X-SIM: Cross-Embodiment Learning via Real-to-Sim-to-Real

Anonymous Authors

Abstract: Human videos offer a scalable way to train robot manipulation policies, but lack the action labels needed by standard imitation learning algorithms. Existing cross-embodiment approaches try to map human motion to robot actions, but often fail when the embodiments differ significantly. We propose X-SIM, a real-to-sim-to-real framework that uses object motion as a dense and transferable signal for learning robot policies. X-SIM starts by reconstructing a photorealistic simulation from an RGBD human video and tracking object trajectories to define object-centric rewards. These rewards are used to train a reinforcement learning (RL) policy in simulation. The learned policy is then distilled into an image-conditioned diffusion policy using synthetic rollouts rendered with varied viewpoints and lighting. To transfer to the real world, X-SIM introduces an online domain adaptation technique that aligns real and simulated observations during deployment. Importantly, X-SIM does not require any robot teleoperation data. We evaluate it across 5 manipulation tasks in 2 environments and show that it: (1) improves task progress by 30% on average over hand-tracking and sim-to-real baselines, (2) matches behavior cloning with 10× less data collection time, and (3) generalizes to new camera viewpoints and test-time changes.

Keywords: Learning from Human Videos, Sim-to-Real, Representation Learning

1 Introduction

Human videos offer a natural and scalable source of demonstrations for robot policy learning. However, recent advances in robot foundation models [1, 2] rely entirely on large-scale datasets of robot embodiments [3, 4]. Collecting such data requires labor-intensive and expensive teleoperation to provide high-quality expert demonstrations, making it intractable to scale across diverse tasks and environments. In contrast, human videos (e.g. from YouTube) are abundant and capture a wide range of tasks in natural environments.

Despite their potential, human videos cannot be directly used in widely-adopted imitation learning pipelines [5, 6], as they lack explicit robot action labels. To bridge this gap, prior work attempts to map human trajectories to robot actions, typically assuming visual or kinematic compatibility. Some methods retarget human hand motion to the robot's end-effector [7], but this assumes that human movements are feasible for the robot to replicate [8], which is rarely the case in practice. Other methods reduce the human-robot visual gap by overlaying robot arms on human videos [9, 10], but these rely on solving inverse kinematics, which may be ill-posed due to embodiment mismatch. Another line of work directly translates human videos into robot actions [11, 12, 13], but requires paired human-robot demonstrations, which are expensive and difficult to collect at scale.

We tackle the problem of generating robot training data from action-less human videos. ***Our key insight is that, while human actions are unavailable, the object motion they produce provides a dense supervisory signal for training robot policies in simulation.*** By reconstructing a photorealistic simulation [14] of the human video and tracking object trajectories [15], we define object-centric reward functions that guide RL agents to reproduce the effects of human behavior — even when the robot must take entirely different actions. This enables distillation into real-world image-conditioned robot policies ***without any robot teleoperation data***.

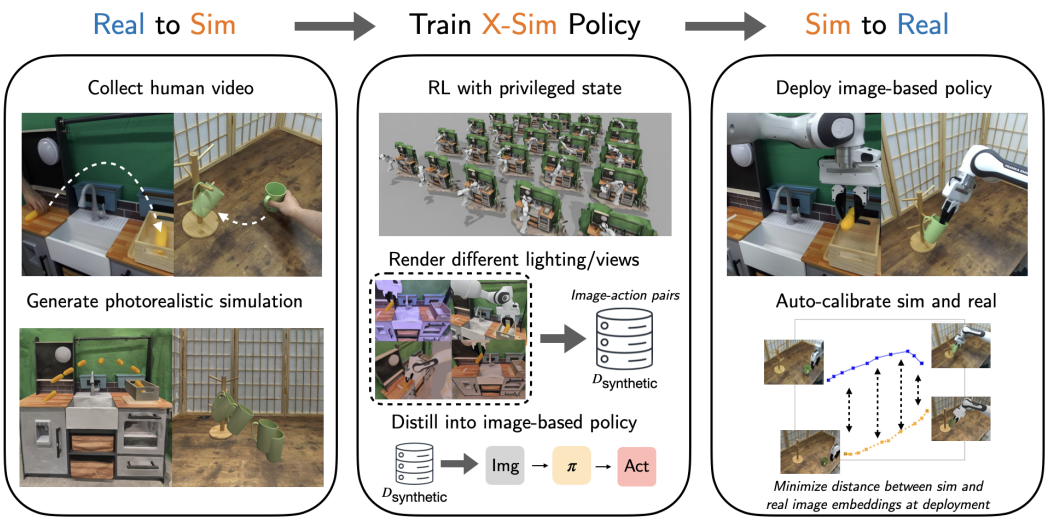

Figure 1: **Overview of X-SIM:** We introduce X-SIM, a real-to-sim-to-real framework that bridges the human-robot embodiment gap by learning robot policies. **Real-to-Sim**. We generate photorealistic simulation using object-centric rewards generated from human videos. **Training X-Sim**. We first train RL policies with privileged state using GPU-parallelized environment. Then, we collect a diverse image-action dataset use it to distill behaviors into an image-conditioned policy. **Sim-to-Real.** Image-based policy is deployed in the real-world. Its observation encoder automatically calibrates itself by aligning real and sim image observations at test-time.

We propose X-SIM, a real-to-sim-to-real framework that bridges the human-robot embodiment gap by learning robot policies in simulation on rewards generated from human videos (Fig. 1). X-SIM first extracts object states from a RGBD human video and transfers them into a photorealistic simulation. It defines a dense object-centric reward to efficiently train state-based RL policies in simulation. X-SIM generates a large synthetic dataset of paired image-action data by rolling out the trained RL policy and rendering the resulting scenes under varied robot poses, object states, viewpoints, and lighting. Using this dataset, it trains an image-conditioned diffusion policy and transfers directly to the real-world without needing any real robot action data. To narrow the sim-to-real gap at deployment, X-SIM utilizes an online domain adaptation technique to align the robot's real world and simulation observations. Our contributions are summarized as follows:

1. We propose X-SIM, a real-to-sim-to-real framework that learns image-based robot policies from action-less human videos by tracking object states and matching their motion in simulation.
2. We introduce an online domain adaptation technique to continually reduce the sim-to-real gap by aligning real-world observations with simulation at test time, enabling robust sim-to-real transfer.
3. We evaluate X-SIM across 5 manipulation tasks in 2 environments, showing that it (1) improves task progress by 30% on average over hand-tracking and sim-to-real baselines, (2) matches behavior cloning with 10x less data collection time, and (3) enables generalization to test-time environment changes, including novel camera viewpoints.

2 Related Work

Imitation Learning. Imitation learning, particularly behavior cloning (BC), is the dominant paradigm for training visuomotor robot policies. Recent algorithms like Diffusion Policy [5] and ACT [6] achieve state-of-the-art results by learning from expert demonstrations consisting of image-action pairs. However, these methods typically require collecting data via human teleoperation of the specific target robot, using kinesthetic teaching [16], wearable devices [17], or specialized control interfaces [18, 19, 20]. Recent efforts have attempted to build large robotic dataset across different robot embodiments [3, 4] leading to the development of foundation models [1, 2] for robotic control. Still, scaling up such datasets remains a significant challenge because of the heavy reliance on robot teleoperation. While UMI [21] proposes hand-held grippers to collect data without direct robot involvement, these demonstrations can be dynamically infeasible for robots and still require

active collection in lab settings. In contrast, our approach bypasses the need for robot action data entirely by leveraging human videos to generate synthetic robot data.

