Robotics: Science and Systems 2025
Los Angeles, CA, USA, June 21–25, 2025

Sim-and-Real Co-Training: A Simple Recipe for Vision-Based Robotic Manipulation

Abhiram Maddukuri[*,1], Zhenyu Jiang[*,1,2], Lawrence Yunliang Chen[*2,3], Soroush Nasiriany[*1,2], Yuqi Xie[2], Yu Fang[2], Wenqi Huang[2], Zu Wang[2,4], Zhenjia Xu[2], Nikita Chernyadev[2], Scott Reed[2], Ken Goldberg[3], Ajay Mandlekar[†,2], Linxi Fan[†,2], Yuke Zhu[†,1,2]

[1]UT Austin [2]NVIDIA [3]UC Berkeley [4]New York University

Abstract*—Large real-world robot datasets hold great potential to train generalist robot models, but scaling real-world human data collection is time-consuming and resource-intensive. Simulation has great potential in supplementing large-scale data, especially with recent advances in generative AI and automated data generation tools that enable scalable creation of robot behavior datasets. However, training a policy solely in simulation and transferring it to the real world often demands substantial human effort to bridge the reality gap. A compelling alternative is to *co-train* the policy on a mixture of simulation and real-world datasets. Preliminary studies have recently shown this strategy to substantially improve the performance of a policy over one trained on a limited amount of real-world data. Nonetheless, the community lacks a systematic understanding of sim-and-real co-training and what it takes to reap the benefits of simulation data for real-robot learning. This work presents a simple yet effective recipe for utilizing simulation data to solve vision-based robotic manipulation tasks. We derive this recipe from comprehensive experiments that validate the co-training strategy on various simulation and real-world datasets. Using two domains—a robot arm and a humanoid—across diverse tasks, we demonstrate that simulation data can enhance real-world task performance by an average of 38%, even with notable differences between the simulation and real-world data. Videos and additional results can be found at co-training.github.io.

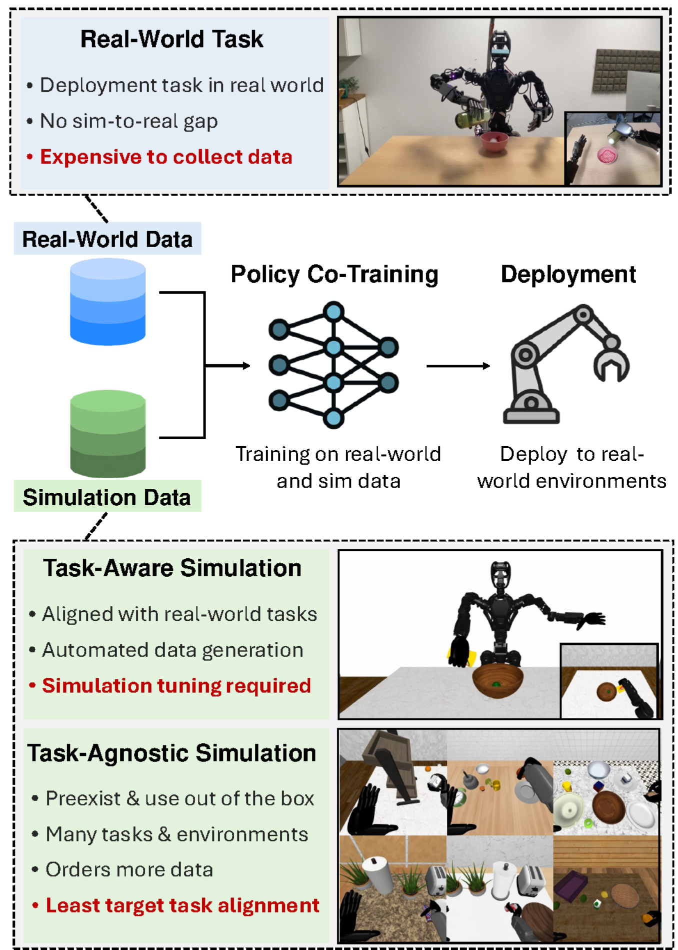

Fig. 1: **Sim-and-Real Co-Training.** We show how co-training policies on real-world and simulation data can attain superior performance in the real-robot deployment, compared to training solely on real-world data. We specifically study two forms of simulation data: (1) task-aware data from *digital cousins* built with knowledge of the real-world tasks, and (2) task-agnostic data from multi-task *prior simulations* covering more diverse settings but with less alignment to the target task.

I. Introduction

The ability to generalize across diverse environments and tasks is a critical step toward realizing generalist robotic systems. Recent advancements in robot foundation models [1]–[3]—trained on a mixture of web-scale vision-language datasets and robot-specific datasets—have demonstrated significant potential for cross-domain generalization. Large real-world robot datasets [4], [5] embody this diversity and are a crucial source of data. Despite this progress, challenges remain in achieving reliable real-world deployment. To bridge this gap, there have recently been multiple efforts on collecting even larger-scale real-robot datasets [1], [6]. These works have shown the potential of data-driven methods in acquiring versatile robotic skills. However, they involve considerable cost, time, and scalability challenges, and it remains unclear whether simply scaling real-world data collection alone is sufficient to train generalist robot models.

Simulation is a promising alternative to mitigate the data hunger of large models. The recent proliferation of generative AI tools allows for the automated generation of assets, scenes, and tasks in simulation, all of which can be produced with high-fidelity physics simulators and photorealistic renderers [7], [8]. Furthermore, automated data generation tools can be applied to these simulation environments to synthesize large amounts of diverse, high-quality robot trajectories with

*Equal contribution. †Project leads.

minimal human effort [9]–[12], offering massive training data for generalist manipulation policies. However, approaches that use simulation data must deal with the reality gap since the visuals and physics in simulation do not align perfectly with the real world. Prior approaches on sim-to-real policy transfer typically rely on extensive tuning of simulation to match the real world [13]–[17], or meticulously randomizing a specific set of simulation parameters [18]–[22]. Such approaches can require significant human effort.

A compelling alternative to sim-to-real transfer is to directly co-train policies on a *mixture* of simulation and real-world data. Preliminary findings in recent work [7], [23]–[25] suggest that incorporating simulation data in this way can greatly improve policy performance compared to using real-world data alone. Moreover, sim-and-real co-training may not require the high level of alignment between simulation and reality typically needed for sim-to-real transfer, making it a promising strategy to tap into the potential of large synthetic datasets with minimal human effort. Despite its promise, the community lacks a systematic understanding of this strategy and what it takes to reap the benefits of simulation data for real-robot learning. It remains unclear how different the simulation data can be from the real-world data, and what kinds of dataset mixtures and compositions are ideal.

This work presents a simple recipe for supplementing real robot datasets with synthetic simulation datasets to facilitate learning vision-based manipulation policies for real robots. We derive this recipe by conducting a comprehensive set of experiments that co-train robot policies on various simulation and real-world datasets. As shown in Figure 1, we focus on two concrete sources of simulation data — task-aware simulation, in which the simulation environment is intentionally designed to align with the real world loosely, akin to the "digital cousin" concept introduced in Dai et al. [26], and task-agnostic simulation, which comprises prior simulation data made independently of the particular target task. Our study is carried out on two distinct robot embodiments (robot arm and humanoid) across several diverse tasks, spanning pick-and-place, articulated object manipulation, and non-prehensile manipulation (*e.g.*, pouring). We investigate a number of critical data composition factors to understand the degree to which simulation and real-world data must be aligned for co-training to be effective. For example, should the tasks, scenes, and objects be the same between the sim and the real? How about the location of workspace cameras and object placements? We make use of synthetic data generation tools [9], [10] to test different simulation dataset compositions with ease, resulting in actionable insights for robotics practitioners.

We summarize our contributions as follows:

1) We establish a systematic study for co-training on real-robot data and synthetically generated data from simulation, resulting in a simple recipe to leverage synthetic simulation data for real-world manipulation;
2) We demonstrate empirically how co-training on synthetic simulation data can be broadly useful in facilitating policy learning for downstream real-world tasks, improving policy performance across two domains by an average of 38%;
3) We derive insight into what types of simulation data are most effective for sim-and-real co-training. Surprisingly, we find that simulation data provides substantial benefits even with notable differences from the real-world data, and that diverse simulation data can facilitate generalization to unseen scenarios in the real world.

II. Related Work

A. Learning Manipulation from Demonstration Data

Behavior cloning [27] is a widely adopted approach for learning robot policies from demonstration data [28]–[38]. In this framework, policies are trained to predict actions based on ground truth state-action pairs provided in a demonstration dataset. This method has been extensively applied in robot manipulation tasks [39]–[50]. However, its success in real-world applications typically hinges on the availability of large amounts of high-quality demonstration data, which can be prohibitively expensive to collect. To address this limitation, our work explores the use of simulation data to enhance real-world robot manipulation through imitation learning, thereby reducing the dependency on costly real-world data collection.

B. Sim-to-Real and Sim-Real Co-Training

Sim-to-real transfer has been a pivotal focus in robotics research, aimed at enabling models trained in simulation to perform effectively in the real world. One popular approach is domain randomization [18]–[22], [51], [52], which introduces variability into the simulation environment to train policies that are robust to discrepancies between simulation and reality. However, domain randomization approaches can require careful tuning and a significant human burden to determine proper randomization ranges for the parameters that enable the policy to transfer to the real world. Another common approach seeks to minimize the sim-to-real gap by improving simulation fidelity to match the real world closely. Techniques such as system identification [13]–[17], [53]–[55] and the creation of digital twins [56], [57] aim to align simulated dynamics more closely with real-world conditions. These methods often require significant human effort, preventing their applicability to diverse tasks and environments. Instead, recent research has trained real-world manipulation policies using a mixture of simulation and real-world data [3], [7], [23], [24], [58], and demonstrated superior performance to solely using the same quantity of real-world data. Furthermore, the simulation data in these approaches does not necessarily need to be perfectly aligned with the real world, making it a compelling alternative to other approaches. Building on these findings, our work systematically investigates sim-and-real co-training. We examine the effectiveness of co-training real-world policies using two sources of simulation data with varying levels of alignment: task-aware digital cousins [26] and task-agnostic prior simulation, as shown in Figure 1. Our study offers practical insights and methodologies for practitioners aiming to achieve robust performance in real-world scenarios.

C. Dataset Composition in Robot Learning

Recent research [59], [60] has highlighted the importance of dataset composition in robot learning, particularly in understanding how variations in data quality and diversity influence policy generalization [61], [62]. Studies such as MimicLabs [63] have conducted large-scale analyses to identify the types of data that maximize the utility of robotic datasets and improve downstream policy performance. Inspired by this line of work, we investigate the optimal composition of simulation and real-world data specifically for real-world robotic manipulation tasks. Our study aims to provide actionable guidelines on how to strategically combine these data sources to achieve superior policy learning outcomes in the real world.

III. Problem Statement and Preliminaries

A. Co-Training on Real-World and Simulation Data

We assume access to robot trajectory demonstrations collected in real environments, $\mathcal{D}_{\text{real}} = \{\xi_i\}_{i=1}^N$. Instead of training a policy on demonstrations solely from the real world, we have additional demonstrations from simulation environments, $\mathcal{D}_{\text{sim}} = \{\xi_i\}_{i=1}^M$, where generally $M \gg N$. We train a visuomotor policy π_θ on these two data sources. We adopt the *co-training* formulation following prior work [7], where we minimize the behavioral cloning action loss

$$\mathcal{L}_{\text{total}}(\theta; \mathcal{D}_{\text{real}}, \mathcal{D}_{\text{sim}}) = \alpha \cdot \mathcal{L}(\theta; \mathcal{D}_{\text{sim}}) + (1-\alpha) \cdot \mathcal{L}(\theta; \mathcal{D}_{\text{real}}) \quad (1)$$

where $\mathcal{L}(\theta; \mathcal{D}) = \frac{1}{|\mathcal{D}|} \sum_{(o_i, a_i) \in \mathcal{D}} -\log \pi_\theta(a_i|o_i)$ and $\alpha \in [0, 1]$ is the *co-training ratio* balancing the relative weight of simulation and real-world data. In practice, we use an equivalent formulation of α, which represents the probability of sampling from simulation data in each training batch. Specifically, we reweight each sample (o_i, a_i) such that the probability of drawing it from the simulation dataset is $P[(o_i, a_i) \in \mathcal{D}_{\text{sim}}] = \alpha$, while the probability of drawing it from the real dataset is $P[(o_i, a_i) \in \mathcal{D}_{\text{real}}] = 1 - \alpha$ during training batch sampling. We further detail the relative weighting procedure in Appendix VIII-G. As we will see in experiments, the choice of α is crucial to policy performance. Our end objective is to produce vision-based manipulation policies that maximize task performance on one or multiple downstream tasks in real-world environments.

