CyberDemo: Augmenting Simulated Human Demonstration for Real-World Dexterous Manipulation

Jun Wang[1*] Yuzhe Qin[1*] Kaiming Kuang[1] Yigit Korkmaz[2] Akhilan Gurumoorthy[1]
Hao Su[1] Xiaolong Wang[1]
[1]UC San Diego [2]University of Southern California

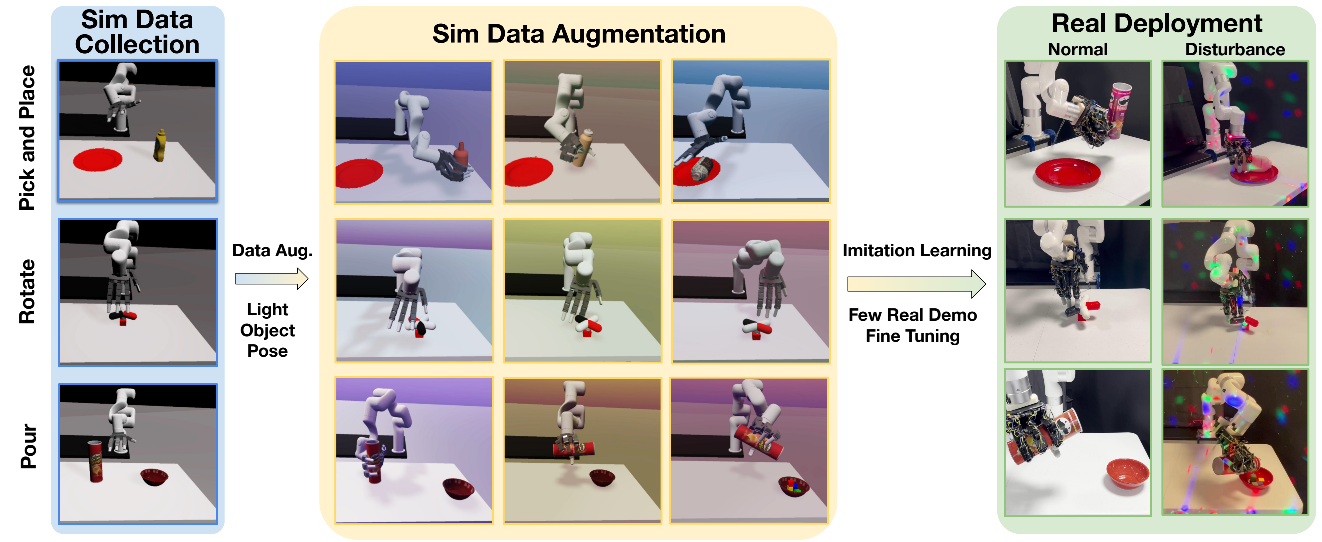


Figure 1. We propose CyberDemo, a novel pipeline for learning real-world dexterous manipulation by using simulation data. First, we collect human demos in a simulated environment (blue region), followed by extensive data augmentation within the simulator (yellow region). Then, the imitation learning model, trained on augmented data and fine-tuned on a few real data, can be deployed on a real robot.

Abstract

*We introduce **CyberDemo**, a novel approach to robotic imitation learning that leverages simulated human demonstrations for real-world tasks. By incorporating extensive data augmentation in a simulated environment, CyberDemo outperforms traditional in-domain real-world demonstrations when transferred to the real world, handling diverse physical and visual conditions. Regardless of its affordability and convenience in data collection, CyberDemo outperforms baseline methods in terms of success rates across various tasks and exhibits generalizability with previously unseen objects. For example, it can rotate novel tetra-valve and penta-valve, despite human demonstrations only involving tri-valves. Our research demonstrates the significant potential of simulated human demonstrations for real-world dexterous manipulation tasks. More details can be found at https://cyber-demo.github.io/*

[*]Equal contributions.