Learning from Human Videos. The ease of collecting human videos has motivated interest in learning robot motion directly from them. Common strategies include retargeting hand motion [22, 23, 24, 25, 7], reducing the visual gap via inpainting [26, 9, 10], or using pretrained open-world vision models for constructing object-relative hand trajectories [27, 28, 29]. All of these methods rely on the robot's capability to match its end-effector with the human's hand positions, which often falls down in practice due to large embodiment differences. Hierarchical frameworks [13, 30, 31] learn high-level plans instead, while one-shot imitation methods [11, 12, 32] learn from prompt videos. These methods typically require human-robot paired data or self-supervised alignment from unpaired data [**?** 33]. In either case, a common limitation among these methods is the need for robot teleoperation data to guide low-level control [8]. RL provides an alternative, using video [34] similarity, language matching [35] or object tracking [36] for rewards, but suffers from the sim-to-real gap. Cross-embodiment RL [37, 38] methods that have been deployed on real robots require object tracking at test-time which can be brittle to noisy observations. Instead, we leverage a real-to-sim-to-real pipeline to directly transfer image-based policies from simulation.

Real-to-Sim-to-Real. Advances in 3D computer vision have enabled the development of photorealistic, physically accurate simulations from real-world data. Recent works increasingly use real-to-sim methods to learn robot behaviors in simulation. For instance, RialTo [39] trains RL policies in simulation to improve policy robustness, using point cloud inputs for real-world deployment. ResiP [40] learns residual actions in simulation starting from an image-based policy trained in the real world. However, both these approaches still require real-world robot data collection. To directly learn actions, motion planners are used in simulation but deployed open-loop in the real world [41, 42]. More recently, real-to-sim-to-real has been applied to learn from human videos [43, 44]. However, Video2Policy [43] only extracts the initial and final object states from human videos, and relies on object segmentation masks at test time for policy transfer. Human2Sim2Robot [44] defines rewards for RL using object state tracking from videos, but does not use a photo-realistic simulation. However, real-world deployment additionally requires object tracking at test time. RL training also requires tracking human hand trajectories for guiding the policy, and is applied only to dexterous hands with minimal embodiment gap. Our work offers distinct advantages over these methods: (a) we bypass the need for robot teleoperation data and human hand tracking for RL training, and (b) we transfer image-based policies from simulation to the real world using environment randomization and domain adaptation methods.

3 Approach

X-SIM addresses the problem of training real-world, image-conditioned robot policies from action-less RGBD human videos by using object motion as a transferable supervision signal. The framework consists of three stages: (1) reconstructing a photorealistic simulation environment from the human video and extracting object trajectories to define dense object-centric rewards; (2) training privileged-state reinforcement learning (RL) policies in simulation to reproduce the observed object motion, and generating synthetic image-action data through rollouts; and (3) distilling the learned behaviors into an image-conditioned diffusion policy, and deploying it in the real world with an online domain adaptation technique that continually aligns simulated and real observations.

3.1 Real-to-Sim Transfer from Human Videos

While human videos do not provide direct supervision for robot actions, the resulting object motion can serve as a transferable task specification. This stage reconstructs a realistic simulation environment and extracts object trajectories from the human video, enabling policy learning through object-centric rewards (Fig. 2).

Object Pose Tracking. We first use an off-the-shelf 3D scanning app [45] to obtain high-fidelity object meshes of items being manipulated. To track these objects across the video, we employ

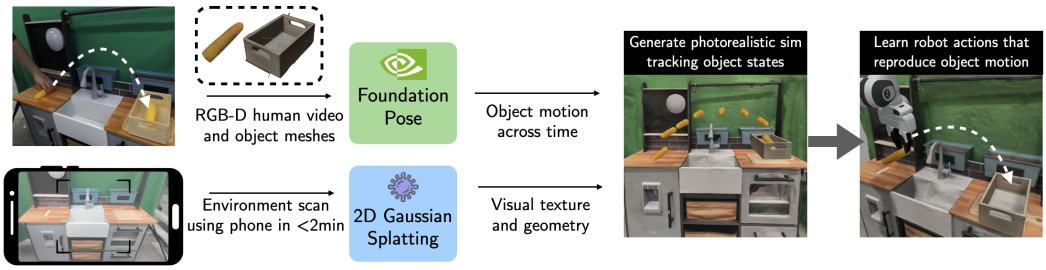

Figure 2: **Real-to-Sim:** X-SIM reconstructs a photorealistic environment with accurate geometry from multi-view images. It tracks object motion across time from an RGBD human video to define a dense object-centric reward function to train RL policies in simulation.

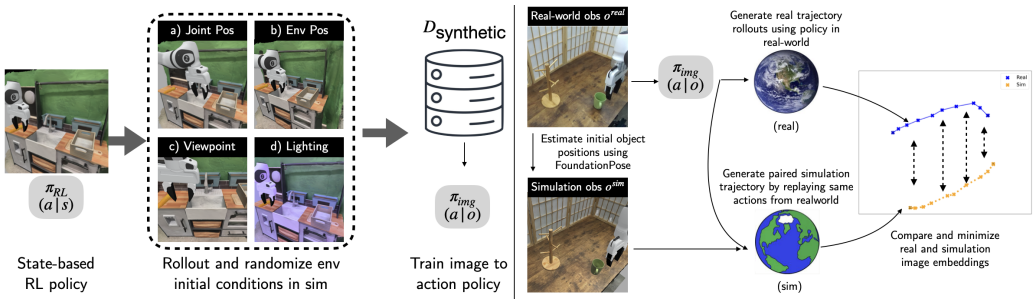

Figure 3: **Sim-to-Real: (Left)** X-SIM distills privileged-state policies into image-conditioned policies by generating and a synthetic image-action dataset using multiple environment randomizations. **(Right)** During deployment, real policy rollouts are replayed in simulation to generate paired images across real and sim. Their discrepancy is utilized to minimize and calibrate the sim-to-real visual gap.

FoundationPose [15] which takes in the human video, the 3D mesh of each object and a 2D mask identifying the objects in the first frame of the video, generated by Segment Anything (SAM) [46]. FoundationPose tracks the position and rotation of each object over the course of the video. Formally, given a human video $\mathbf{v_H} = \{v_H^t\}_{t=1}^T$ where $v_H^t \in \mathbb{R}^{H \times W \times 4}$ is the RGBD image at timestep t, we convert $\mathbf{v_H}$ into $\mathbf{s_H} = \{s_H^t\}_{t=1}^T$ where $s_H^t \in \mathrm{SE}(3)^K$ represents the position and rotation of K objects being tracked in the scene.

Environment Reconstruction. We next construct a geometrically accurate and photorealistic environment mesh using 2D Gaussian Splatting [14], an open-source module that performs 3D reconstruction from multi-view images. This reconstructed environment is then transferred directly into the ManiSkill [47] simulator along with the object states. Physical properties and dynamics are set to default values for simplicity, though the approach is compatible with system identification [48, 49], domain randomization [50, 51, 52], and additional methods to handle articulations [53, 54].

3.2 Generating Robot Actions in Simulation

To bridge the embodiment gap and obtain robot actions to complete the task specified by the human video, we define an object-centric reward function to train a privileged-state policy via RL. Then, we rollout the policy and render the scene under varied robot poses, object states, viewpoints, and lighting conditions to collect a synthetic dataset of image-action pairs (Fig. 3, Left).