B. Data Composition Factors

$\mathcal{D}_{\text{real}}$ and $\mathcal{D}_{\text{sim}}$ can comprise demonstration trajectories from either a single task or a diverse array of tasks, embodiments, and environments. To reason about how the particular choices in constructing these datasets can affect the success of co-training, it is useful to decompose these datasets into a set of *data composition factors*. We assume that each dataset follows a distribution of factors $\{\mathcal{Z}^{(1)}, \mathcal{Z}^{(2)}, \cdots, \mathcal{Z}^{(K)}\}$, borrowing notation from recent work [63]. We do not assume that the simulation dataset is perfectly aligned with the real-world dataset, *i.e.*, for some factors $\mathcal{Z}_{\text{sim}}^{(i)} \neq \mathcal{Z}_{\text{real}}^{(i)}$. Despite these alignment gaps, we are interested in transferring knowledge from simulation domains to learn a more effective policy π for real-world tasks.

Common data composition factors include, but are not limited to, the following:

- **Task composition:** Which tasks, and by extension, subtasks and motions, are present in the simulation and real-world data. Even if the real-world and simulation data involve solving the same task, there may be multiple valid ways to achieve the task. Different datasets may exhibit different orderings of subtasks and different manipulation skills;
- **Scene composition:** The number of scenes in simulation and real-world data, in addition to the scope and diversity of various components across these scenes. For example, the number of fixtures, articulation properties of interactive objects, range of lighting conditions, and range of background textures;
- **Object composition:** Which object categories are present in the simulation and real-world data, and the number of unique object instances per object category;
- **Initialization distribution:** The initial state distribution in the datasets, representing the distribution of states in the initial state of each trajectory in the dataset. This comprises the initial robot base pose and arm joints, in addition to the initialization distribution of objects and fixtures in the scene;
- **Camera parameters:** We assume that we have a set of N cameras used to train each visuomotor agent. Each camera has a range of values across several parameters. The most prominent parameters are camera intrinsics and camera extrinsics;
- **Dynamics parameters:** Key physical parameters such as friction, mass of objects, and inertia. Other parameters include robot controller variables such as the type of controller and its gains.

We define these parameters in more detail and quantify them in Section IV, when we introduce the domains and tasks, and we study how important it is to align each factor between simulation and the real world for co-training success.

C. Automated Synthetic Data Generation

A key advantage of simulation is ease of data collection—we leverage automated synthetic data generation tools to generate large, high-quality simulation datasets and use them for co-training with smaller real-world datasets. For each task in the simulation, we first collect dozens of source human demonstrations. We then use MimicGen [9] to generate large synthetic trajectory datasets at scale. For bimanual and humanoid robots, we use DexMimicGen [10], a method that builds on top of MimicGen. The process is as follows. First, we segment these source demonstrations into a sequence of object-centric segments. (Dex)MimicGen then generates new demonstrations by applying linear transformations to selected source demonstration segments and concatenating these transformed segments to form novel trajectories. By leveraging these methods, we can use physics simulations to multiply the number of trajectories by orders of magnitude.

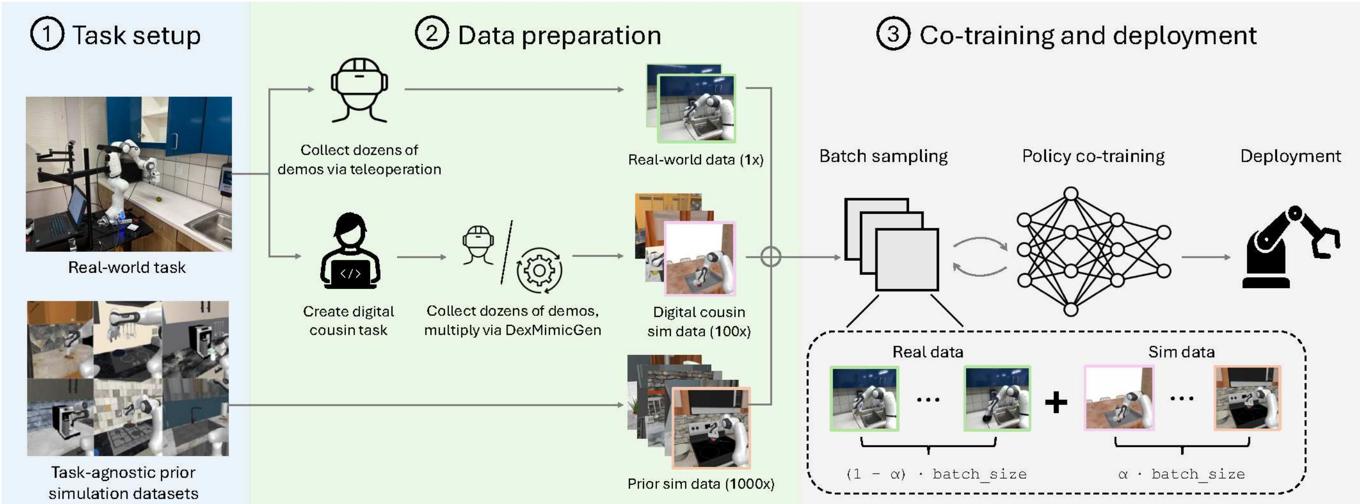

Fig. 2: **Method Overview.** Our workflow consists of three components: (1) We start with a real-world target task in mind and some prior simulation data; (2) Given real-world tasks and prior simulation data, we build simulated digital cousin environments that share semantic similarities with their real-world counterparts but may still hold discrepancies in visual and physical aspects. We leverage synthetic data generation methods to multiply trajectories in digital cousins, producing a large quantity of demonstrations in simulation. From here, we consolidate prior simulation data, digital cousin data, and real-world data; (3) We co-train the policy on a mixture of real-world and simulation data. We sample simulation data according to a sampling ratio of α, which is crucial for the method's effectiveness. After training the policy, we deploy the learned policy directly to the real robot.

IV. Study Setup

Our goal is to develop a simple recipe for co-training on real-robot and simulation data to significantly improve real-world policy performance compared to training on real data alone. We are broadly interested in two scenarios:

(1) **Co-training with prior large-scale simulation data.** Can we use existing large prior simulation datasets as co-training data? Note, these datasets often have significant discrepancies with the real world in terms of visual features, task semantics, and behaviors. We are interested in understanding the extent to which these datasets can help *out of the box* in learning downstream real-world tasks in spite of these domain gaps.

(2) **Co-training with task-aware simulation data.** Given knowledge of the real-world tasks, we can create customized simulation datasets that are better aligned with the real-world tasks. However, tuning simulation environments to match the real-world environments precisely is impractical. Which data composition factors are most important to align between simulation and the real-world setup, and can we forgo *perfect alignment*, allowing us to reduce human effort?

See Figure 2 for an overview of our workflow, including the real-world setup, simulation pipeline, and the co-training procedure. We describe all of these components in detail in the following sections.

A. Real-World Domains

We seek a co-training recipe that is broadly applicable to a wide range of embodiments, tasks, and environments. To this end, we conduct a comprehensive study featuring two distinct domains, each with a unique robot embodiment and diverse tasks (see Figure 3 for an illustration):

Panda Kitchen. A real-world kitchen environment with the Franka Emika Panda robot. We adopt the DROID tabletop hardware setup [6], with some minor modifications (see Appendix VIII-C for details). We experiment with three real-world tasks and collect 50 human demonstrations for each task:

- `CounterToSinkPnP`: move an object from the counter to the sink basin. This task features nine object categories with diverse shapes: can, cup, coffee cup, water bottle, lemon, garlic, bowl, granola bar, and pear.
- `CounterToCabPnP`: move an object from the counter to the cabinet. This task features eight object categories.[1]
- `CloseDoor`: close the door for an overhead cabinet.

Humanoid Tabletop. A real-world tabletop environment with a Fourier GR-1 humanoid robot. We control the robot using a mink-based [64] IK controller. We use a first-person view RGB camera mounted to the head of the humanoid. We choose three tasks and collect 20 human demonstrations for each task. We describe more details in Appendix VIII-C.

- `CupPnP`: move a cup from the plate to the table.
- `MilkPnP`: move a box of milk from the table to the second level of the shelf.
- `Pouring`: pick up a cup with a ping-pong ball inside and pour the ball into a bowl on the table.

Our study is grounded in the real world—we compare the efficacy of different co-training methods by directly evaluating

[1] We use the same object categories as CounterToSinkPnP but exclude the water bottle due to difficulties in placing it stably into the cabinet.

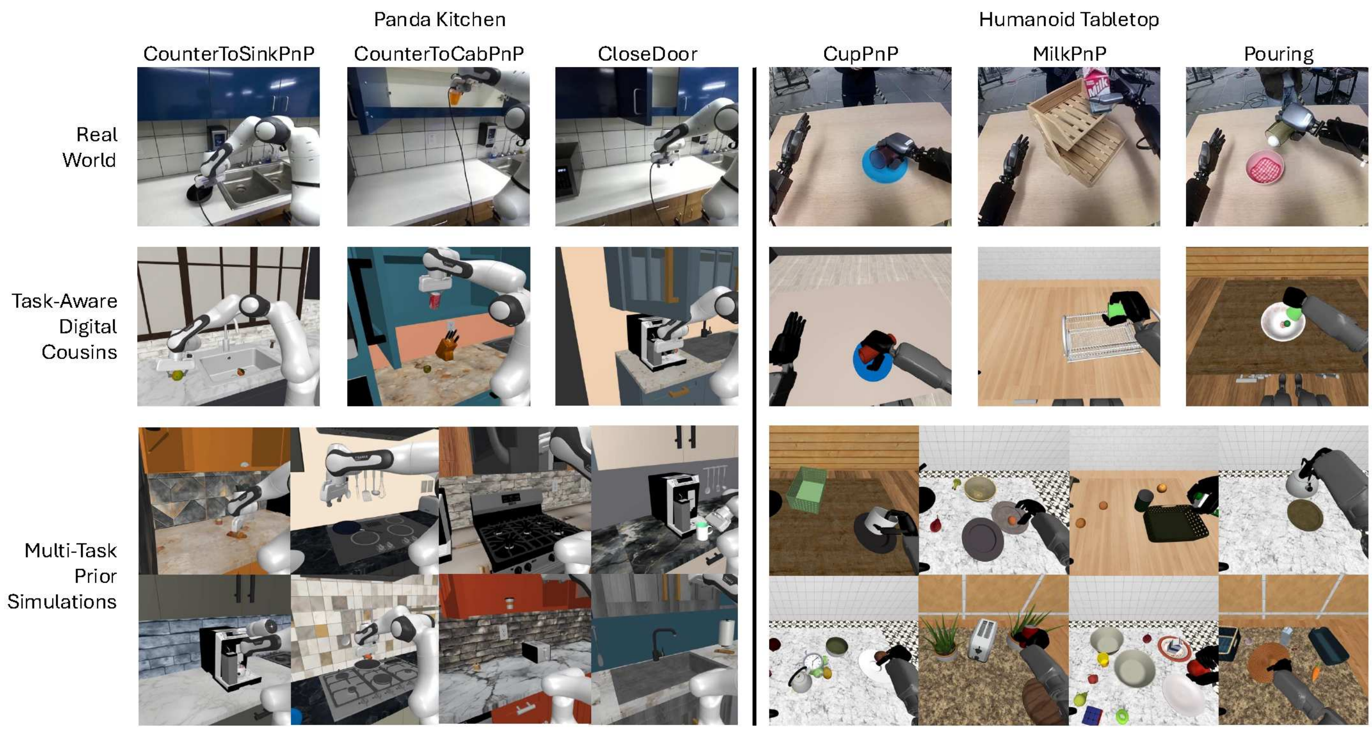

Fig. 3: **Real-World and Simulation Tasks.** We experiment with co-training on three data sources on the two robot domains of Kitchen Panda and Humanoid Tabletop: (top) data collected for real-world tasks; (middle) data from task-aware digital cousin environments that resemble the target tasks but are not perfectly aligned; (bottom) prior multi-task data from simulation that comprise a wide range of tasks and environments but have larger discrepancies with real-world tasks.

policies on these real-world tasks.