1 . Introduction

Imitation learning has been a promising approach in robot manipulation, facilitating the acquisition of complex skills from human demonstration. However, the effectiveness of this approach is critically dependent on the availability of high-quality demonstration data, which often necessitates substantial human effort for data collection [6, 9, 48]. This challenge is further amplified in the context of manipulation with a multi-finger dexterous hand, where the complexity and intricacy of the tasks require highly detailed and precise demonstrations.

In imitation learning, in-domain demonstrations, which refer to the data collected directly from the deployment environment, are commonly used for robot manipulation tasks [43]. It is generally believed that the most effective way to solve a specific task is to collect demonstrations directly from the real robot on that task. This belief has been upheld as the gold standard, but we wish to challenge it. We argue that collecting human demonstrations in simulation can yield superior results for real-world tasks, not only because it does not require real hardware and can be executed remotely and in parallel, but also due to its potential to enhance final task performance by employing simulator-only data augmentation [34, 37, 42, 44, 53, 58]. This allows the generation of a dataset that is hundreds of times larger than the initial demonstration set. However, while existing studies employ the generated dataset to train in-domain policies within the simulation, the sim2real challenge of transferring policies to the real world remains an unresolved problem.

In this paper, we study the problem of how to utilize simulated human demos for real-world robot manipulation tasks. We introduce **CyberDemo**, a novel framework designed for robotic imitation learning from visual observations, leveraging simulated human demos. We first collect a modest amount of human demonstration data via teleoperation using low-cost devices in a simulated environment. Then, CyberDemo incorporates extensive data augmentation into the original human demonstration. The augmented set covers a broad spectrum of visual and physical conditions not encountered during data collection, thereby enhancing the robustness of the trained policy against these variations. These augmentation techniques are also designed with the downstream sim2real transfer in mind. We employ a unique curriculum learning strategy to train the policy on the augmented dataset, then fine-tune it using a few real-world demos (3-minute trajectories), facilitating effective transfer to real-world conditions. While policies trained on only real-world demonstrations may suffer from variations in lighting conditions, object geometry, and object initial pose, our policy is capable of handling these without the need for additional human effort.

Our system, which utilizes a low-cost motion capture device for teleoperation (i.e., RealSense camera) and demands minimal human effort (i.e., a 30-minute demo trajectory), can learn a robust imitation learning policy. Despite its affordability and minimal human effort requirements, CyberDemo can still achieve better performance on the real robot. Compared with pre-trained policies, e.g. R3M [46] fine-tuned on real-world demonstrations, CyberDemo achieves a success rate that is 35% higher for quasi-static *pick and place* tasks, and 20% higher for non-quasi-static *rotate* tasks. In the generalization test, while baseline methods struggle to handle unseen objects during testing, our method can rotate novel tetra-valve and penta-valve with 42.5% success rate, even though human demonstrations only cover tri-valve (second row of Figure 1). Our method can also manage significant light disturbances (last column of Figure 1). In our ablation study, we observe that the use of data augmentation, coupled with an increased number of demonstrations in the simulator, results in superior performance compared to an equivalent increase in real-world demonstrations. To foster further research, we will make our code and human demonstration dataset publicly available.