Defining Object-Centric Rewards. We use the human video's object pose trajectory to define an object-centric reward function, with each goal indicating where the objects should be positioned and oriented. The reward encourages the robot to move objects toward their next desired pose, as specified by the human trajectory. For a goal state s_H^B, the reward is defined as:

$$r_{\text{goal}} \propto -d_{pos}(s_H^B, s_R^t) - d_{rot}(s_H^B, s_R^t) \tag{1}$$

where the objects current positions and rotations are encouraged to match the next goal. In practice, there is an additional default reward which brings the robot end-effector near the relevant objects

$r_{\text{obj}} = r_{\text{approach}} + r_{\text{goal}}$. The policy's privileged state includes information about the current object states, goal states, and robot proprioception, and the goal is updated online as each target is achieved, enabling multi-step object manipulation (see Appendix for details).

Collecting Synthetic Image-Action Data. We train a robot policy using Proximal Policy Optimization (PPO) [55] to optimize our object-centric reward r_{obj}. This policy learns to predict actions that enable the robot to manipulate objects matching the human demonstration. To learn robust behaviors, we randomize the starting pose of objects during RL training. The RL policy conditions on privileged simulation states $\pi_{\text{RL}}(a|s)$ to output robot actions. After training the RL policy, we roll it out in our simulator to generate synthetic data of image-action pairs, only keeping successful trajectories. We systematically vary the simulation conditions by randomizing object starting positions, camera viewpoints, and lighting conditions across different rollouts. Each robot rollout is defined as $\xi_R = \{(o_R^t, a_R^t)\}_{t=1}^N$, an N step trajectory of RGB image $o_R^t \in \mathbb{R}^{H\times W\times 3}$ and action a_R^t pairs. This process builds a diverse synthetic dataset $D_{\text{synthetic}}$ suitable for image-conditioned policy training.

3.3 Sim-to-Real Transfer of Image-Based Policies

We distill robot behaviors into image-conditioned policies trained on synthetic image-action pairs generated in simulation. To improve real-world transfer, we introduce an online domain adaptation technique that collects real image observations from closed-loop rollouts and automatically pairs them with simulated views of the same robot trajectories, then used to minimize the sim-to-real visual gap (Fig. 3, Right). Notably, this procedure does not require any robot teleoperation data.

Training Image-Conditioned Policies. Given the synthetic dataset $D_{\text{synthetic}}$, we train an image-conditioned policy that operates directly on RGB observations without access to privileged state. We employ Diffusion Policy (DP) [5], a state-of-the-art behavior cloning architecture, to predict actions given the current image observation. The policy $\pi_{\text{img}}(a|o)$ takes the current RGB observation o_R^t as input and predicts a sequence of actions $\mathbf{a_R} = \{a_R^h\}_{h=1}^H$ over a horizon H to complete the task.

Auto-Calibration of Real and Sim. While our photo-realistic simulation enables zero-shot transfer to the real world, visual discrepancies between simulated and real observations can still limit performance. To address this, we introduce an online domain adaptation technique that allows our policy's observation encoder to focus on task-relevant features by aligning images across domains. After deploying our initial policy in the real world, we collect image observations from these robot rollouts, **even including failures**. We then spawn our simulation with same initial state as the rollouts (using FoundationPose on the first frame), and replay the exact robot trajectories in simulation to create a paired dataset of real and simulated images (Fig 3, Right). The paired images can be used to supervise the policy's observation encoder by encouraging image pairs to map to the same embeddings while being distinguishable from images corresponding to different environment states.

This yields a dataset of paired observations $D_{\text{paired}} = \{(o_R^{\text{sim}}, o_R^{\text{real}})\}_{i=1}^D$. During training, we supervise the policy with a standard behavior cloning loss on $D_{\text{synthetic}}$ and apply an additional contrastive InfoNCE loss [56] on D_{paired}:

$$\mathcal{L}_{\text{calibration}} = -\sum_{(o_R^{\text{sim}}, o_R^{\text{real}})\in D_{\text{paired}}} \frac{\exp(s(\phi(o_R^{\text{sim}}), \phi(o_R^{\text{real}}))/\tau)}{\sum_{(_, o_R'^{\text{real}})\in D_{\text{paired}}} \exp(s(\phi(o_R^{\text{sim}}), \phi(o_R'^{\text{real}}))/\tau)} \tag{2}$$

Here, ϕ is the policy's image encoder, s is cosine similarity, and τ is a temperature hyperparameter. This loss pulls together embeddings of corresponding simulation and real images, while pushing apart mismatched ones—guiding the encoder to focus on task-relevant semantics and reducing overfitting to simulation-specific features.

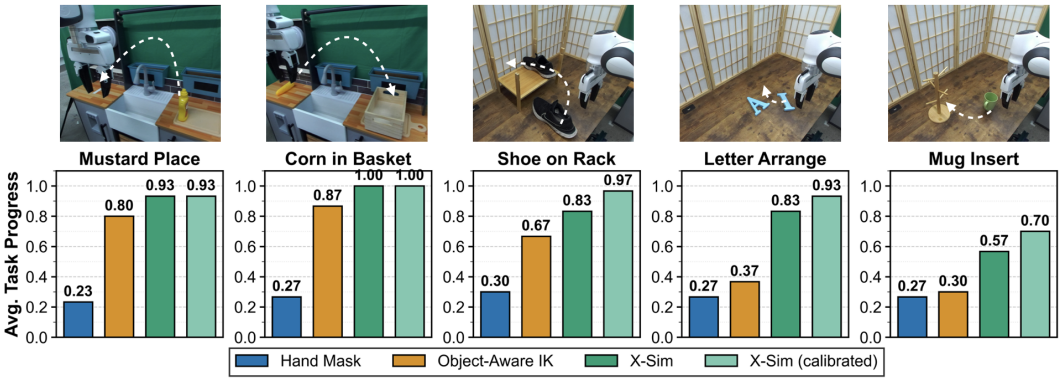

Figure 4: **Performance on Real-World Tasks:** We report *Avg. Task Progress* on 5 tasks across two environments, and find that X-SIM both with and without calibration consistently outperforms hand-tracking baselines that attempt to retarget human hand motion. A rough sketch of each task is visualized on top.

4 Experiments

We evaluate whether X-SIM can generate synthetic data from action-less human videos that is sufficient to train high-performing real-world robot policies. Our experiments span 5 tasks across two environments and aim to answer the following core questions:

1. **Bridging the Embodiment Gap via Simulation:** How does X-SIM perform across a variety of tasks compared to hand-tracking baselines when given a single human demonstration video?
2. **Sim-to-Real Policy Transfer:** What is the practicality of X-SIM's sim-to-real transfer of image-based policies versus alternate observation representations?
3. **Data Efficiency:** How does X-SIM's performance scale with time spent on data collection compared to behavior cloning methods with teleoperated robot data?
4. **Robustness to Test-Time Changes:** In what ways can X-SIM generate synthetic data to enable real-world policy robustness beyond standard data collection procedures?

Experimental Setup. We conduct all experiments using a 7-DOF Franka arm across two real environments: *Kitchen* and *Tabletop* (Fig. 4). RGBD human videos are recorded using a ZED 2 stereo camera, with no constraints on motion or grasp style allowing for natural human execution. Tasks include pick-and-place (`Mustard Place`, `Corn in Basket`, `Shoe on Rack`), non-prehensile manipulation (`Letter Arrange`), and precise insertion (`Mug Insert`). We transfer human videos into simulation using our real-to-sim pipeline. For each task, we train privileged-state policies using PPO [55] in ManiSkill [47] and randomize object and robot poses around the initial demonstration state. Then, the RL policy is distilled into an image-only Diffusion Policy [5]. We assume approximate knowledge of the test-time camera viewpoint and render randomized viewpoints around it during training, adding robustness to small variations. At inference time, X-SIM operates solely on real RGB image input. To align the observation encoder for X-SIM (CALIBRATED), we rollout 10 trajectories of X-SIM in the real-world to collect paired real and sim data. More details about each task and hyperparameters are in the Appendix.