B. Prior Task-Agnostic Simulation Data

We leverage synthetically generated data to supplement real-robot datasets for policy training. One approach is to directly co-train with existing large-scale simulation datasets, or *prior task-agnostic simulation datasets*. We define a *prior task-agnostic simulation dataset* as any simulation dataset that existed before the creation of the downstream, real-world task. For the purpose of direct co-training, we use prior task-agnostic datasets that contain the same robot embodiment and action space, but this is not a strict requirement. We otherwise assume that these datasets cover a broad range of tasks and environments. We are interested in co-training with these datasets *out of the box*, without expending additional efforts designing new tasks in simulation and collecting new data. These datasets may have numerous discrepancies with real-world data, but they present a simple and convenient way to leverage simulation data. We use the following prior simulation datasets:

Panda Kitchen. We use the multi-task RoboCasa dataset [7]. We choose RoboCasa, given its focus on kitchen environments, a diverse range of scenes and tasks, and the availability of large robot data. In addition, Nasiriany et al. [7] showed preliminary findings that co-training with simulation data can aid transfer in a real-world kitchen environment. The dataset comprises 72k demonstrations across 24 tasks and 100 scenes; for each task, 3,000 demonstrations were generated from 50 source human demonstrations using the MimicGen data generation system [9]. Refer to Appendix VIII-D for in-depth details about the tasks and datasets. Note that three of these tasks semantically correspond to our real-world tasks but include notable discrepancies with the real-world setup, including initial robot joint positions, controller parameters, physical parameters, object categories, and the robot base position. We present a quantitative comparison of data composition for the real-world and prior simulation datasets in Appendix VIII-F.

Camera alignment differences between simulation and real-world data can be a major discrepancy. We address this discrepancy by re-rendering the simulation demonstrations to approximately match the camera poses of the real-world setup.[2] Note that this does not represent perfect alignment, but, as we demonstrate in our experiments, it can significantly help nonetheless. Beyond this simple post-processing operation, we do not make any further changes to the prior data.

Humanoid Tabletop. To mirror the setup for the kitchen domain, we create a prior task-agnostic dataset comprising 10 tasks in RoboCasa involving a single kitchen countertop and a GR-1 robot. Each task involves grasping a specified

[2]This operation introduces occlusions for the drawer and stove knob manipulation tasks, so we opt to exclude these. In total, we use 60k demonstrations across 20 tasks.

object from a source receptacle and placing it into a target receptacle (*e.g.*, from bowl to basket). Refer to Appendix VIII-D for additional details about these tasks. While semantically similar to the real-world setup, the prior tasks and datasets were developed independently and involve numerous discrepancies such as object categories, visual textures, distractor objects, and physical parameters. None of these 10 tasks is semantically equivalent to the real-world tasks—they involve different source and/or target receptacles. We use DexMimicGen [10], a data generation framework built on top of MimicGen for humanoid and other bimanual robots, to synthesize robot trajectories from dozens of human demonstrations. We generate 1,000 demonstrations for each task, resulting in 10k total demonstrations.

C. Building Task-Aware Simulation Datasets

The tasks and datasets presented in the previous section may have a number of large discrepancies with the real-world tasks, potentially limiting their utility. Alternatively, we can expend additional effort in creating custom tasks in simulation that are better aligned with the real-world tasks. Creating a perfect *digital twin* [56], [57], [65] copy of the real-world task is challenging, requiring extensive manual tuning, system identification, and sourcing identical 3D assets. Instead, we opt to create tasks in simulation that share the same task semantics, namely the object categories in the environment and the same behaviors. We refer to these as *task-aware digital cousins*. The term "digital cousin" was recently introduced by Dai et al. [26] to describe simulation environments that are close to, but not perfectly aligned with, their real-world counterparts. We extend this notion with a more precise definition: a *task-aware digital cousin* is a simulation dataset that preserves four key elements of the real-world task:

1) The same robot and action space;
2) The same task goal—specifically, the same success check and, if applicable, the same language instructions;
3) The same object categories, though individual instances may differ in geometry or texture;
4) The same environmental fixture categories (*e.g.*, kitchen counters, tabletops, cabinet doors).

We outline our efforts to create these tasks as follows:

Panda Kitchen. The real-world Panda Kitchen tasks are already represented in the RoboCasa prior dataset, but have several discrepancies as outlined in the prior section. We outline the changes that we made to the task and dataset as follows. First, we adjust the initial state distribution of the robot joints and robot base position in simulation to match the real environment. In addition, we restrict the objects in the task to a curated list of 10 object categories, which includes all of the nine object categories used in the real-world `CounterToSinkPnP` task. This is in contrast to the prior dataset, in which the authors feature a wider range of 66 possible object categories. For each task, we then collect 100 source human demonstrations. Finally, we generate 10,000 demonstrations for each task using MimicGen. This is in contrast to the prior dataset, where the authors collected 50 source human demonstrations per task and generated 3,000 demonstrations per task with MimicGen. We provide a more in-depth comparison between the real data, task-agnostic prior simulation data, and task-aware digital cousin simulation data in Appendix VIII-F.

Humanoid Tabletop. For each of the three tasks in this domain, we construct a digital cousin of the real-world environment in RoboCasa. In each of the real-world tasks, we use a fixed set of objects for both data collection and evaluation. In the digital cousin, however, we randomly select objects from the same category as those in the real-world task to increase the diversity of simulation demonstrations. Additionally, we align the robot's initial pose and camera position to closely replicate the real-world setup. We then collect 10 source demonstrations and generate 1,000 trajectories for each task using DexMimicGen [10]. A detailed analysis of the data composition is provided in Appendix VIII-F.

D. Training and Evaluation Protocol

In our study, we compare the effect of co-training with different forms of real-world and simulation data. For each task, we have access to the following forms of data:

- **Real-world data (`Real`)**: demonstrations collected for the target task in the real-world. See Section IV-A for the list of tasks.
- **Prior simulation data (`Prior`)**: task-agnostic simulation data outlined in Section IV-B.
- **Task-aware digital cousin data (`DC`)**: synthetic simulation data outlined in Section IV-C.

See Appendix Tables III and IV for an overview and comparison of these datasets for the Panda Kitchen and Humanoid Tabletop domains.

We compare various mixtures of these datasets by co-training a policy on the data and evaluating the resulting policy on our real-world tasks outlined in Section IV-A. We train visuomotor policies with the Diffusion Policy implementation from Chi et al. [38]. The policy takes RGB images and robot proprioceptive information as input and produces a sequence of actions to execute. We outline specific settings and hyperparameters in detail in Appendix VIII-G. Following training, we evaluate the policy across a number of trials and record the success rate. See Appendix VIII-H for details on our evaluation protocol.

V. EXPERIMENTS

In this section, we present a comprehensive empirical study of co-training real-world policies using simulation data. We begin by showcasing the benefits of co-training using our full-fledged recipe, which has been informed by systematic experimentation (Section V-A and Section V-B). Specifically, we demonstrate how co-training with simulation data enhances the real-world policy's in-domain performance (Section V-A) and improves its generalization to novel scenarios (Section V-B).

Data Composition	Panda Arm			GR-1 Humanoid			Average
	C2SPnP	C2CPnP	CloseDoor	CupPnP	MilkPnP	Pouring	
Real	44%	38%	10%	65%	50%	65%	45.3%
Real + DC	67%	**72%**	**100%**	**95%**	70%	85%	81.1%
Real + Prior	58%	53%	**100%**	80%	**80%**	70%	76.8%
Real + DC + Prior	**72%**	**72%**	**100%**	85%	**80%**	**90%**	**83.2%**

TABLE I: **Effect of different simulation data in the co-training mix.** We compare co-training with different simulation data on six tasks across two robot platforms. Note that we abbreviate the CounterToSinkPnP task as C2SPnP and CounterToCabPnP as C2CPnP. Co-training with Prior (third row) consistently boosts performance over policies trained on real data only (first row). On top of these results, adding better aligned DC further improves the overall performance (last row).

Next, we delve into the systematic experiments that guided the development of our recipe (Section V-D). These experiments identify key elements for effective sim-and-real co-training, providing insights into what factors matter most. Finally, we conclude with a concise and actionable recipe based on our findings (Section V-E), ensuring clarity for practitioners.

A. *Effectiveness of Sim-and-Real Co-Training*

Co-training with task-aware digital cousin data significantly enhances real-world performance beyond real-only policies. Table I presents our main results. In the second row, compared to policies trained only on Real, those trained on Real and DC exhibit a 35.8% higher average success rate. These results indicate that incorporating simulation data more closely aligned with real-world tasks significantly enhances real-world policy performance. It is important to note that DC data is generated in task-aware digital cousin simulation environments. These environments share the same task definitions, similar scene setups, and comparable camera views with the real world, though none are perfectly aligned. Nevertheless, with minimal effort, we were able to construct and approximately align these digital cousins with real-world environments (see Section IV-C), demonstrating the feasibility of leveraging such simulations for improved real-world policy training.

Co-training with task-agnostic prior simulation data also improves real-world performance. As shown in the third row of Table I, policies trained on Real and Prior consistently outperform those trained solely on Real across all tasks, achieving an average success rate improvement of 31.5%. This is a particularly surprising and encouraging result, as the Prior data are generated without any prior knowledge of real-world tasks. These results indicate that even without manual alignment of the simulation environment, co-training with simulation data yields substantial benefits. This finding highlights the potential of leveraging readily available simulation data to enhance real-world policy performance.

Finally, in the last row of Table I, policies trained on the combination of Real, DC, and Prior data in general perform the best, achieving an improvement of 37.9% over the real-only policies on average.

We observe a dramatic performance gap between Real and the other co-trained polices for the CloseDoor task. We further investigate the robustness of this gap by training the Real policy with more demos. See Section VIII-L for more results.

B. *Generalization Beyond Real Demonstrations*

To understand how simulation data enhances real-world policy performance, we investigate whether exposure to diverse situations in simulation—ones not explicitly covered in real-world demonstrations—can improve a policy's ability to generalize to similar, unseen situations in the real world. This question is particularly important because generating broad-coverage data in simulation is relatively easy, whereas collecting diverse real-world demonstrations is often expensive and impractical. If simulation data can effectively bridge this gap, it would provide a powerful and scalable way to improve real-world policies with minimal real-world data.

To explore this, we evaluate the generalization capabilities of co-trained policies beyond real-world demonstrations. Specifically, we consider two key axes of variation: novel objects and novel initial positions. We use the CounterToSinkPnP and CupPnP tasks as our testbed, where, by default, these factors are randomized in simulation to assess the policy's ability to handle novel scenarios in the real world (see Section IV-C).

Co-training with simulation data enhances policy robustness to novel object entities. For the CounterToSinkPnP task, we evaluate on eight new object categories (carrot, ladle, lime, apple, orange, sponge, cucumber, and banana) and new instances of the original object categories with differing size, color, and shape. For the CupPnP task, we replace the red cup with cups of different colors and introduce novel objects. The settings of generalization experiments are detailed in Appendix VIII-I. As shown in Table II, the policy trained solely on Real achieves a success rate of only 33% and 10% on novel objects, whereas the co-trained policy significantly outperforms it with success rates of 50% and 80%. The diversity in simulation data contributes to improved generalizability in real-world policy performance.

