levine. Sim2real view invariant visual servoing by recurrent control. In *International Conference on Robotics and Automation*, 2017.

[37] D. Shah, A. Sridhar, A. Bhorkar, N. Hirose, and S. Levine. Gnm: A general navigation model to drive any robot. In *International Conference on Robotics and Automation*, 2023.

[38] E. Parisotto, J. L. Ba, and R. Salakhutdinov. Actor-mimic: Deep multitask and transfer reinforcement learning, 2016.

[39] Y. Duan, M. Andrychowicz, B. C. Stadie, J. Ho, J. Schneider, I. Sutskever, P. Abbeel, and W. Zaremba. One-shot imitation learning, 2017.

[40] C. Finn, T. Yu, T. Zhang, P. Abbeel, and S. Levine. One-shot visual imitation learning via meta-learning, 2017.

[41] K. Hausman, J. T. Springenberg, Z. Wang, N. Heess, and M. Riedmiller. Learning an embedding space for transferable robot skills. In *International Conference on Learning Representations*, 2019.

[42] Z. Xu, K. Wu, Z. Che, J. Tang, and J. Ye. Knowledge transfer in multi-task deep reinforcement learning for continuous control, 2020.

[43] S. Sodhani, A. Zhang, and J. Pineau. Multi-task reinforcement learning with context-based representations. In *International Conference on Machine Learning*, 2021.

[44] A. Brohan et al. Rt-1: Robotics transformer for real-world control at scale, 2022.

[45] M. Ahn et al. Do as i can, not as i say: Grounding language in robotic affordances. 2022.

[46] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. 2017.

[47] Z.-W. Hong, C. Yu-Ming, S.-Y. Su, T.-Y. Shann, Y.-H. Chang, H.-K. Yang, B. H.-L. Ho, C.-C. Tu, Y.-C. Chang, T.-C. Hsiao, H.-W. Hsiao, S.-P. Lai, and C.-Y. Lee. Virtual-to-real: Learning to control in visual semantic segmentation. In *International Joint Conferences on Artificial Intelligence*, 2018.

[48] Y. Chebotar, A. Handa, V. Makoviychuk, M. Macklin, J. Issac, N. Ratliff, and D. Fox. Closing the sim-to-real loop: Adapting simulation randomization with real world experience. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8973–8979, 2019. doi:10.1109/ICRA.2019.8793789.

[49] M. Kaspar, J. D. M. Osorio, and J. Bock. Sim2real transfer for reinforcement learning without dynamics randomization, 2020.

[50] D. Ho, K. Rao, Z. Xu, E. Jang, M. Khansari, and Y. Bai. Retinagan: An object-aware approach to sim-to-real transfer, 2021.

[51] K. Bousmalis, A. Irpan, P. Wohlhart, Y. Bai, M. Kelcey, M. Kalakrishnan, L. Downs, J. Ibarz, P. Pastor, K. Konolige, S. Levine, and V. Vanhoucke. Using simulation and domain adaptation to improve efficiency of deep robotic grasping. 2017.

[52] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2242–2251, 2017. doi:10.1109/ICCV.2017.244.

[53] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks, 2016.

[54] K. Fang, Y. Bai, S. Hinterstoisser, S. Savarese, and M. Kalakrishnan. Multi-task domain adaptation for deep learning of instance grasping from simulation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3516–3523, 2018. doi:10.1109/ICRA.2018.8461041.

[55] M. Xu, M. Islam, C. M. Lim, and H. Ren. Learning domain adaptation with model calibration for surgical report generation in robotic surgery. In *International Conference on Robotics and Automation*, 2021.

[56] A. Gupta, C. Devin, Y. Liu, P. Abbeel, and S. Levine. Learning invariant feature spaces to transfer skills with reinforcement learning, 2017.

[57] N. H. Kim, Z. Xie, and M. van de Panne. Learning to correspond dynamical systems, 2020.

[58] S. J. Wang and A. M. Johnson. Domain adaptation using system invariant dynamics models. In S. J. Wang and A. M. Johnson, editors, *Proceedings of the 3rd Conference on Learning for Dynamics and Control*, volume 144 of *Proceedings of Machine Learning Research*, pages 1130–1141. PMLR, 07 – 08 June 2021. URL https://proceedings.mlr.press/v144/wang21c.html.