Evaluation Metrics. We report *Average Task Progress* as our primary metric, which captures partial credit across distinct stages of task completion rather than relying on binary success. For grasp-based tasks (`Mustard Place`, `Corn in Basket`, `Shoe on Rack`, `Mug Insert`), progress is divided into three stages: approaching the correct object, successfully grasping it, and completing the manipulation to match the goal configuration from the human video. For the non-prehensile task (`Letter Arrange`), the stages correspond to approaching, rotating, and placing the object correctly. We evaluate all methods over 10 trials, each with slight variations in the object's initial position relative to the demonstrated human video.

4.1 Bridging the Embodiment Gap via Simulation

We evaluate whether X-SIM can overcome the limitations of hand-retargeting approaches. We compare against two representative baselines:

- **Hand Mask:** [9, 10] Applies a black mask over the human hand in demonstration videos to train an image-conditioned behavior cloning policy. At inference time, the robot arm is similarly masked. This approach, used in PHANTOM [10], assumes all human hand poses can be replicated by the robot. Without this assumption, we do not overlay a robot arm during training.
- **Object-Aware Inverse Kinematics (IK):** [28, 29, 56] Extracts hand trajectories relative to nearby objects, and replays them by applying IK to move the robot end-effector along the same path.

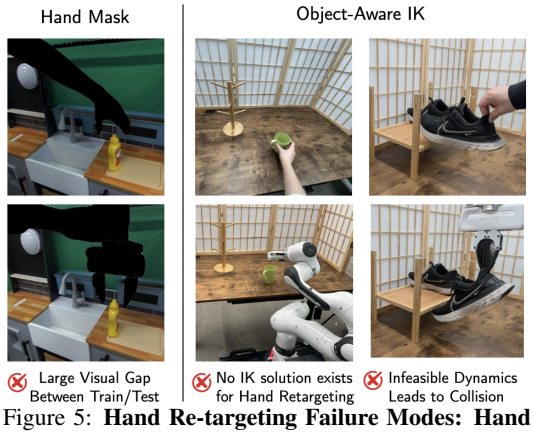

Figure 5: **Hand Re-targeting Failure Modes: Hand Mask** fails due to a significant visual domain gap between human and robots, even when the motions are physically feasible for the robot. **Object-Aware IK** fails under execution mismatch, as certain human hand motions are kinematically or dynamically infeasible.

Neither baseline uses simulation. Both extract action labels from human hand pose estimates using HAMER [57], using the same procedure as PHANTOM [10]. We evaluate X-SIM and baselines across 10 real-world rollouts per task (Fig. 4). **Hand Mask** fails due to a large visual gap between human and robot observations, retaining only object location information and rarely progressing beyond the approach phase (Fig. 5). **Object-Aware IK** performs well in *Kitchen* tasks where human and robot have similar execution styles, but breaks down in *Tabletop* tasks due to kinematic infeasibility and mismatched dynamics when directly mimicking human motions. In contrast, **X-SIM**, even without sim-to-real calibration, learns feasible strategies in simulation and transfers them effectively—achieving consistently higher task progress and over 30% gains in the most mismatched settings.

4.2 Sim-to-Real Policy Transfer

Comparison with State-Based Policy. We evaluate X-SIM's ability to transfer from simulation to the real world using only RGB images, and compare it to policy learning approaches based on privileged state, such as object poses. A closely related method, **Human2Sim2Robot** [44], learns in simulation using accurate 6D object poses and attempts to replicate this setup in the real world through object tracking. However, even small tracking errors at inference can push pose-based policies out-of-distribution, leading to failure. These methods often rely on precise observations that are hard to obtain in practice due to occlusions, depth noise, and imperfect vision models. In contrast, **X-SIM** uses raw images, which provide a more robust and transferable representation. Image-based inputs are less sensitive to real-world noise and align well with modern visuomotor policy architectures. On the `Letter Arrange` task, X-SIM significantly outperforms pose-based baselines in sim-to-real transfer (Table 6), showing that images are a more practical and effective observation modality for real-world deployment.

Metric ↓	H2S2R	X-SIM
Avg. Task Progress	43.3%	83.3%

Figure 6: We evaluate Avg. Task Progress of X-SIM with image observations against a sim-to-real baseline that uses object state observations on the `Letter Arrange` task.

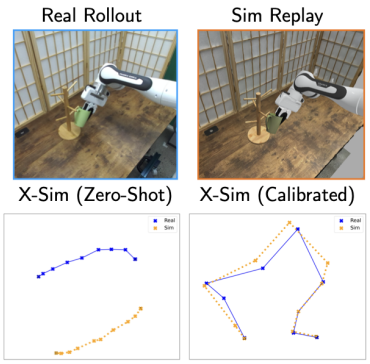

Figure 7: **Sim-to-Real Calibration:** We compare X-SIM image embeddings using t-SNE before and after calibration for one rollout.

Calibration after Deployment. Recent sim-to-real methods [40, 58] often rely on co-training with real-world demonstrations to bridge the domain gap. In contrast, X-SIM uses only simulation data collected in a photorealistic environment, avoiding the need for teleoperation. While this reduces the

observation gap, some visual discrepancies remain due to imperfections in 3D reconstruction and rendering. To address this, X-SIM (CALIBRATED) aligns real and simulated observations online using closed-loop rollouts, as described in Sec. 3.3. Notably, this procedure is agnostic to success/failure, and can even benefit from unsuccessful rollouts. We find that X-SIM (CALIBRATED) leads to additional benefits over our base method, with an average increase of 8% in task progress across all tasks and most notably a 13% increase for the most challenging task `Mug Insert`, indicating the ability to learn even from failures (Fig. 4). To further analyze the effects of our calibration procedure, we probe policy observation encoders on a paired simulation/real robot videos and plot the t-SNE embeddings over time in Fig. 7. X-SIM (CALIBRATED) better aligns image embeddings compared to X-SIM, ensuring that the policy avoids overfitting to domain-specific attributes with its calibration loss while still encoding task relevant features with its action prediction loss.

4.3 Data Efficiency

We study how X-SIM's performance scales with data by modifying the `Mustard Place` task to significantly broaden the initial state distribution of the mustard bottle (visualizations in the Appendix). In this setting, behavior cloning requires extensive robot teleoperation data to cover the distribution. In contrast, X-SIM scales by collecting more human videos—which are faster to obtain (20s per video vs. 60s per robot demo)—and perturbing object poses in simulation for broader coverage. As shown in Fig. 8, X-SIM achieves 90% success with just 1 minute of human video data, compared to 70% success with 10 minutes of robot demonstrations. This highlights X-SIM's efficiency and scalability for training robust robot policies.

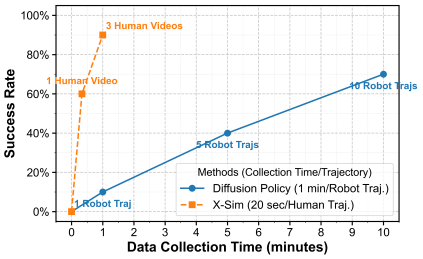

Figure 8: **Data Efficiency:** X-SIM scales more efficiently with data collection time than behavior cloning from robot teleoperation, achieving comparable success on `Mustard Place` with 10× less time.

4.4 Robustness to Test-Time Changes

Image-conditioned policies are particularly sensitive to viewpoint bias, demanding additional data for each perspective. X-SIM overcomes this by leveraging simulation to render trajectories from any desired view, enabling efficient coverage. We evaluate this by collecting simulated rollouts from *Side* and *Frontal* camera views, and training policies with data from each view individually and jointly for `Shoe on Rack`. As shown in Fig. 9, combining diverse viewpoints in simulation significantly improves generalization, even to unseen camera angles. More details are in the appendix.