Co-training with simulation data enhances policy robustness to novel object positions. In this experiment, we exclude real demonstrations where the object is placed in the middle of the workspace, retaining only those where the object is positioned along borders or corners of the rectangular sampling

Data Composition	Unseen Objects		Unseen Positions	
	Panda	GR-1	Panda	GR-1
`Real`	33%	10%	11%	43%
`Real + DC`	50%	80%	28%	100%

TABLE II: **Co-training with sim enhances policy generalization across novel objects and novel positions.** We select the `CounterToSinkPnP` task on Panda and the `CupPnP` task on the humanoid and evaluate the policies' performance when the object is changed and when the object is placed at unseen positions.

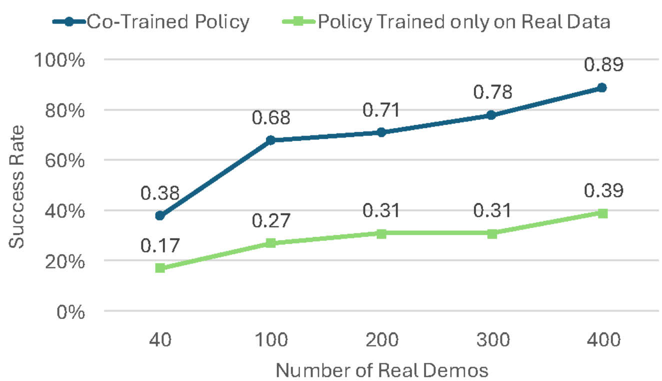

Fig. 4: **Effect of the quantity of real demonstrations.** We use a total of 4,000 simulation `DC` demos and vary the total number of real demos from 40 to 400 on task `MultiTaskPnP`. The results show that our co-training recipe remains beneficial with larger real datasets.

region. During evaluation, we place the objects in the center of the sampling region, which are unseen positions in the real demonstrations. The setups are visualized in Appendix VIII-I. In simulation, we still include data with objects distributed uniformly in the rectangular region. As shown in Table II, policies co-trained with `DC` achieve a twice higher success rate compared with the policies trained solely on `Real` for both humanoid and Panda experiments. This result indicates that diverse simulation data substantially improve policy robustness to spatial variations.

C. Effectiveness of Co-Training in Data-Rich Settings

Our main results (Section V-A) examine the effectiveness of sim-and-real co-training in single-task settings with different embodiment and scene setups. Meanwhile, recent works [1], [2], [66] have demonstrated incredible performance by training policies on large-scale, multi-task real-world manipulation datasets. This raises the question—can co-training with synthetic data still be beneficial with larger real-robot datasets? To evaluate the effectiveness of our co-training approach in a scaled-up setting, we conduct experiments on a new humanoid `MultiTaskPnP` task. In this task, the robot must pick an object from one container and place it into another, with four different source-target container combinations.

For each real-world task variation, we construct a corresponding task-aware digital cousin. We train policies using a fixed set of 4,000 `DC` demonstrations (1,000 per task) while varying the number of real-world demonstrations. During

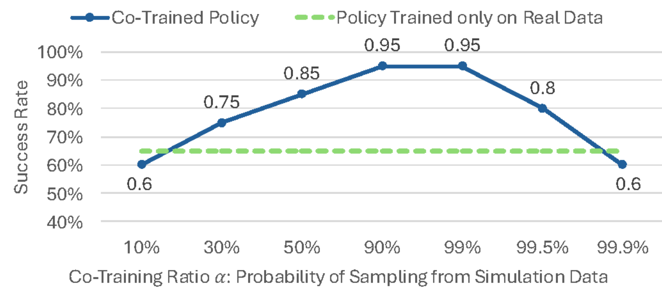

Fig. 5: **Effect of the different co-training ratios.** The co-training ratio, α, is the probability of sampling from simulation data in each minibatch. We experiment on the `CupPnP` task with 20 real demos and 1000 simulation demos from task-aware digital cousins. Tuning the co-training ratio is important for the good performance of co-trained policies.

testing, we evaluate the generalizability of the policy using three unseen objects. The detailed task setup is provided in Appendix VIII-J.

As shown in Figure 4, increasing the number of real-world demonstrations improves the success rate for both real-only and co-trained policies. **Even with 400 real demonstrations, the co-trained policy consistently outperforms the real-only policy, demonstrating that sim-and-real co-training remains beneficial even in data-rich settings.**

D. Key Elements of Effective Co-Training

In the sections above, we demonstrated the effectiveness of sim-and-real co-training with our full-fledged recipe. In this section, we present systematic studies that help identify key elements for successful co-training. These elements include the quantity of real and simulation demonstrations, the co-training ratio, and the importance of camera alignment in digital cousin (`DC`) environments. Our findings highlight practical strategies for optimizing co-training and improving real-world policy performance.

A sufficient number of simulation demonstrations is crucial for effective co-training. First, we analyze the impact of varying the number of simulation demonstrations. We reduce the number of `DC` demonstrations for the Panda `CounterToSinkPnP` and GR-1 `CupPnP` tasks and train policies using `Real` data combined with the reduced `DC` data. In the Panda `CounterToSinkPnP` task, decreasing the number of simulation demonstrations from 10k to 500 causes the success rate to drop from 67% to 53%. Similarly, in the GR-1 `CupPnP` task, reducing the number of simulation demonstrations from 1k to 100 lowers the success rate from 95% to 75%. Therefore, having sufficient simulation demonstrations is essential for achieving strong performance in co-trained policies.

Tuning the co-training ratio is required for effective co-training. We investigate the impact of the co-training ratio—the proportion of simulation data used during training—on the `CupPnP` task (Section IV-A). As shown in Figure 5, a 1:1 ratio (50%) is suboptimal. In our experiments,

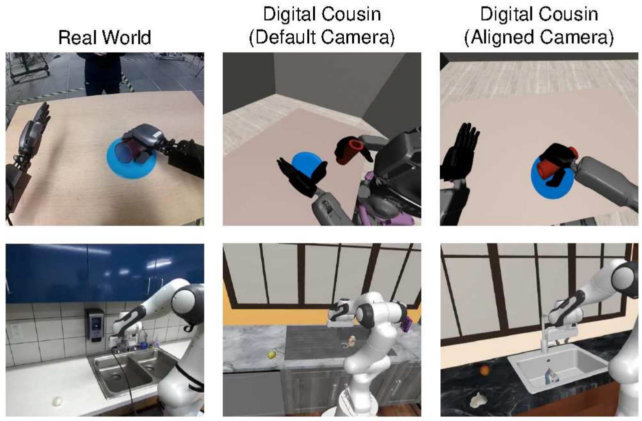

Fig. 6: **Camera alignment visualization.** We visualize the default and aligned camera views of the GR-1 `CupPnP` task and Panda `CounterToSinkPnP` task.

a co-training ratio of 99% yielded the best performance. However, pushing the co-training ratio further to values like 99.5% and 99.9% resulted in drops in success rate, from 95% to 60%. These findings highlight the importance of carefully tuning the co-training ratio to achieve optimal performance.

Camera alignment is critical for successful co-training with task-aware digital cousin data. In `DC`, we approximate the alignment of the simulation environment's camera with the real-world camera view. To evaluate the importance of this alignment, we render data using the default, unaligned camera view and trained policies on the resulting misaligned simulation data. The results indicate a significant drop in performance compared to policies co-trained with properly aligned `DC` data. On the Panda arm `CounterToSinkPnP` task, the co-training success rate dropped from 67% to 56%, while in the GR-1 humanoid `CupPnP` task, it declined from 95% to 70%. We visualize the default and aligned camera views in Figure 6. Notably, the aligned camera is not strictly identical to the real-world camera. For example, the camera mounted on the real-world humanoid has a fisheye effect, whereas our aligned simulation camera does not model this distortion. This suggests that while alignment enhances performance, perfect camera alignment is not necessary for effective co-training.

E. A Simple Recipe for Sim-and-Real Co-Training

Based on our empirical findings, we provide a set of recommendations to help practitioners reap the benefits of co-training with synthetic simulation data.

- **Task and scene composition.** The greatest performance gains are observed when co-training with simulation data from task-aware digital cousins, where the task and scene compositions closely mirror those of the real-world setting. Nevertheless, co-training with large multi-task prior simulation data—despite differences in task and scene composition—still provides meaningful benefits.
- **Object composition and initialization distribution.** Incorporating diverse objects and varying their placements in simulation data helps real-world policies generalize to unseen scenarios.
- **Alignment between the task-aware digital cousin and the real world.** It is essential that the simulation task shares the same definition and success criteria as its real-world counterpart. Additionally, maintaining similar camera viewpoints between simulation and real-world settings can improve performance, though perfect alignment is not required.
- **Co-training hyperparameters.** We recommend utilizing a sufficiently large amount of simulation data (ideally, orders of magnitude more than real-world data) and carefully tuning the co-training ratio to optimize performance.

VI. Limitations

Although we conduct systematic studies on several tasks across both a tabletop manipulator and a humanoid robot, most tasks are centered around pick-and-place. Extending our approach to a broader set of manipulation tasks, such as high-precision insertion, and longer-horizon tasks, is left for future work. While our co-training recipe consistently improves success rates compared to solely training on real-robot data collection, the policy's performance is still not perfect. Future efforts could look into building on top of this recipe to further boost real-world performance. Finally, certain real-world tasks—particularly those involving deformable objects and liquids—remain difficult to simulate accurately, inherently limiting the applicability of simulation data. Applying this co-training strategy to such tasks presents a challenge. Future work could explore the use of co-training data produced by video generation models and world models [67]–[69] as a way to bridge this gap.

VII. Conclusion

In this work, we systematically investigate how to effectively leverage synthetically generated data from physics simulations to solve real-world, vision-based manipulation tasks. By analyzing key factors that impact the dataset distributions and co-training strategies, we demonstrate that large-scale simulation data can effectively complement real-world data—even in the presence of significant discrepancies—leading to policies that outperform those trained on real-world data alone.

Furthermore, we find that simulation data enhances policy generalization to scenarios not covered in real-world datasets, underscoring its potential for developing more robust and adaptable robotic systems. Our findings highlight the promise of leveraging diverse simulation data to advance generalist robot autonomy.

In addition, we offer a set of practical recommendations for practitioners to harness the benefits of synthetic simulation data without requiring extensive manual effort in constructing or aligning simulation environments. They reinforce the importance of systematically integrating simulation and real-world data. We hope our insights will inspire future research to further unleash the potential of simulation in building generalizable robot models in the real world.

REFERENCES

[1] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter *et al.*, "π_0: A vision-language-action flow model for general robot control," *arXiv preprint arXiv:2410.24164*, 2024.

[2] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi *et al.*, "OpenVLA: An open-source vision-language-action model," *arXiv preprint arXiv:2406.09246*, 2024.

[3] NVIDIA, J. Bjorck, F. Castañeda, N. Cherniadev, X. Da, R. Ding, L. Fan, Y. Fang, D. Fox, F. Hu, S. Huang *et al.*, "Gr00t n1: An open foundation model for generalist humanoid robots," *arXiv preprint arXiv:2503.14734*, 2025.

[4] 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 *Proceedings of Robotics: Science and Systems*, New York City, NY, USA, 6 2022.

[5] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu *et al.*, "RT-1: Robotics transformer for real-world control at scale," *arXiv preprint arXiv:2212.06817*, 2022.

[6] A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M. K. Srirama, L. Y. Chen *et al.*, "Droid: A large-scale in-the-wild robot manipulation dataset," in *Proceedings of Robotics: Science and Systems*, 2024.

[7] S. Nasiriany, A. Maddukuri, L. Zhang, A. Parikh, A. Lo, A. Joshi, A. Mandlekar, and Y. Zhu, "Robocasa: Large-scale simulation of everyday tasks for generalist robots," in *Robotics: Science and Systems (RSS)*, 2024.

[8] Y. Wang, Z. Xian, F. Chen, T.-H. Wang, Y. Wang, K. Fragkiadaki, Z. Erickson, D. Held, and C. Gan, "Robogen: Towards unleashing infinite data for automated robot learning via generative simulation," in *Forty-first International Conference on Machine Learning*, 2023.

[9] A. Mandlekar, S. Nasiriany, B. Wen, I. Akinola, Y. Narang, L. Fan, Y. Zhu, and D. Fox, "Mimicgen: A data generation system for scalable robot learning using human demonstrations," in *Conference on Robot Learning*. PMLR, 2023, pp. 1820–1864.