2 . Related Work

Data for Learning Robot Manipulation. Imitation learning has been proven to be an effective approach to robotic manipulation, enabling policy training with a collection of demonstrations. Many works have focused on building large datasets using pre-programmed policies [17, 23, 31, 33, 84], alternative data sources such as language [30, 66, 67, 69] and human video [5, 47, 54, 63, 64] or extensive real-world robot teleoperation [2, 3, 6, 9, 20, 32, 39, 43, 48]. However, such works predominantly targeted parallel grippers. Collecting large-scale demonstration datasets for high-DoF dexterous hands continues to be a significant challenge. Meanwhile, data augmentation presents a viable strategy to improve policy generalization by increasing the diversity of data distribution. Previous studies have applied augmentation in low-level visual space [16, 28, 56, 65], such as color jitter, blurring, and cropping, while more recent works propose semantic-aware data augmentation with generative models [7, 14, 15, 40, 78, 86]. However, these augmentations operate at the image level and are not grounded in physical reality. CyberDemo extends data augmentation to the trajectory level using a physical simulator, accounting for both visual and physical variations. Concurrent to our work, MimicGen [44] proposes a system to synthesize demonstrations for long-horizon tasks by integrating multiple human trajectories. However, it confines demonstrations to in-domain learning, i.e., it only trains simulation policies with simulated demos without transferring to real robots. In contrast, our work aims to harness simulation for real-world problem-solving. We exploit the convenience of simulators for collecting robot demonstrations and employ a sim2real approach to transfer these demos to a dexterous robot equipped with a multi-finger humanoid hand. Our research emphasizes a general framework that leverages simulated demonstrations for real-world robot manipulation.

Pre-trained Visual Representation for Robotics Recent progress in large-scale Self-Supervised Learning [10, 26, 27] has enabled the development of visual representations that are advantageous for downstream robotic tasks [62, 80, 83]. Several studies have focused on pretraining on non-robotic datasets, such as ImageNet [18] and Ego4D [22], and utilizing the static representations for downstream robot control [46, 49, 75]. Other research has focused on pretraining visual representations on robot datasets, using action-supervised self-learning objectives that depend on actions [60, 64], or utilizing the temporal consistency of video as a learning objective [59, 61, 70, 76]. These investigations primarily aimed to learn features for effective training of vision-based robotic manipulation. In addition to training visual representations on offline datasets, some