Train → Test ↓	Side	Frontal	Side & Frontal
Side	83.3%	23.3%	**96.7%**
Frontal	23.3%	76.7%	**80.0%**
Novel	33.3%	30.0%	**53.5%**

Figure 9: We show that we can flexibly collect image-action data in simulation from multiple viewpoints (Side and Frontal) with X-SIM and train robust policies that generalize to novel viewpoints (`Shoe on Rack`).

5 Discussion

X-SIM presents a scalable framework for learning robot manipulation policies from human videos without requiring action labels or robot teleoperation. By leveraging object motion as a dense supervisory signal and training in photorealistic simulation, X-SIM bridges the embodiment gap and transfers image-conditioned policies to the real-world using online domain adaptation. Across five manipulation tasks, it improves task progress on average by over 30% compared to hand-tracking baselines, matches behavior cloning performance with 10× less data, and generalizes to novel viewpoints and test-time changes. While this work focuses on training policies from scratch, the same real-to-sim-to-real approach can naturally extend to fine-tuning pre-trained robot learning models, including foundation models such as PI-0.5 [2] and OpenVLA [1]. By generating targeted synthetic rollouts conditioned on human video demonstrations, X-SIM could adapt generalist policies to new tasks and embodiments in a low-cost, scalable manner and offers a complementary path toward efficient specialization of large robot models without requiring new robot data.

6 Limitations

In this paper, we chose to maximize the ability of the real-to-sim pipeline by making simplifying assumptions, while still maintaining the input/output contract (images to actions) that is most practical to deploy in unstructured environments. This is because the focus of the paper is to show the effectiveness of image-based policy transfer given ideal real-to-sim transfer. However, we acknowledge that while X-SIM provides an effective approach for learning robot policies from human videos, its application to unstructured, in-the-wild internet videos remains an open challenge. Below, we outline key assumptions that limit X-SIM's current ability to move towards this broader vision and suggest pathways towards their solutions in the near future:

Requiring Object Meshes for Tracking. Our pipeline currently uses FoundationPose, which requires a 3D object mesh for tracking, limiting applicability to videos where we either don't know or don't have the object mesh manipulated by the human. One way to extend this to internet videos is by estimating approximate meshes directly from using tools like InstantMesh [59]. Alternatively, object meshes can be retrieved from large 3D asset libraries [60], as shown in prior work on digital cousin generation [61], which suffices since simulation is only used for synthetic data generation.

Restricted to Rigid Object Manipulation. Our current pipeline relies on tracking object states through 6D poses, which limits it to rigid objects and excludes articulated or deformable items commonly seen in real-world tasks. For articulated objects like drawers or doors, recent vision research [62] has shown that visual priors and foundation models can be used to identify and track articulation parameters from RGB input. For deformable objects, emerging representations like particle-based models [63] offer promising avenues for capturing non-rigid dynamics. While these approaches are still maturing, our framework can continue to improve rigid manipulation skills, and its image-conditioned policies may complement existing models trained on separate data to handle deformables and articulations more effectively.

Environment Scan for Generating Simulation. X-SIM currently requires an explicit 3D scan of the environment to reconstruct the simulation scene, which limits its applicability to scenarios where such scans are unavailable. Recent works like St4RTrack [64] have shown that it is possible to generate both geometric and visual reconstructions directly from monocular human videos. While these methods typically rely on dynamic camera motion, many human video datasets—such as Ego4D [65]—naturally satisfy this condition, offering a viable path toward removing the explicit environment scanning requirement.

Estimating Physics Parameters from Vision Alone. In this work, we use default physics parameters—such as mass, friction, and stiffness—for simulation, rather than estimating them from the human video. However, recent approaches suggest viable paths forward: vision language models (e.g., GPT) can provide plausible physics guesses given object categories or visual context [54], and domain randomization can be applied around these estimates to build robustness. Additionally, while our proposed online sim-to-real calibration targets visual alignment, the same framework could be extended to iteratively adapt physical parameters by comparing real and simulated rollouts—enabling self-supervised refinement of both perception and dynamics.

References

[1] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. R. Sanketi, Q. Vuong, T. Kollar, B. Burchfiel, R. Tedrake, D. Sadigh, S. Levine, P. Liang, and C. Finn. Openvla: An open-source vision-language-action model. *ArXiv*, abs/2406.09246, 2024.

[2] P. Intelligence, K. Black, N. Brown, J. Darpinian, K. Dhabalia, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, M. Y. Galliker, D. Ghosh, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, D. LeBlanc, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, A. Z. Ren, L. X. Shi, L. Smith, J. T. Springenberg, K. Stachowicz, J. Tanner, Q. Vuong, H. R. Walke, A. Walling, H. Wang, L. Yu, and U. Zhilinsky. π 0.5: A vision-language-action model with open-world generalization. 2025.

[3] A. Padalkar, A. Pooley, A. Jain, A. Bewley, A. Herzog, A. Irpan, A. Khazatsky, A. Rai, A. Singh, A. Brohan, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023.

[4] A. Khazatsky, K. Pertsch, S. Nair, A. Balakrishna, S. Dasari, S. Karamcheti, S. Nasiriany, M. K. Srirama, L. Y. Chen, K. Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

[5] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. *arXiv preprint arXiv:2303.04137*, 2023.

[6] T. Zhao, V. Kumar, S. Levine, and C. Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *ArXiv*, abs/2304.13705, 2023.

[7] H. Bharadhwaj, A. Gupta, S. Tulsiani, and V. Kumar. Zero-shot robot manipulation from passive human videos. volume abs/2302.02011, 2023.

[8] J. Ren, P. Sundaresan, D. Sadigh, S. Choudhury, and J. Bohg. Motion tracks: A unified representation for human-robot transfer in few-shot imitation learning. *ArXiv*, abs/2501.06994, 2025.

[9] M. Lepert, R. Doshi, and J. Bohg. Shadow: Leveraging segmentation masks for cross-embodiment policy transfer. *ArXiv*, abs/2503.00774, 2025.

[10] M. Lepert, J. Fang, and J. Bohg. Phantom: Training robots without robots using only human videos. *ArXiv*, abs/2503.00779, 2025.

[11] E. Jang, A. Irpan, M. Khansari, D. Kappler, F. Ebert, C. Lynch, S. Levine, and C. Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. volume abs/2202.02005, 2022.

[12] V. Jain, M. Attarian, N. Joshi, A. Wahid, D. Driess, Q. Vuong, P. R. Sanketi, P. Sermanet, S. Welker, C. Chan, I. Gilitschenski, Y. Bisk, and D. Dwibedi. Vid2robot: End-to-end video-conditioned policy learning with cross-attention transformers. volume abs/2403.12943, 2024.

[13] C. Wang, L. Fan, J. Sun, R. Zhang, L. Fei-Fei, D. Xu, Y. Zhu, and A. Anandkumar. Mimicplay: Long-horizon imitation learning by watching human play. 2023.

[14] B. Huang, Z. Yu, A. Chen, A. Geiger, and S. Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *International Conference on Computer Graphics and Interactive Techniques*, 2024.

[15] B. Wen, W. Yang, J. Kautz, and S. T. Birchfield. Foundationpose: Unified 6d pose estimation and tracking of novel objects. *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 17868–17879, 2023.

[16] A. G. Billard, S. Calinon, and F. Guenter. Discriminative and adaptive imitation in uni-manual and bi-manual tasks. *Robotics Auton. Syst.*, 54:370–384, 2006.