[10] Z. Jiang, Y. Xie, K. Lin, Z. Xu, W. Wan, A. Mandlekar, L. Fan, and Y. Zhu, "Dexmimicgen: Automated data generation for bimanual dexterous manipulation via imitation learning," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*, 2025.

[11] C. Garrett, A. Mandlekar, B. Wen, and D. Fox, "Skillmimicgen: Automated demonstration generation for efficient skill learning and deployment," *arXiv preprint arXiv:2410.18907*, 2024.

[12] M. Dalal, A. Mandlekar, C. R. Garrett, A. Handa, R. Salakhutdinov, and D. Fox, "Imitating task and motion planning with visuomotor transformers," in *Conference on Robot Learning*. PMLR, 2023, pp. 2565–2593.

[13] F. Ramos, R. C. Possas, and D. Fox, "Bayessim: adaptive domain randomization via probabilistic inference for robotics simulators," *arXiv preprint arXiv:1906.01728*, 2019.

[14] F. Muratore, T. Gruner, F. Wiese, B. Belousov, M. Gienger, and J. Peters, "Neural posterior domain randomization," in *Conference on Robot Learning*. PMLR, 2022, pp. 1532–1542.

[15] V. Lim, H. Huang, L. Y. Chen, J. Wang, J. Ichnowski, D. Seita, M. Laskey, and K. Goldberg, "Real2sim2real: Self-supervised learning of physical single-step dynamic actions for planar robot casting," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 8282–8289.

[16] M. Memmel, A. Wagenmaker, C. Zhu, P. Yin, D. Fox, and A. Gupta, "Asid: Active exploration for system identification in robotic manipulation," *arXiv preprint arXiv:2404.12308*, 2024.

[17] A. Z. Ren, H. Dai, B. Burchfiel, and A. Majumdar, "Adaptsim: Task-driven simulation adaptation for sim-to-real transfer," *arXiv preprint arXiv:2302.04903*, 2023.

[18] 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," in *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2017, pp. 23–30.

[19] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *2018 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2018, pp. 3803–3810.

[20] Y. Zhu, Z. Wang, J. Merel, A. Rusu, T. Erez, S. Cabi, S. Tunyasuvunakool, J. KramÃ¡r, R. Hadsell, N. de Freitas, and N. Heess, "Reinforcement and imitation learning for diverse visuomotor skills," in *Proceedings of Robotics: Science and Systems*, 2018.

[21] O. M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray *et al.*, "Learning dexterous in-hand manipulation," *The International Journal of Robotics Research*, vol. 39, no. 1, pp. 3–20, 2020.

[22] A. Handa, A. Allshire, V. Makoviychuk, A. Petrenko, R. Singh, J. Liu, D. Makoviichuk, K. Van Wyk, A. Zhurkevich, B. Sundaralingam *et al.*, "Dextreme: Transfer of agile in-hand manipulation from simulation to reality," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2023, pp. 5977–5984.

[23] K. Bousmalis, A. Irpan, P. Wohlhart, Y. Bai, M. Kelcey, M. Kalakrishnan, L. Downs, J. Ibarz, P. Pastor, K. Konolige *et al.*, "Using simulation and domain adaptation to improve efficiency of deep robotic grasping," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 4243–4250.

[24] L. Ankile, A. Simeonov, I. Shenfeld, M. Torne, and P. Agrawal, "From imitation to refinement–residual rl for precise assembly," *arXiv preprint arXiv:2407.16677*, 2024.

[25] J. Wang, Y. Qin, K. Kuang, Y. Korkmaz, A. Gurumoorthy, H. Su, and X. Wang, "Cyberdemo: Augmenting simulated human demonstration for real-world dexterous manipulation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 17 952–17 963.

[26] T. Dai, J. Wong, Y. Jiang, C. Wang, C. Gokmen, R. Zhang, J. Wu, and L. Fei-Fei, "Automated creation of digital cousins for robust policy learning," *arXiv preprint arXiv:2410.07408*, 2024.

[27] D. A. Pomerleau, "Alvinn: An autonomous land vehicle in a neural network," in *Advances in neural information processing systems*, 1989, pp. 305–313.

[28] M. Drolet, S. Stepputtis, S. Kailas, A. Jain, J. Peters, S. Schaal, and H. Ben Amor, "A comparison of imitation learning algorithms for bimanual manipulation," *IEEE Robotics and Automation Letters (RA-L)*, 2024.

[29] A. Tung, J. Wong, A. Mandlekar, R. Martín-Martín, Y. Zhu, L. Fei-Fei, and S. Savarese, "Learning multi-arm manipulation through collaborative teleoperation," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 9212–9219.

[30] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, "Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware," in *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, 7 2023.

[31] J. Aldaco, T. Armstrong, R. Baruch, J. Bingham, S. Chan, K. Draper, D. Dwibedi, C. Finn, P. Florence, S. Goodrich *et al.*, "Aloha 2: An enhanced low-cost hardware for bimanual teleoperation," *arXiv preprint arXiv:2405.02292*, 2024.

[32] S. Schaal, "Is imitation learning the route to humanoid robots?" *Trends in cognitive sciences*, vol. 3, no. 6, pp. 233–242, 1999.

[33] A. J. Ijspeert, J. Nakanishi, and S. Schaal, "Movement imitation with nonlinear dynamical systems in humanoid robots," *Proceedings 2002 IEEE International Conference on Robotics and Automation*, vol. 2, pp. 1398–1403 vol.2, 2002.

[34] M. Seo, S. Han, K. Sim, S. H. Bang, C. Gonzalez, L. Sentis, and Y. Zhu, "Deep imitation learning for humanoid loco-manipulation through human teleoperation," in *2023 IEEE-RAS 22nd International Conference on Humanoid Robots (Humanoids)*. IEEE, 2023, pp. 1–8.

[35] R. Ding, Y. Qin, J. Zhu, C. Jia, S. Yang, R. Yang, X. Qi, and X. Wang, "Bunny-visionpro: Real-time bimanual dexterous teleoperation for imitation learning," *arXiv preprint arXiv:2407.03162*, 2024.

[36] X. Cheng, J. Li, S. Yang, G. Yang, and X. Wang, "Open-television: teleoperation with immersive active visual feedback," *arXiv preprint arXiv:2407.01512*, 2024.

[37] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín, "What matters in learning from offline human demonstrations for robot manipulation," in *Conference on Robot Learning (CoRL)*, 2021.

[38] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," in *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[39] C. Finn, T. Yu, T. Zhang, P. Abbeel, and S. Levine, "One-shot visual imitation learning via meta-learning," in *Conference on robot learning*. PMLR, 2017, pp. 357–368.

[40] A. Billard, S. Calinon, R. Dillmann, and S. Schaal, "Robot programming by demonstration," in *Springer Handbook of Robotics*, 2008.

[41] S. Calinon, F. D'halluin, E. L. Sauser, D. G. Caldwell, and A. Billard, "Learning and reproduction of gestures by imitation," *IEEE Robotics and Automation Magazine*, vol. 17, pp. 44–54, 2010.

[42] A. Mandlekar, D. Xu, R. Martín-Martín, S. Savarese, and L. Fei-Fei, "GTI: Learning to Generalize across Long-Horizon Tasks from Human Demonstrations," in *Proceedings of Robotics: Science and Systems*, Corvalis, Oregon, USA, 7 2020.

[43] A. Zeng, P. Florence, J. Tompson, S. Welker, J. Chien, M. Attarian, T. Armstrong, I. Krasin, D. Duong, V. Sindhwani *et al.*, "Transporter networks: Rearranging the visual world for robotic manipulation," in *Conference on Robot Learning*. PMLR, 2021, pp. 726–747.

[44] C. Wang, R. Wang, A. Mandlekar, L. Fei-Fei, S. Savarese, and D. Xu, "Generalization through hand-eye coordination: An action space for learning spatially-invariant visuomotor control," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 8913–8920.

[45] C. Lynch, M. Khansari, T. Xiao, V. Kumar, J. Tompson, S. Levine, and P. Sermanet, "Learning latent plans from play," in *Conference on Robot Learning*, 2019.

[46] K. Pertsch, Y. Lee, Y. Wu, and J. J. Lim, "Demonstration-guided reinforcement learning with learned skills," in *Conference on Robot Learning*, 2021.

[47] A. Ajay, A. Kumar, P. Agrawal, S. Levine, and O. Nachum, "Opal: Offline primitive discovery for accelerating offline reinforcement learning," in *International Conference on Learning Representations*, 2021.

[48] K. Hakhamaneshi, R. Zhao, A. Zhan, P. Abbeel, and M. Laskin, "Hierarchical few-shot imitation with skill transition models," in *International Conference on Learning Representations*, 2021.

[49] Y. Zhu, P. Stone, and Y. Zhu, "Bottom-up skill discovery from unsegmented demonstrations for long-horizon robot manipulation," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4126–4133, 2022.

[50] S. Nasiriany, T. Gao, A. Mandlekar, and Y. Zhu, "Learning and retrieval from prior data for skill-based imitation learning," in *Conference on Robot Learning (CoRL)*, 2022.

[51] 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)*. IEEE, 2019, pp. 8973–8979.

[52] B. Mehta, M. Diaz, F. Golemo, C. J. Pal, and L. Paull, "Active domain randomization," in *Conference on Robot Learning*. PMLR, 2020, pp. 1162–1176.

[53] P. Huang, X. Zhang, Z. Cao, S. Liu, M. Xu, W. Ding, J. Francis, B. Chen, and D. Zhao, "What went wrong? closing the sim-to-real gap via differentiable causal discovery," in *Conference on Robot Learning*. PMLR, 2023, pp. 734–760.

[54] A. Kumar, Z. Fu, D. Pathak, and J. Malik, "Rma: Rapid motor adaptation for legged robots," *arXiv preprint arXiv:2107.04034*, 2021.

[55] B. Evans, A. Thankaraj, and L. Pinto, "Context is everything: Implicit identification for dynamics adaptation," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 2642–2648.

[56] Z. Jiang, C.-C. Hsu, and Y. Zhu, "Ditto: Building digital twins of articulated objects from interaction," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5616–5626.

[57] M. Torne, A. Simeonov, Z. Li, A. Chan, T. Chen, A. Gupta, and P. Agrawal, "Reconciling reality through simulation: A real-to-sim-to-real approach for robust manipulation," *arXiv preprint arXiv:2403.03949*, 2024.

[58] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn *et al.*, "Rt-2: Vision-language-action models transfer web knowledge to robotic control," *arXiv preprint arXiv:2307.15818*, 2023.

[59] J. Hejna, C. Bhateja, Y. Jian, K. Pertsch, and D. Sadigh, "Re-mix: Optimizing data mixtures for large scale imitation learning," *arXiv preprint arXiv:2408.14037*, 2024.

[60] A. Xie, L. Lee, T. Xiao, and C. Finn, "Decomposing the generalization gap in imitation learning for visual robotic manipulation," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 3153–3160.

[61] W. Pumacay, I. Singh, J. Duan, R. Krishna, J. Thomason, and D. Fox, "The colosseum: A benchmark for evaluating generalization for robotic manipulation," *arXiv preprint arXiv:2402.08191*, 2024.

[62] J. Gao, A. Xie, T. Xiao, C. Finn, and D. Sadigh, "Efficient data collection for robotic manipulation via compositional generalization," *arXiv preprint arXiv:2403.05110*, 2024.

[63] V. Saxena, M. Bronars, N. R. Arachchige, K. Wang, W. C. Shin, S. Nasiriany, A. Mandlekar, and D. Xu, "What matters in learning from large-scale datasets for robot manipulation," in *CoRL 2024 Workshop on Mastering Robot Manipulation in a World of Abundant Data*.

[64] K. Zakka, "Mink: Python inverse kinematics based on MuJoCo," Jul. 2024. [Online]. Available: https://github.com/kevinzakka/mink

[65] S. Nasiriany, H. Liu, and Y. Zhu, "Augmenting reinforcement learning with behavior primitives for diverse manipulation tasks," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2022.