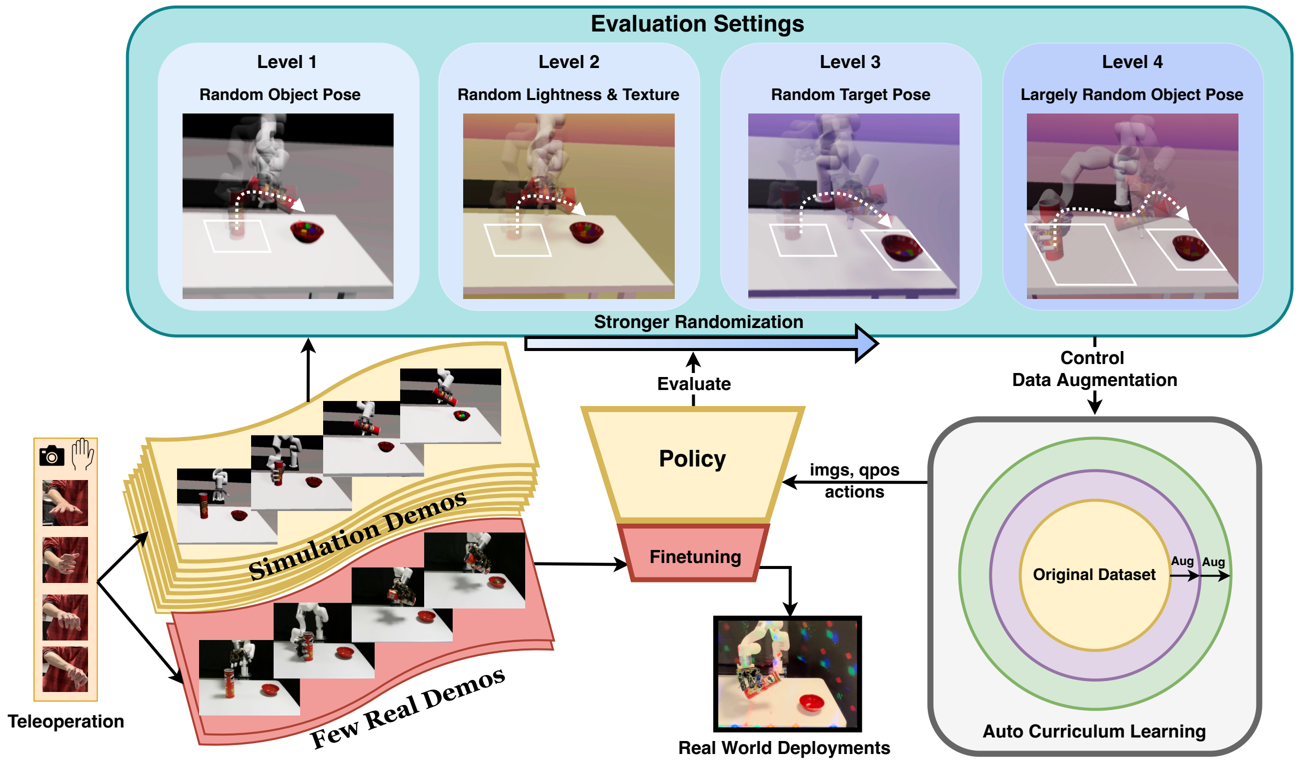


Figure 2. **CyberDemo Pipeline.** First, we collect both simulated and real demonstrations via vision-based teleoperation. Following this, we train the policy on simulated data, incorporating the proposed data augmentation techniques. During training, we apply automatic curriculum learning, which incrementally enhances the randomness scale based on task performance. Finally, the policy is fine-tuned with a few real demos before being deployed to the real world.

researchers have also explored learning the reward function to be used in reinforcement learning [4, 36, 41, 45, 82]. Unlike prior studies, our work diverges by utilizing simulation data for pre-training rather than employing Self-Supervised Learning for representation learning. This not only enhances the learning of image representations but also incorporates task priors into the neural network through the use of action information. By pre-training in simulated environments, the manipulation policy can better generalize to new objects with novel geometries and contact patterns.

Sim2Real Transfer The challenge of transferring skills from simulation to real-world scenarios, known as sim2real transfer, has been a key focus in robot learning. Some approaches have employed system identification to build a mathematical model of real systems and identify physical parameters [12, 29, 35, 38, 52, 73]. Instead of calibrating real-world dynamics, domain randomization [50, 71] generates simulated environments with randomized properties and trains a model function across all of them. Subsequent research demonstrated that the selection of randomization parameters could be automated [1, 11, 24, 81]. However, due to the extensive sample requirements to learn robust policies, domain randomization is typically used with RL involving millions of interaction samples. Domain adaptation (DA) refers to a set of transfer learning strategies developed to align the data distribution between sim and real. Common techniques include domain adversarial training [21, 72] and the use of generative models to make simulated images resemble real ones [8, 28]. Most of these DA approaches focus on bridging the visual gap. However, the challenge of addressing the dynamics gap remains significant. The sim2real gap becomes even more pronounced for dexterous robotic hands that have high-DoF actuation and complex interaction patterns [24, 51, 77, 79]. In this work, we extend the concept of domain randomization to human demonstration collected in the simulator and focus on data augmentation techniques that can effectively utilize the simulation for transfer to a real robot. We demonstrate that there can be a significant benefit in collecting human demonstration in the simulator, despite the sim2real gap, instead of solely relying on real data.

3 . CyberDemo

In CyberDemo, we initially gather human demonstrations of the same task in a simulator through teleoperation (Section 3 .1). Taking advantage of the simulator's sampling capabilities and oracle state information, we enhance the simulated demonstration in various ways, increasing its vi-