[17] A. Iyer, Z. Peng, Y. Dai, I. Güzey, S. Haldar, S. Chintala, and L. Pinto. Open teach: A versatile teleoperation system for robotic manipulation. *ArXiv*, abs/2403.07870, 2024.

[18] S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *J. Mach. Learn. Res.*, 17:39:1–39:40, 2015.

[19] P. Wu, Y. Shentu, Z. Yi, X. Lin, and P. Abbeel. Gello: A general, low-cost, and intuitive teleoperation framework for robot manipulators. *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 12156–12163, 2023.

[20] T. Z. Zhao, J. Tompson, D. Driess, P. Florence, K. Ghasemipour, C. Finn, and A. Wahid. Aloha unleashed: A simple recipe for robot dexterity. In *Conference on Robot Learning*, 2024.

[21] 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. *ArXiv*, abs/2402.10329, 2024.

[22] A. Sivakumar, K. Shaw, and D. Pathak. Robotic telekinesis: Learning a robotic hand imitator by watching humans on youtube. 2022.

[23] K. Shaw, S. Bahl, and D. Pathak. Videodex: Learning dexterity from internet videos. In *Conference on Robot Learning*, 2022.

[24] S. P. Arunachalam, S. Silwal, B. Evans, and L. Pinto. Dexterous imitation made easy: A learning-based framework for efficient dexterous manipulation. 2022.

[25] J. Ye, J. Wang, B. Huang, Y. Qin, and X. Wang. Learning continuous grasping function with a dexterous hand from human demonstrations. volume 8, pages 2882–2889, 2022.

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

[27] Y. Zhu, A. Lim, P. Stone, and Y. Zhu. Vision-based manipulation from single human video with open-world object graphs. *arXiv preprint arXiv:2405.20321*, 2024.

[28] P. Vitiello, K. Dreczkowski, and E. Johns. One-shot imitation learning: A pose estimation perspective. In *Conference on Robot Learning*, 2023.

[29] J. Li, Y. Zhu, Y. Xie, Z. Jiang, M. Seo, G. Pavlakos, and Y. Zhu. Okami: Teaching humanoid robots manipulation skills through single video imitation. *ArXiv*, abs/2410.11792, 2024.

[30] H. Bharadhwaj, R. Mottaghi, A. Gupta, and S. Tulsiani. Track2act: Predicting point tracks from internet videos enables diverse zero-shot robot manipulation. *arXiv preprint arXiv:2405.01527*, 2024.

[31] H. Bharadhwaj, A. Gupta, V. Kumar, and S. Tulsiani. Towards generalizable zero-shot manipulation via translating human interaction plans. *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6904–6911, 2023. URL https://api.semanticscholar.org/CorpusID:265551754.

[32] H. Bharadhwaj, D. Dwibedi, A. Gupta, S. Tulsiani, C. Doersch, T. Xiao, D. Shah, F. Xia, D. Sadigh, and S. Kirmani. Gen2act: Human video generation in novel scenarios enables generalizable robot manipulation. *ArXiv*, abs/2409.16283, 2024.

[33] M. Xu, Z. Xu, C. Chi, M. Veloso, and S. Song. XSkill: Cross embodiment skill discovery. In *7th Annual Conference on Robot Learning*, 2023. URL https://openreview.net/forum?id=8L6pHd9aS6w.

[34] K. Zakka, A. Zeng, P. R. Florence, J. Tompson, J. Bohg, and D. Dwibedi. Xirl: Cross-embodiment inverse reinforcement learning. In *Conference on Robot Learning*, 2021.

[35] L. Shao, T. Migimatsu, Q. Zhang, K. Yang, and J. Bohg. Concept2robot: Learning manipulation concepts from instructions and human demonstrations. volume 40, pages 1419 – 1434, 2020.

[36] A. Patel, A. Wang, I. Radosavovic, and J. Malik. Learning to imitate object interactions from internet videos. *ArXiv*, abs/2211.13225, 2022.

[37] S. Kumar, J. Zamora, N. Hansen, R. Jangir, and X. Wang. Graph inverse reinforcement learning from diverse videos. 2022.

[38] I. Güzey, Y. Dai, G. Savva, R. M. Bhirangi, and L. Pinto. Bridging the human to robot dexterity gap through object-oriented rewards. *ArXiv*, abs/2410.23289, 2024.

[39] M. M. L. Torné, 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*, abs/2403.03949, 2024.

[40] L. L. Ankile, A. Simeonov, I. Shenfeld, M. M. L. Torné, and P. Agrawal. From imitation to refinement – residual rl for precise assembly. 2024.

[41] S. Patel, X. Yin, W. Huang, S. Garg, H. Nayyeri, F.-F. Li, S. Lazebnik, and Y. Li. A real-to-sim-to-real approach to robotic manipulation with vlm-generated iterative keypoint rewards. *ArXiv*, abs/2502.08643, 2025.

[42] J. Kerr, C. M. Kim, M. Wu, B. Yi, Q. Wang, K. Goldberg, and A. Kanazawa. Robot see robot do: Imitating articulated object manipulation with monocular 4d reconstruction. In *Conference on Robot Learning*, 2024. URL https://api.semanticscholar.org/CorpusID:272910721.

[43] W. Ye, F. Liu, Z. Ding, Y. Gao, O. Rybkin, and P. Abbeel. Video2policy: Scaling up manipulation tasks in simulation through internet videos. *ArXiv*, abs/2502.09886, 2025.

[44] T. Ga, W. Lum, O. Y. Lee, C. K. Liu, J. Bohg, and P.-M. H. Pose. Crossing the human-robot embodiment gap with sim-to-real rl using one human demonstration. 2025.

[45] Polycam. Polycam, 2020. URL https://poly.cam. Accessed: 2025-04-30.

[46] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. B. Girshick. Segment anything. *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3992–4003, 2023.

[47] T. Mu, Z. Ling, F. Xiang, D. Yang, X. Li, S. Tao, Z. Huang, Z. Jia, and H. Su. Maniskill: Generalizable manipulation skill benchmark with large-scale demonstrations. In *NeurIPS Datasets and Benchmarks*, 2021.

[48] V. Lim, H. Huang, L. Y. Chen, J. Wang, J. Ichnowski, D. Seita, M. Laskey, and K. Goldberg. Planar robot casting with real2sim2real self-supervised learning. *arXiv preprint arXiv:2111.04814*, 2021.

[49] P. Chang and T. Padif. Sim2real2sim: Bridging the gap between simulation and real-world in flexible object manipulation. In *2020 Fourth IEEE International Conference on Robotic Computing (IRC)*, pages 56–62. IEEE, 2020.

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

[51] 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)*, pages 3803–3810. IEEE, 2018.

[52] 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)*, pages 23–30. IEEE, 2017.

[53] C.-C. Hsu, Z. Jiang, and Y. Zhu. Ditto in the house: Building articulation models of indoor scenes through interactive perception. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3933–3939. IEEE, 2023.

[54] H. Xia, E. Su, M. Memmel, A. Jain, R. Yu, N. Mbiziwo-Tiapo, A. Farhadi, A. Gupta, S. Wang, and W.-C. Ma. Drawer: Digital reconstruction and articulation with environment realism. *arXiv preprint arXiv:2504.15278*, 2025.

[55] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *ArXiv*, abs/1707.06347, 2017.

[56] A. van den Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *ArXiv*, abs/1807.03748, 2018.

[57] G. Pavlakos, D. Shan, I. Radosavovic, A. Kanazawa, D. F. Fouhey, and J. Malik. Reconstructing hands in 3d with transformers. *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9826–9836, 2023.