[66] Open X-Embodiment Collaboration, A. O'Neill, A. Rehman, A. Maddukuri, A. Gupta, A. Padalkar, A. Lee, A. Pooley, A. Gupta, A. Mandlekar, A. Jain *et al.*, "Open X-Embodiment: Robotic learning datasets and RT-X models," https://arxiv.org/abs/2310.08864, 2023.

[67] NVIDIA, N. Agarwal, A. Ali, M. Bala, Y. Balaji, E. Barker, T. Cai, P. Chattopadhyay, Y. Chen, Y. Cui, Y. Ding, D. Dworakowski, J. Fan *et al.*, "Cosmos world foundation model platform for physical AI," 2025.

[68] J. Bruce, M. Dennis, A. Edwards, J. Parker-Holder, Y. Shi, E. Hughes, M. Lai, A. Mavalankar, R. Steigerwald, C. Apps, Y. Aytar, S. Bechtle, F. Behbahani, S. Chan, N. Heess, L. Gonzalez, S. Osindero, S. Ozair, S. Reed, J. Zhang, K. Zolna, J. Clune, N. de Freitas, S. Singh, and T. Rocktäschel, "Genie: Generative interactive environments," 2024. [Online]. Available: https://arxiv.org/abs/2402.15391

[69] A. Hu, L. Russell, H. Yeo, Z. Murez, G. Fedoseev, A. Kendall, J. Shotton, and G. Corrado, "Gaia-1: A generative world model for autonomous driving," 2023. [Online]. Available: https://arxiv.org/abs/2309.17080

[70] Y. Zhu, A. Joshi, P. Stone, and Y. Zhu, "Viola: Imitation learning for vision-based manipulation with object proposal priors," *6th Annual Conference on Robot Learning*, 2022.

[71] O. Khatib, "Inertial properties in robotic manipulation: An object-level framework," *IJRR*, 1995.

[72] E. Perez, F. Strub, H. de Vries, V. Dumoulin, and A. Courville, "Film: Visual reasoning with a general conditioning layer," 2017. [Online]. Available: https://arxiv.org/abs/1709.07871

[73] C. Chi, Z. Xu, C. Pan, E. Cousineau, B. Burchfiel, S. Feng, R. Tedrake, and S. Song, "Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots," in *Proceedings of Robotics: Science and Systems (RSS)*, 2024.

[74] A. Dosovitskiy, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[75] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention*, 2015, pp. 234–241.

[76] K. Rao, C. Harris, A. Irpan, S. Levine, J. Ibarz, and M. Khansari, "Rl-cyclegan: Reinforcement learning aware simulation-to-real. in 2020 ieee," in *CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020, pp. 11 154–11 163.

[77] D. Ho, K. Rao, Z. Xu, E. Jang, M. Khansari, and Y. Bai, "Retinagan: An object-aware approach to sim-to-real transfer," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 10 920–10 926.

[78] P. M. Scheikl, E. Tagliabue, B. Gyenes, M. Wagner, D. Dall'Alba, P. Fiorini, and F. Mathis-Ullrich, "Sim-to-real transfer for visual reinforcement learning of deformable object manipulation for robot-assisted surgery," *IEEE Robotics and Automation Letters*, vol. 8, no. 2, pp. 560–567, 2022.

[79] D. Liu, Y. Chen, and Z. Wu, "Digital twin (dt)-cyclegan: Enabling zero-shot sim-to-real transfer of visual grasping models," *IEEE Robotics and Automation Letters*, vol. 8, no. 5, pp. 2421–2428, 2023.

[80] Z. Yang, J. Teng, W. Zheng, M. Ding, S. Huang, J. Xu, Y. Yang, W. Hong, X. Zhang, G. Feng *et al.*, "Cogvideox: Text-to-video diffusion models with an expert transformer," *arXiv preprint arXiv:2408.06072*, 2024.

VIII. Appendix

A. Overview

The Appendix contains the following content:

- **Author Contributions** (Appendix VIII-B): list of each author's contributions to the paper
- **Real-World Tasks** (Appendix VIII-C): details about real-world domains and tasks
- **Task-Agnostic Simulation Datasets** (Appendix VIII-D): details about task-agnostic simulation environments and datasets
- **Task-Aware Digital Cousin Datasets** (Appendix VIII-E): details about the tuning process for digital cousin environments and generating large-scale data
- **Data Composition Analysis** (Appendix VIII-F): comparing data composition factors across real, digital cousin, and prior datasets
- **Training Details** (Appendix VIII-G): training algorithm, model architecture, and training protocols
- **Policy Evaluation** (Appendix VIII-H): experiment evaluation protocols
- **Generalization Experiment Details** (Appendix VIII-I): details about the generalization experiments presented in Section V-B
- **MultiTaskPnP Task Setup** (Appendix VIII-J): details about the multi-task experiment setup presented in Section V-C
- **Improving Visual Realism with Vid2Vid** (Appendix VIII-K): additional experiments on bridging the visual gap between simulation and real-world data with Vid2Vid techniques
- **Training CloseDoor with More Demos** (Appendix VIII-L): additional experiments to improve performance on real-world CloseDoor task
- **FAQ** (Appendix VIII-M): additional topics

B. Author Contributions

- **Algorithm design:** Soroush Nasiriany, Zhenyu Jiang, Lawrence Yunliang Chen, Abhiram Maddukuri, Ajay Mandlekar, Yuke Zhu
- **Experiments (Panda Kitchen):** Abhiram Maddukuri, Soroush Nasiriany
- **Experiments (Humanoid Tabletop):** Zhenyu Jiang, Lawrence Yunliang Chen, Yuqi Xie, Zu Wang
- **Infrastructure support:** Yu Fang, Wenqi Huang, Nikita Chernyadev, Zhenjia Xu, Soroush Nasiriany
- **Digital release:** Lawrence Yunliang Chen, Zhenyu Jiang
- **Technical advice:** Ken Goldberg, Scott Reed
- **Project leads:** Yuke Zhu, Linxi Fan, Ajay Mandlekar

C. Real-World Tasks

Panda Kitchen. For our Panda Kitchen tasks, we use the DROID [6] setup and make some modifications. We use the controller from Deoxys [70], which supports OSC-based end effector control [71], and opt to keep the original parallel-jaw Panda gripper instead of the Robotiq gripper. We use two side-view third-person cameras and an eye-in-hand camera.

For each task, we collect data via teleoperation using a SpaceMouse. The robot starts from a fixed initial position and joint configuration, with camera poses remaining constant across each demo. For the `CounterToSinkPnP` task, we uniformly sample object placements from the rectangular counter region to the left of the sink. For the `CounterToCabPnP` task, we uniformly sample object placements from the rectangular counter region below the cabinet. Finally, in the `CloseDoor` task, we uniformly sample the initial angle of the open door from the interval [85°, 115°]. Refer to Figure 7 for a visualization of the start states and sampling regions for each task.

Humanoid Tabletop. We use a Fourier GR-1 robot with the default 6-DoF dexterous hands. The joints of the robot's lower body and waist are locked, while the two arms and hands are activated. During data collection, a human operator wears a pair of MANUS gloves with a VIVE tracker positioned on the back of each hand to capture the finger pose and wrist pose, respectively. We implement an inverse kinematic (IK) controller based on the mink framework [64] to control the robot during teleoperation, where the human input consists of wrist pose and finger joint commands. During policy learning and deployment, we use the joint positions computed by the IK solver as the action space. Additionally, an OAK-D camera mounted on the robot's head provides egocentric visual input for the system. Only one of the stereo RGB images (and not the depth) is used for the policy, and no additional third-person view is used.

For each of the three single tasks—`CupPnP`, `MilkPnP`, and `Pouring`, we collect 20 demonstrations in the real world. The initial pose of the robot varies around a standard reset pose according to the human teleoperator's pose. Only one set of objects is used for each task, but the object positions are randomized uniformly in the region as visualized in Figure 7.

For the multi-task setting (`Cuttingboard2Basket`, `Cuttingboard2Pan`, `Mat2Basket`, and `Plate2Bowl`), we additionally select multiple object instances during data collection. See Appendix VIII-J for details.

D. Task-Agnostic Simulation Datasets

Panda Kitchen. We use a subset of the 72k trajectories from the RoboCasa dataset [7]. Specifically, we exclude the `OpenDrawer`, `CloseDrawer`, `TurnOnStove`, and `TurnOffStove` tasks from the original dataset due to lack of visibility introduced by camera pose alignment. This results in 60k trajectories over the following 20 tasks. We cite the task descriptions from RoboCasa here:

1) `PickPlaceCounterToCabinet`: Pick an object from the counter and place it inside the cabinet.
2) `PickPlaceCabinetToCounter`: Pick an object from the cabinet and place it on the counter.

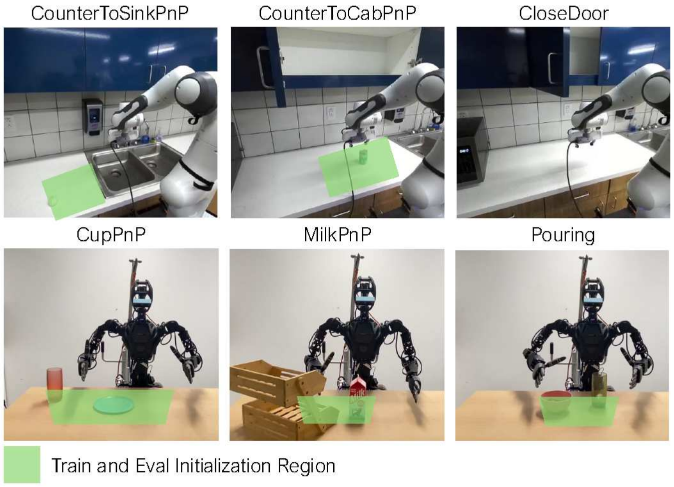

Fig. 7: **Visualization of start states for our experiments.** We visualize the starting states of the Panda and GR-1 for our experiments, including initialization regions for our pick-and-place tasks.

3) `PickPlaceCounterToSink`: Pick an object from the counter and place it in the sink.
4) `PickPlaceSinkToCounter`: Pick an object from the sink and place it on the counter area next to the sink.
5) `PickPlaceCounterToMicrowave`: Pick an object from the counter and place it inside the microwave.
6) `PickPlaceMicrowaveToCounter`: Pick an object from inside the microwave and place it on the counter.
7) `PickPlaceCounterToStove`: Pick an object from the counter and place it in a pan or pot on the stove
8) `PickPlaceStoveToCounter`: Pick an object from the stove (via a pot or pan) and place it on the counter.
9) `OpenSingleDoor`: Open a microwave door or a cabinet with a single door.
10) `OpenDoubleDoor`: Open a cabinet with two opposite-facing doors.
11) `CloseSingleDoor`: Close a microwave door or a cabinet with a single door.
12) `CloseDoubleDoor`: Close a cabinet with two opposite-facing doors.
13) `TurnOnSinkFacuet`: Turn on the sink faucet to begin the flow of water.
14) `TurnOffSinkFaucet`: Turn off the sink faucet to stop the flow of water.
15) `TurnSinkSpout`: Turn the sink spout.
16) `CoffeePressButton`: Press the button on the coffee machine to pour coffee into the mug.
17) `TurnOnMicrowave`: Turn on the microwave by pressing the start button.
18) `TurnOffMicrowave`: Turn off the microwave by pressing the stop button.
19) `CoffeeSetupMug`: Pick the mug from the counter and insert it into the coffee machine mug holder area.
20) `CoffeeServeMug`: Remove the mug from the coffee machine mug holder and place it on the counter.

Humanoid Tabletop. The tabletop humanoid datasets comprise 10 tasks in the RoboCasa simulation framework:

1) `CounterToPlate`: Pick up an object from the tabletop and place it on the nearby plate.
2) `PnPAppleToPlate`: Pick up the apple from the tabletop and place it on the nearby plate.
3) `PnPCanToBowl`: Pick up the can from the tabletop and place it in the nearby bowl.
4) `PnPMugToPlate`: Pick up the mug from the tabletop and place it on the nearby plate.
5) `PnPFruitPlacement`: Pick up the fruit from the tabletop and place it on the nearby plate.
6) `PnPKettleToPlate`: Pick up the kettle from the tabletop and place it on the nearby plate.
7) `PnPMilkPlateToPlate`: Pick up the milk carton located on a plate and transfer it to another plate.
8) `PnPMilkToBasket`: Pick up the milk carton from the tabletop and place it in the nearby basket.
9) `PnPPlateToPlate`: Pick the object located on a plate and transfer it to another plate.
10) `PnPVegetableBowlToPlate`: Pick the vegetable located inside a bowl and transfer it to a plate.