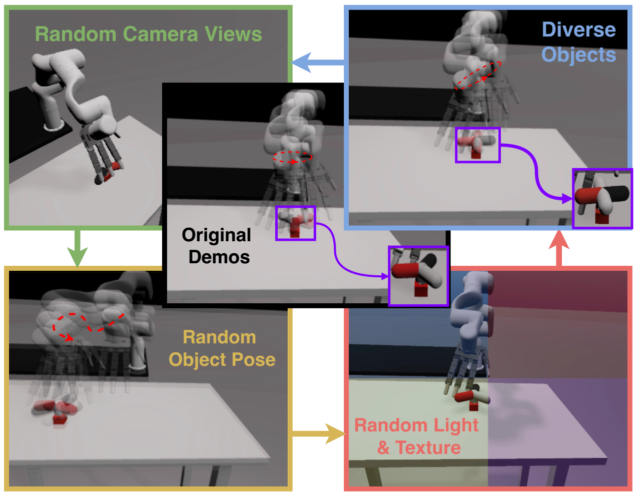


Figure 3. **Data Augmentation**. Our dataset augmentation encompasses four dimensions: (a) random camera views, (b) diverse objects, (c) random object pose, (d) random light and texture.

sual, kinematic, and geometric diversity, thereby enriching the simulated dataset (Section 3 .2). With this augmented dataset, we train a manipulation policy with Automatic Curriculum Learning and Action Aggregation (Section 3 .3).

3 .1. Collecting Human Teleoperation Data

For each dexterous manipulation task in this work, we collect human demonstrations using teleoperation in both simulated and real-world environments. For real-world data, we utilize the low-cost teleoperation system referenced in [55]. This vision-based teleoperation system solely needs a camera to capture human hand motions as input, which are then translated into real-time motor commands for the robot arm and the dexterous hand. We record the observation (RGB image, robot proprioception) and the action (6D Cartesian velocity of robot end effector, finger joint position control target) for each frame at a rate of 30Hz. For this work, we collect only three minutes of robot trajectories for each task on the real robot.

For data in simulation, we build the real-world task environments within the SAPIEN [74] simulator to replicate the tables and objects used in real scenarios. It is worth noting that, for teleoperation, there is no requirement of reward design and observation spaces as in reinforcement learning settings, making the process of setting up new tasks in the simulator relatively simple. We employ the same teleoperation system [55] to collect human demonstrations in the simulator.

3 .2. Augmenting Human Demo in Simulator

Unlike real-world data collection, where we are limited to recording observations of physical sensors, such as camera RGB images and robot proprioception, the simulation system enables us to record the ground-truth state and contact information within the virtual environment. This unique benefit of simulation provides a more comprehensive data format for the simulated demonstrations compared to its real-world counterparts. Thus, we can take advantage of demonstration replay techniques on these simulated demonstrations, which are not feasible with real-world data.

When developing data augmentation techniques in the simulator, it is essential to keep in mind that the ultimate goal is to deploy the trained policy to a real robot. The augmentation should accordingly focus on the visual and dynamical variations that are likely to be encountered in the real world. Moreover, we aim for the manipulation policy to generalize to novel objects not encountered during the data collection process. For example, manipulating the tetra-valve when collecting data only on the tri-valve in Figure 3. Specifically, we chose to augment the lighting conditions, camera views, and object textures to enhance the policy's robustness against visual variations. In addition, we modified the geometric shape of the objects and the initial poses of the robot and objects to improve the policy's robustness against dynamical variations as follows:

Randomize Camera Views. Precisely aligning camera views between demo collection and final evaluation, not to mention between simulation and reality, poses a significant challenge. To solve this problem, we randomize the camera pose during training and replay the internal state of the simulator to render image sequences from new camera views. Unlike standard image augmentation techniques such as cropping and shifting, our method respects the perspective projection in a physically realistic manner.

Random Light and Texture. To facilitate sim2real transfer and improve the policy's robustness against visual variations, we randomize the visual properties of both lights and objects (Figure 3, lower right). Light properties include directions, colors, shadow characteristics, and ambient illumination. Object properties include specularity, roughness, metallicity, and texture. Similar to camera view randomization, we can simply replay the simulation state to render new image sequences.

Add Diverse Objects. In this approach, we replace the manipulated object in the original demos with novel objects (Figure 3 upper right). However, directly replaying the same trajectory would not work as the object shape is different. Instead, we perturb the action sequence from the original demo with Gaussian noises to generate new trajectories. These trajectories provide reasonable manipulation strategies but are slightly different from the original one. With the highly cost-effective sampling in the simulator, we can enumerate the perturbation until it is successful. It is important to note that this technique is feasible with real-world demonstrations.