[58] Nvidia, J. Bjorck, F. Castaneda, N. Cherniadev, X. Da, R. Ding, LinxiJimFan, Y. Fang, D. Fox, F. Hu, S. Huang, J. Jang, Z. Jiang, J. Kautz, K. Kundalia, L. Lao, Z. Li, Z. Lin, K. Lin, G. Liu, E. Llontop, L. Magne, A. Mandlekar, A. Narayan, S. Nasiriany, S. Reed, Y. L. Tan, G. Wang, Z. Wang, J. Wang, Q. Wang, J. Xiang, Y. Xie, Y. Xu, Z.-T. Xu, S. Ye, Z. Yu, A. Zhang, H. Zhang, Y. Zhao, R. Zheng, and Y. Zhu. Gr00t n1: An open foundation model for generalist humanoid robots. *ArXiv*, abs/2503.14734, 2025.

[59] J. Xu, W. Cheng, Y. Gao, X. Wang, S. Gao, and Y. Shan. Instantmesh: Efficient 3d mesh generation from a single image with sparse-view large reconstruction models. *ArXiv*, abs/2404.07191, 2024. URL https://api.semanticscholar.org/CorpusID:269033473.

[60] C. Li, R. Zhang, J. Wong, C. Gokmen, S. Srivastava, R. Martín-Martín, C. Wang, G. Levine, M. Lingelbach, J. Sun, M. Anvari, M. Hwang, M. Sharma, A. Aydin, D. Bansal, S. Hunter, K.-Y. Kim, A. Lou, C. R. Matthews, I. Villa-Renteria, J. H. Tang, C. Tang, F. Xia, S. Savarese, H. Gweon, C. K. Liu, J. Wu, and L. Fei-Fei. Behavior-1k: A benchmark for embodied ai with 1, 000 everyday activities and realistic simulation. In *Conference on Robot Learning*, 2022.

[61] T. Dai, J. Wong, Y. Jiang, C. Wang, C. Gokmen, R. Zhang, J. Wu, and F.-F. Li. Automated creation of digital cousins for robust policy learning. *ArXiv*, abs/2410.07408, 2024.

[62] H. Xia, E. Su, M. Memmel, A. Jain, R. Yu, N. Mbiziwo-Tiapo, A. Farhadi, A. Gupta, S. Wang, and W.-C. Ma. Drawer: Digital reconstruction and articulation with environment realism. 2025.

[63] J. Abou-Chakra, K. Rana, F. Dayoub, and N. Sünderhauf. Physically embodied gaussian splatting: A visually learnt and physically grounded 3d representation for robotics. In *Conference on Robot Learning*.

[64] H. Feng, J. Zhang, Q. Wang, Y. Ye, P. Yu, M. J. Black, T. Darrell, and A. Kanazawa. St4rtrack: Simultaneous 4d reconstruction and tracking in the world. 2025. URL https://api.semanticscholar.org/CorpusID:277857146.

[65] K. Grauman, A. Westbury, E. Byrne, Z. Q. Chavis, A. Furnari, R. Girdhar, J. Hamburger, H. Jiang, M. Liu, X. Liu, M. Martin, T. Nagarajan, I. Radosavovic, S. K. Ramakrishnan, F. Ryan, J. Sharma, M. Wray, M. Xu, E. Z. Xu, C. Zhao, S. Bansal, D. Batra, V. Cartillier, S. Crane, T. Do, M. Doulaty, A. Erapalli, C. Feichtenhofer, A. Fragomeni, Q. Fu, C. Fuegen, A. Gebreselasie, C. González, J. M. Hillis, X. Huang, Y. Huang, W. Jia, W. Khoo, J. Kolár, S. Kottur, A. Kumar, F. Landini, C. Li, Y. Li, Z. Li, K. Mangalam, R. Modhugu, J. Munro, T. Murrell, T. Nishiyasu, W. Price, P. R. Puentes, M. Ramazanova, L. Sari, K. K. Somasundaram, A. Southerland, Y. Sugano, R. Tao, M. Vo, Y. Wang, X. Wu, T. Yagi, Y. Zhu, P. Arbeláez, D. J. Crandall, D. Damen, G. M. Farinella, B. Ghanem, V. K. Ithapu, C. V. Jawahar, H. Joo, K. Kitani, H. Li, R. A. Newcombe, A. Oliva, H. S. Park, J. M. Rehg, Y. Sato, J. Shi, M. Z. Shou, A. Torralba, L. Torresani, M. Yan, and J. Malik. Ego4d: Around the world in 3,000 hours of egocentric video. pages 18973–18990, 2021.

7 Appendix

7.1 Task Descriptions

We provide descriptions and visualizations (Fig. 10) of tasks we report results for in Fig. 4.

- `Mustard Place`: Pick up the Mustard bottle from the right side of the kitchen countertop and place it on the left side.
- `Corn in Basket`: Pick up the corn from the left side of the kitchen countertop and put it inside of the basket.
- `Shoe on Rack`: Pick up the left shoe and place it on top of the shoe rack, next to the right shoe.
- `Letter Arrange`: Move the letter *'I'* next to the letter *'A'* so that they are aligned.
- `Mug Insert`: Insert the mug's handle onto the holder.

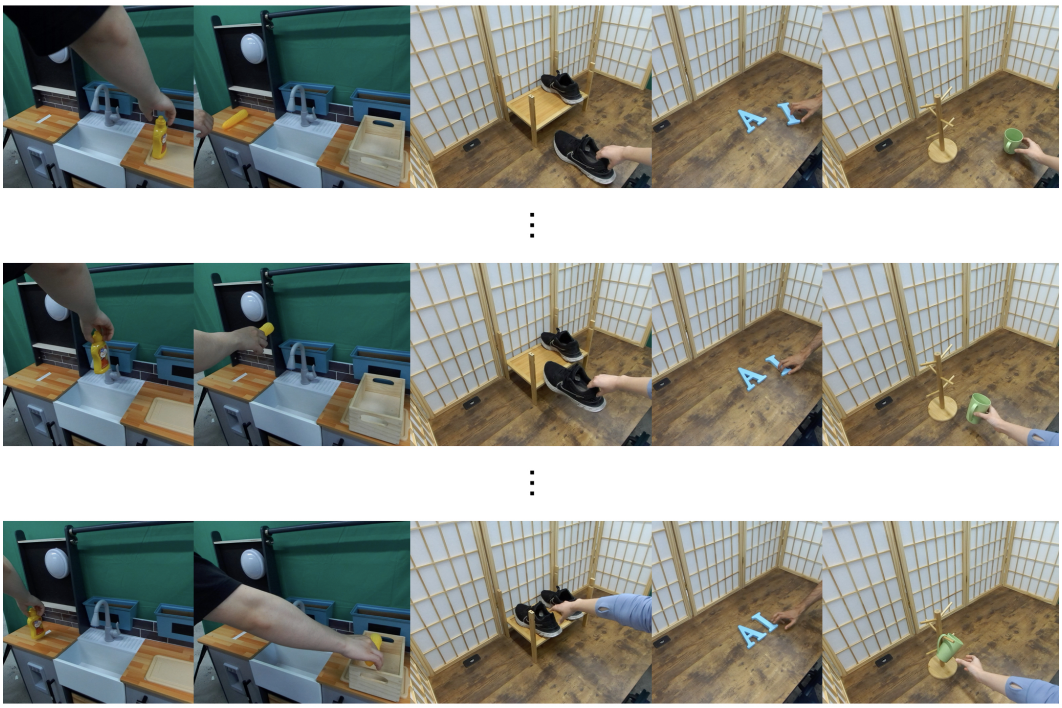

Figure 10: Visualization of tasks that we report results for in Fig. 4

7.2 Real-to-Sim Scans

In order to transfer our environments and objects into simulation, we employ 2D Gaussian Splatting [14]. We take videos (multi-view images) of the environment for < 2 minutes, which are supplied as input to the module to get a photo-realistic 3D reconstruction of the scene. Individual objects to be tracked are scanned with Polycam [45], a phone app, with a similar procedure in < 1 minute per object. The environment and objects are scaled manually to the correct size before being transferred into simulation, though alternate calibration methods could be used to automate this process.