E. Task-Aware Digital Cousin Datasets

Panda Kitchen. We design a *digital cousin* for each of the three tasks in the Panda Kitchen. Among the set of data composition factors outlined in Section V-D, we focus on approximately aligning the following factors: Task composition, Object composition, and Initialization distribution. See Tables III and V for a summary of the alignments.

- **Task composition:** We align our digital cousin to contain the same task semantics and motions as in the real world. This involves using identical language instructions to those in the real task and approximately matching the initial joint configurations and positions to ensure similar trajectory distributions.
- **Object composition:** We design the digital cousin for our `Pick-and-Place` tasks to predominantly include object categories present in the real-world setting. Specifically, in the `CounterToSinkPnP DC`, 9/10 object categories overlap with those used in real-world tasks. Similarly, in the `CounterToCabPnP DC`, 8/10 object categories are shared with the real-world setting.
- **Initialization distribution:** We approximately align starting joint states, initial positions, and object initialization regions for our digital cousins. Specifically, we update the default starting position of the Panda to be farther away from the counterspace and increase the starting height to better match the starting position in the real world. We also reduce the area of the default sampling region to better match the sampling region in the real-world setting.

For our `CounterToSinkPnP DC`, `CounterToCabPnP DC`, and `CloseDoor DC`, we generate 10,000 synthetic trajectories each from an initial source of 100 human

	Dataset	Obj. Cat.	Obj. Ins.	Scenes	Demos
Real	CounterToSinkPnP	9	13	1	50
	CounterToCabPnP	8	12	1	50
	CloseDoor	N.A.	N.A.	1	50
DC	CounterToSinkPnP	10	67	100	10k
	CounterToCabPnP	10	67	100	10k
	CloseDoor	N.A.	N.A.	100	50
Prior	RoboCasa (20 tasks)	70	381	100	60k

TABLE III: **Aggregate dataset statistics of the Panda Kitchen domain.** We outline the number of object categories, object instances, scenes, and number of demonstrations for the real, digital cousin (DC), and task-agnostic prior (Prior) datasets.

	Dataset	Obj. Cat.	Table Tex.	Demos
Real	CupPnP	1	1	20
	MilkPnP	1	1	20
	Pouring	1	1	20
DC	CupPnP	1	1	1k
	MilkPnP	3	10	1k
	Pouring	2	10	1k
Prior	RoboCasa (10 tasks)	66	10	10k

TABLE IV: **Aggregate dataset statistics of the Humanoid Tabletop domain.** We outline the number of object categories, table textures, and number of demonstrations for the real, digital cousin (DC), and task-agnostic prior (Prior) datasets. Note that for `CupPnP`, we randomize the color of the object, although we are using a single object category.

demonstrations using MimicGen [9]. We further diversify these demonstrations by rendering them using combinations of the 100 floor, 100 wall, 100 cabinet, and 100 counter AI-generated textures provided by RoboCasa [7].

Humanoid Tabletop. Similarly, we design one digital cousin (DC) for each of the three tasks in the Humanoid Tabletop domain. We focus on the following factors when designing the DC: Task composition, Object composition, and Initialization distribution.

- **Task composition**: we set up the scene and the success check to make the simulation task semantically similar to the real-world task. During the simulation source demo collection, we try to mimic the motion pattern of the real-world demos. We add a small random offset (uniform distribution between $[-0.2, 0.2]$) to the initial joint distribution as real demos have different initial joints.
- **Object composition**: since we have a fixed set of objects in the real-world, we use a smaller set of objects in DC for the GR-1 humanoid, as noted in Table IV.
- **Initialization distribution**: we align object initialization regions for our DC with the real-world counterparts.

For our `CupPnP`, `MilkPnP`, and `Pouring` digital cousins, we generate 1000 synthetic demonstrations each from an initial source of 10 human demonstrations using DexMimicGen [10]. We instantiate the `MilkPnP` and `Pouring` digital cousins in RoboCasa, and we render them with 10 table textures.

F. Data Composition Analysis

We provide a detailed comparison of the real-world, task-aware digital cousin and task-agnostic prior datasets in Table III for the Panda Kitchen domain and Table IV for the Humanoid Tabletop domain.

We also add a detailed comparison of additional factors such as camera positions and initialization parameters for the `CounterToSinkPnP` task in Table V.

G. Training Details

Panda Kitchen. We adopt the open-source Diffusion Policy implementation from Chi et al. [38]. We use the transformer-based variant with ResNet visual encoders. The policy takes three 128×170 image views and robot proprioception information (end effector position and rotation, gripper joint values), and outputs a 7-DoF action for delta end-effector control and gripper action. We modify the hyperparameters to use a larger transformer network and a larger batch size of 256. We also add language conditioning to facilitate training on diverse multi-task data; we encode language using the CLIP sentence encoder, and we add FiLM conditioning layers [72] to the vision encoder. We set a default co-training ratio of 0.10 for real-world data and 0.90 for simulation data. We do this as we have a significantly higher quantity of simulation demonstrations than real-world demonstrations.

As mentioned in Section III-A, with a training batch of size B, on average, $\alpha \cdot B$ samples are drawn from the simulation dataset $\mathcal{D}_{\text{sim}}$, and $(1-\alpha) \cdot B$ samples are drawn from the real-world dataset $\mathcal{D}_{\text{real}}$. To enforce this, we reweight each data sample such that $P[(o_i, a_i) \in \mathcal{D}_{\text{sim}}] = \alpha$ and $P[(o_i, a_i) \in \mathcal{D}_{\text{real}}] = 1 - \alpha$, where (o_i, a_i) denotes an observation-action pair within the batch. We achieve this by first normalizing the weight of each sample according to the size of its dataset, and then multiplying the normalized weight by α if the sample belongs to $\mathcal{D}_{\text{sim}}$ and by $1 - \alpha$ if it belongs to $\mathcal{D}_{\text{real}}$.

Humanoid Tabletop. Since there is no ambiguity in terms of objects to manipulate, we use Diffusion Policy (DP) without language conditioning and train one policy for each task. We use the DP implementation from UMI [73] with Vision Transformers [74] as vision encoders and UNet [75] as the diffusion backbone. The input observation contains an RGB image from a first-person-view camera and joint position observations. The output action is the target joint positions of the arms and the dexterous hands. Figure 5 shows that co-training ratios α of 0.9 and 0.99 are optimal, and we use 0.99 for all other experiments.

H. Policy Evaluation

We use similar evaluation protocols for the Franka Panda arm and the GR-1 humanoid. We evaluate the model at three checkpoints during training, spaced at equal intervals. At each checkpoint, we assess the model's success rate and use the highest success rate among these evaluations as the final result.

Panda Kitchen. For both the `CounterToSink` and `CounterToCab` tasks, we perform four random placements on the counter per object and record the overall average success rate for all object categories. For the `CloseDoor` task we sample 10 random joint angles between [$85°$, $115°$]

	Left Cam Pos.	Right Cam Pos.	Wrist Cam Pos.	Init. Robot Joints	Init. Robot Base Pos.	Obj. Init. Reg.
`Real`	[-0.35, 0.49, 0.70]	[-0.35, -0.42, 0.72]	[-0.029, 0, 0.05]	(0.09, -0.20, -0.02, -2.47, -0.01, 2.30, 0.85)	[-18, -26, 0]	27x23
`DC`	[-0.35, 0.49, 0.70]	[-0.35, -0.42, 0.72]	[-0.029, 0, 0.05]	(0.09, -0.20, -0.02, -2.47, -0.01, 2.30, 0.85)	[0, -32, -4]	27x27
`Prior`	[-0.35, 0.49, 0.70]	[-0.35, -0.42, 0.72]	[-0.029, 0, 0.05]	(-0.02, -1.03, -0.02, -2.28, 0.04, 1.52, 0.70)	[0, -20, -22]	30x40

TABLE V: **Additional dataset statistics for `CounterToSinkPnP` task.** All units except for initial robot joints are in centimeters. Camera positions are (x,y,z) coordinates relative to the robot base. Initial robot base positions are (x, y, z) coordinates relative to the middle edge of the sink. Object initialization region is (depth x width)

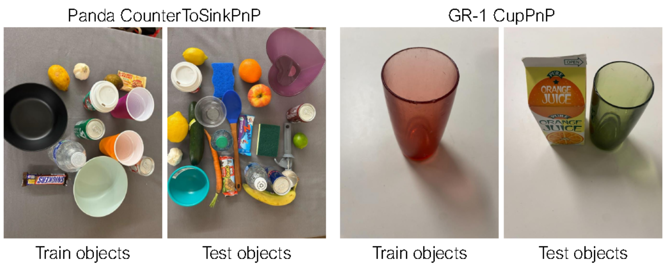

Fig. 8: **Visualization of novel object experiment settings.** We show the picture of train objects and test objects of the generalization experiment conducted on Panda `CounterToSinkPnP` and GR-1 `CupPnP` tasks.

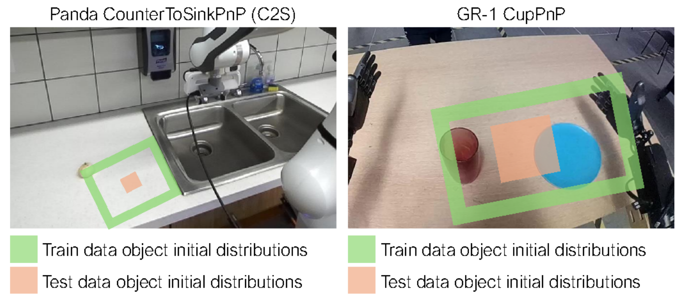

Fig. 9: **Visualization of novel position experiment settings.** We visualize the train and test object initialization range of the generalization experiment conducted on Panda `CounterToSinkPnP` and GR-1 `CupPnP` tasks.

and average the success rate across all sampled joint values; this results in 36, 32, and 10 total rollouts per checkpoint for the `CounterToSink`, `CounterToCab`, and `CloseDoor` respectively. For the `CounterToSink`, we record a success if the policy picks up the object and places it in the left or right basin of the sink, and record a failure otherwise. For the `CounterToCab`, we record a success if the policy picks up the object and securely places it in the cabinet, and record a failure otherwise. Finally, for the `CloseDoor` task, we record a success if the door's joint angle is less than 5° and record a failure otherwise.

The initialization regions and starting conditions (robot joint states, robot positions, object sampling regions) at test time are the same as during training. See Figure 7 for a visualization of the starting state and initialization regions for the tasks. For `CounterToCabPnP` and `CounterToSinkPnP`, we evaluate on seen objects, and for `CloseDoor`, we evaluate on the same door and use the same range of joint angles, [85°, 115°], from the training data.

Humanoid Tabletop. For each task, we evaluate the policy performance using the same objects and position distributions. For each checkpoint, we evaluate 20 different initial positions for the `CupPnP` task and 10 for the `MilkPnP` and `Pouring` tasks. We report the highest success rates among the three checkpoints for each training run. We consider partial successes, where a successful pick is counted as 0.5.

I. Generalization Experiment Details

This section visualizes the settings of the generalization experiments described in Section V-B. Figure 8 illustrates the training and testing objects used in the generalization experiments for the Panda `CounterToSinkPnP` and GR-1 `CupPnP` tasks. The default setup for the Panda `CounterToSinkPnP` task involves manipulating a diverse set of objects, whereas the GR-1 `CupPnP` task primarily uses the same red cup. Consequently, we observe a much greater performance improvement in the `CupPnP` task when employing sim-and-real co-training, as in Table II.

Figure 9 visualizes the training and testing object initialization ranges for the generalization experiments conducted on the Panda `CounterToSinkPnP` and GR-1 `CupPnP` tasks. In the training data, objects are always initialized at the borders of the workspace, while testing is performed with objects placed at the center of the workspace.

J. MultiTaskPnP Task Setup

In the multi-task experiment on the humanoid, we consider the following four tasks:

1) `Cuttingboard2Basket`: Pick up an object from the cutting board and place it into the nearby basket.
2) `Cuttingboard2Pan`: Pick up an object from the cutting board and place it into the nearby pan.
3) `Mat2Basket`: Pick up an object from the mat and place it into the nearby basket.
4) `Plate2Bowl`: Pick up an object from the plate and place it into the nearby bowl.

The tasks are visualized in Figure 10. The train and test objects we used are visualized in Figure 11. We collect 100 demonstrations for each task in the real world, where we randomly select an object and an initial location for each trajectory. At test time, we choose unseen instances of the same object category as the test objects, but the positions are within the demonstration distribution.

We create a digital cousin for each of the four tasks, using target objects and containers of the same category in simulation. Additionally, we align the initial distribution of object positions to approximate the distribution observed in real-world demonstrations. Once the simulation environments are set up, we collect 10 human demonstrations and generate 1000

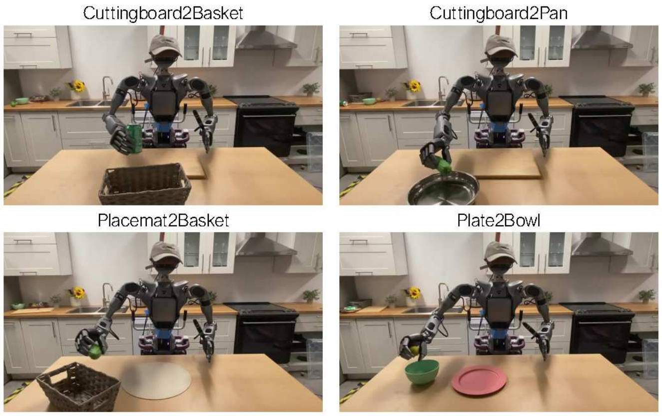

Fig. 10: **`MultiTaskPnP` visualization.** We show the real-world scene setup of the four tasks in `MultiTaskPnP`.

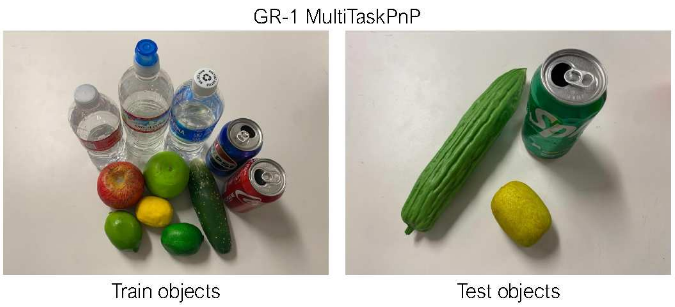

Fig. 11: **`MultiTaskPnP` train and test objects.** We show the real-world train and test objects we used for GR-1 `MultiTaskPnP`.

simulation demonstrations per task using DexMimicGen [10], resulting in a total of 4000 `DC` demonstrations.

We train a single policy across all four tasks, where the policy implicitly conditions on the image observation to determine the current task. For evaluation, we select three unseen objects—can, lemon, and cucumber (see Figure 11 right)—and assess performance across three different initial positions for each object in each task, yielding a total of 36 evaluations across the four tasks. We report the average performance across all tasks, considering partial successes, where a successful pick is counted as 0.5.

In Figure 4, we compare policies trained on `Real + DC`, where we use 4,000 total `DC` demonstrations and vary the number of real-world demonstrations, with 10, 25, 50, 75, and 100 real demonstrations per task. Results show that the co-trained policies consistently outperform the real-only policies.

We also compare co-training with different simulation data in this multi-task setting, similar to the single-task setting we show in Section V-A. We use 200 real demonstrations (50 per task), 4000 `DC` demos (1,000 per task), and 10,000 `Prior` demos (1,000 per task). Similar to Table I, we see that the policy trained on the combination of `Real`, `DC`, and `Prior` data performs the best, getting a success rate of 75.7%, followed by `Real + DC` (70.8%) and `Real + Prior` (68.8), both of which significantly outperform the `Real` policy (30.6%).

K. Improving Visual Realism with Vid2Vid

Given our observation that incorporating task-aware digital cousins, such as camera viewpoint alignment, enhances performance, we pose a follow-up question: *Would improving visual realism by making simulation rendering closer to reality enhance co-training?*

Several studies [23], [76]–[79] have suggested that training generative models (such as CycleGAN) on real and simulated images, and applying them to simulation images to make them visually more realistic, improves sim-to-real policy transfer. Inspired by this, we conduct experiments to assess the extent to which improving the visual realism of simulation data enhances policy performance.

Specifically, we fine-tune CogVideo-X [80], a state-of-the-art video diffusion model, on `CupPnP` real demonstration videos in the Text2Video modality. This enables the model to generate realistic-looking videos of a robot performing the task from pure Gaussian noise.

To adapt the model for style transfer on simulation videos—preserving object positions while enhancing visual realism—we adopt a simple strategy: instead of generating videos from pure noise, we introduce noise into reference simulation videos and use them as initialization for video diffusion. By adjusting the noise strength parameter, we control the extent of noise added and the diffusion starting timestep (see Figure 12 for examples). Lower noise levels retain more of the original simulation textures, producing outputs closer to the inputs, while higher noise levels result in more realistic appearances but may distort object poses.

We set the noise strength to 0.6, as lower values yield outputs too similar to simulation textures, whereas higher values cause excessive deviations from object positions, rendering the original action labels from simulation data inapplicable.

We then conduct co-training experiments in the `Real + DC` setup on the `CupPnP` task, comparing policy performance with and without Vid2Vid augmentations on DC data under different numbers of sim and real demos similar to Figure 4.

We find that improved visual realism is particularly beneficial in low-data regimes and provides a modest improvement in overall policy performance. Experiment results, reported in Table VI, show that Vid2Vid-enhanced visual realism leads to a 5–10% average improvement in policy performance, with the most significant benefits occurring when the number of simulation or real-world trajectories is low. When sufficient real-world demonstrations are available, simulation data plays a minor role. Conversely, when a large and diverse set of simulation data is available, the importance of visual realism diminishes.

These findings highlight the potential of the role of generative models as part of the synthetic data. While our approach utilizes a renderer to provide reference videos for the video diffusion model, future work could explore synthetic generation without reliance on the graphics renderer.

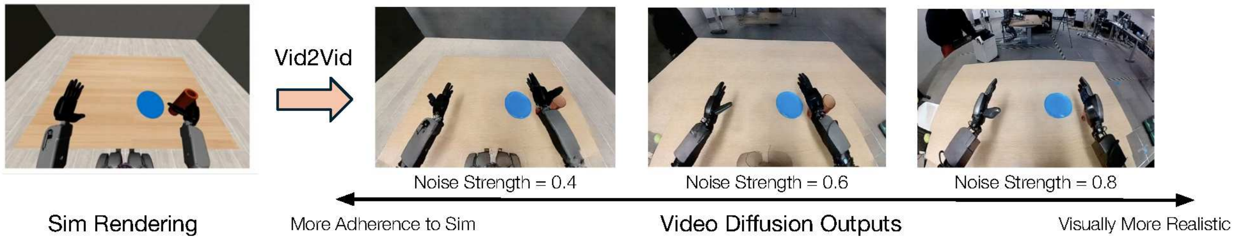

Fig. 12: **Examples of the Video2Video model outputs with different noise strength. Left:** An example video frame from the simulation data. **Right:** The corresponding frames generated by the trained video diffusion model by initializing the noises with the simulation video with different noise strengths. By setting different values for the noise strength parameter, we can control how much noise is added and from which timestep the model starts diffusion.

Real 20 +	Sim 20	Sim 100	Sim 200	Sim 500	Sim 1000
`Real + DC`	48%	73%	85%	88%	95%
`Real + DC w/ V2V`	70%	80%	88%	93%	95%
Sim 1000 +	Real 1	Real 5	Real 10	Real 20	Real 50
`Real + DC`	8%	40%	73%	95%	95%
`Real + DC w/ V2V`	13%	53%	83%	95%	95%

TABLE VI: **Effects of improved visual realism with Vid2Vid models on policy performance in various numbers of simulation and real demonstrations.** We compare the policy co-trained on `Real + DC` for the `CupPnP` task where `DC` is augmented to be visually more realistic using the Vid2Vid model (`DC w/ V2V`) compared to using the OpenGL-based simulation renderings (`DC`).

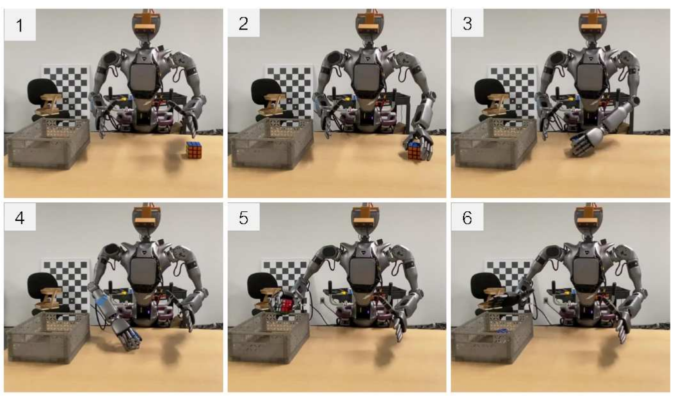

Fig. 13: **Visualization of `BimanualPnP` task.** We show one rollout of our co-trained policy on `BimanualPnP` task. This task involves bimanual and long-horizon manipulation.

L. Training CloseDoor with More Demos

Due to the large performance gap between `real` and all other co-trained policies for the `CloseDoor` task, we investigate whether this gap can be easily closed by training the `real` policy on more demonstrations. Specifically, we train the `real` policy on 100 demos instead of 50. We find that the policy still does not achieve 100% success, only getting an 80% success rate despite training with double demos.

M. FAQs

Is dynamic alignment investigated? We have attempted to align dynamics by tuning physics parameters to reduce the gap between open-loop rollout results in simulation and the real world. However, our experiments on GR-1 `CupPnP` show no difference in success rate (95%) with or without dynamic alignment, suggesting that such alignment is unnecessary for our tasks.

How misaligned is the default camera? Since perfect camera alignment between simulation and the real world is infeasible, quantifying the exact camera pose error is challenging. Instead, we compute the delta pose between the default camera and the "aligned" camera. In Panda, the position and orientation deltas for the third-person cameras are 37 cm and 20°, respectively, while for the wrist camera, the deltas are 9 cm and 180°. In GR-1, the position delta is 36 cm, and the orientation delta is 60°. A visualization of the camera views is provided in Figure 6.

How does co-training compare to domain randomization and domain adaptation? We find that domain randomization and domain adaptation are complementary to co-training but not strictly necessary to reap the benefits of co-training with simulation. For instance, in the Panda Kitchen domain, we observe that adding generative texture randomization to the `digital cousin` data further improves performance, but even without it, co-training alone still leads to performance gains. Similarly, in the Humanoid Tabletop domain, enhancing visual realism via a Video2Video model can boost co-training results, but the gains are marginal in some cases.

Are there plans for bimanual tasks? We introduce a GR-1 `BimanualPnP` task, where the humanoid must use its left hand to pick up a Rubik's cube and place it at the center of the table, then use its right hand to pick up the cube and place it into a basket, as shown in Figure 13. Our results show that with 50 real demonstrations, the policy achieves a 15% success rate. When co-trained with 1,000 DC simulation demonstrations and 50 real demonstrations, the success rate improves to 50%. In contrast, a policy trained only with 100 real demonstrations achieves a 30% success rate. We also briefly tested co-training with our current task-agnostic prior datasets. As discussed in Section VIII-D, all our humanoid prior datasets consist of single-arm pick-and-place data. When co-trained with these datasets using a 99% co-training ratio, the policy exhibits single-arm pick-and-place behavior instead of the expected bimanual behavior, resulting in a near-zero success rate. This suggests that for task-agnostic prior datasets to effectively support real-world manipulation, the behavior patterns in simulation and the real-world need to be consistent.