7.3 RL Training Details

7.3.1 PPO Hyperparameters

We provide details of hyperparameters (Table 1) used for training privileged-state PPO [55] policies in simulation.

Table 1: PPO Hyperparameters

Hyperparameter	**Value**
Learning rate	3×10^{-4}
Discount factor (γ)	0.8
GAE parameter (λ)	0.9
Clipping parameter (ϵ)	0.2
Value function coefficient	0.5
Entropy coefficient	0.0
Target KL divergence	0.1
Maximum gradient norm	0.5
Minibatch size	9,600
Number of parallel environments	1,024
Actor network	MLP (state dim $\rightarrow$ 256 $\rightarrow$ 256 $\rightarrow$ 256 $\rightarrow$ action dim)
Critic network	MLP (state dim $\rightarrow$ 256 $\rightarrow$ 256 $\rightarrow$ 256 $\rightarrow$ 1)
Activation function	Tanh
Optimizer	Adam
Adam epsilon	1×10^{-5}

7.3.2 Simulation State Space

The state-based observation space in simulation consists of the following components:

Table 2: Observation Space Components

Component	**Description**
`ee_pose`	End-effector pose (position and orientation)
`gripper_width`	Gripper opening width
`achieved_goal`	Current object poses
`desired_goal`	Target waypoint poses for objects
`goal_position_diff`	Position difference between current and target poses
`goal_rotation_diff`	Angular difference between current and target orientations
`is_grasped`	Binary object grasp status (if applicable)

7.3.3 Reward Function Formulation

We provide the complete object-centric reward function implementation proposed in Sec. 3:

Approach Reward. The r_{approach} component encourages the agent to approach the target object with:

$$r_{\text{approach}} = (1 - \tanh(k d_{\text{obj}})) \tag{3}$$

where d_{obj} is the distance between the end-effector and the current target object and k is a constant scaling factor.

Goal Reward. r_{goal} penalizes positional and rotational deviations from the target state:

$$r_{\text{goal}} = \big(1 - \tanh(\alpha_d \cdot d_{\text{pos}}(s_H^B, s_R^t))\big) + \big(1 - \tanh(\alpha_\theta \cdot d_{\text{rot}}(s_H^B, s_R^t))\big) + 2 i_{\text{waypoint}} \tag{4}$$

where $d_{\text{pos}}(\cdot)$ measures the Euclidean distance, and $d_{\text{rot}}(\cdot)$ computes the quaternion angular difference, α_d and α_θ are scaling factors for each waypoint automatically computed from the demonstration, and i_{waypoint} is the current waypoint index to serve as a bonus for progressing through the task. Note that the `desired_goal` in the observation is updated when the current goal is reached within an ϵ threshold, and in practice we sample N object waypoints from the human video to summarize the demonstration.

The goal reward also has additional terms: r_{static} encourages stability of the robot when objects are correctly positioned, r_{success} provides a $+1$ bonus upon task completion (objects are placed in their goal configuration), and r_{grasp} is an optional binary reward to encourage grasps for non-prehensile tasks.

Complete Reward. The final reward is $r_{\text{obj}} = r_{\text{approach}} + r_{\text{goal}}$.

7.4 Image-Conditioned Policy Training Details

7.4.1 Synthetic Data Collection

We provide details on randomization parameters (Table 3) used when collecting synthetic data for $D_{\text{synthetic}}$ (Sec. 3).

Table 3: Environment Randomization Parameters

Parameter	**Value**
Object Randomization	
Initial pose position noise (XY)	± 0.025 m
Initial pose rotation noise	$\pm \pi/8$ rad
Robot Randomization	
Initial robot joint angle noise	± 0.02 rad
Camera and Lighting (Evaluation)	
Camera position variation	± 0.03 m
Camera target position variation	± 0.03 m
Lighting configurations	4 presets

For each task, we collect 500 visuomotor demonstrations in simulation by applying these randomization parameters.

7.4.2 Image-Conditioned Diffusion Policy Training

We provide hyperparameters (Table 4) for training Diffusion Policies [5], where input is simply an image of the current state and output is 7-dimensional delta actions in end-effector space (3 position actions, 3 rotation actions, 1 gripper action).

7.5 Ablation Visualizations

7.5.1 Data Efficiency

We provide visualizations (Fig. 12) of the initial state distribution of the Mustard bottle as training input for the data efficiency ablation in Sec. 4.3. The robot teleoperation data takes 10 minutes to collect, while the human videos take 1 minute to collect. In simulation, the starting poses of the object are perturbed to enable robustness of the RL policy and diversity in during synthetic data collection. The evaluation distribution is across the cutting board.

7.5.2 Viewpoint Changes

We provide visualizations (Fig. 12) of the three different viewpoints that we study at train/test time in Sec. 4.4.

Table 4: Diffusion Policy Training Hyperparameters

Parameter	**Value**
Diffusion timesteps (training)	100
Diffusion timesteps (inference)	10
Backbone CNN	ResNet18
Image size	$960 \times 720 \rightarrow 96 \times 96$
Image feature dimension	512
Diffusion step embedding dimension	128
Kernel size	5
Normalization layer	GNN
Action horizon	2
Prediction horizon	8
Shift padding	6
Batch size	64
Learning rate	1×10^{-4}
Weight decay	1×10^{-6}
Gradient clipping	5.0
EMA decay rate	0.01
Action prediction loss weight	1
Auto-calibration loss weight	0.1 (if applicable)

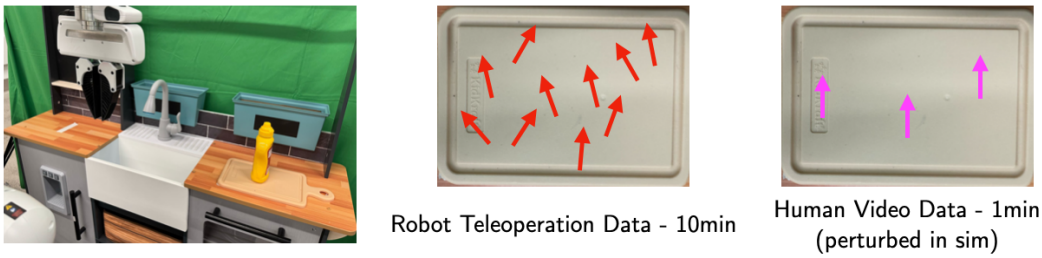

Figure 11: Visualization of training states for results in Fig. 8

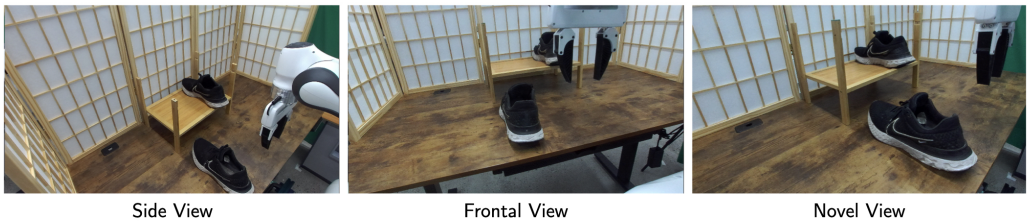

Figure 12: Visualization of viewpoints for results in Fig. 9