Algorithm 1 Auto Curriculum Learning

```
Input: human demo $D_h$, training set $D$, policy network $\pi$, curriculum level $L$, evaluation function $\mathbf{eval}_L()$,
    data aug. function $\mathbf{aug}_L()$, success rate threshold $r_{up}$, number of failure $N_{fail}$, max number of failure $N_{max}$
Output: Trained policy $\pi$
 1: Initialize $\pi$ and set $L = 0$, $N_{fail} = 0$, $D = \{\}$
 2: while $L \leq 4$ do
 3:     Generate augmented data $\mathbf{aug}_L(D_h)$
 4:     Append into training set $D \leftarrow D + \mathbf{aug}_L(D_h)$
 5:     Train $\pi$ on $D$
 6:     Eval success rate $r_{succ} = \mathbf{eval}_L(\pi)$
 7:     if $r_{succ} \geq r_{up}$ or $N_{fail} \geq N_{max}$ then
 8:         $L = L + 1$, $N_{fail} = 0$
 9:     else
10:         Generate more data $D \leftarrow D + \mathbf{aug}_L(D_h)$
11:         $N_{fail} = N_{fail} + 1$
12:     end if
13: end while
```

Randomize Object Pose. A common approach in reinforcement learning to enhance generalizability involves randomizing the object pose during reset. Augmenting imitation learning data to achieve a similar outcome, however, is less intuitive. Denote $T_A^B \in SE(3)$ as the pose of frame B relative to frame A. The original object pose is $T_W^{O_{old}}$, the newly randomized object pose is $T_W^{O_{new}}$, and the original end effector pose is $T_W^{R_{old}}$. The objective is to handle the object pose change $T_W^{O_{new}}(T_W^{O_{old}})^{-1}$. A simple strategy can be first moving the robot end effector to a new initial pose, $T_W^{R_{new}} = T_W^{O_{new}}(T_W^{O_{old}})^{-1}T_W^{R_{old}}$. Then, the relative pose between the robot and the object aligns with the original demonstration, enabling us to replay the same action sequence to accomplish the task. Although this method succeeds in generating new trajectories, it offers minimal assistance for downstream imitation learning. The new trajectory is always composed of two segments: a computed reaching trajectory to the new end effector pose $T_W^{R_{new}}$, and the original trajectory. Given that different augmented trajectories often share a significant portion of redundancy, they fail to provide substantial new information to learning algorithms.

To address this, we propose **Sensitivity-Aware Kinematics Augmentation** to randomize object poses for human demonstrations. Instead of appending a new trajectory ahead of the original one, this method amends the action for each step in the original demo to accommodate the change in object pose $T_W^{O_{new}}(T_W^{O_{old}})^{-1}$. The method includes two steps: (i) Divide the entire trajectory into several segments and compute the sensitivity of each segment; (ii) Modify the end effector pose trajectory based on the sensitivity to compute the new action.

(i) **Sensitivity Analysis for Trajectory Segments.** Sensitivity pertains to the robustness against action noise. For example, a pre-grasp state, when the hand is close to the object, has higher sensitivity compared to a state where the hand is far away. The critical insight is that it is simpler to modify the action of those states with lower sensitivity to handle the object pose variation $\Delta T = T_W^{O_{new}}(T_W^{O_{old}})^{-1}$. The robustness (the multiplicative inverse of sensitivity) of a trajectory segment ψ can be mathematically defined as follows:

$$\begin{aligned} \psi_{seg} = \exp(\max\delta_a) \quad \text{s.t.} \quad \mathbf{eval}(\tau') = 1 \\ \tau' = \{a_1, a_2, ..., a'_n, ..., a'_{n+K-1}, ..., a_N\} \\ \forall i \in seg \quad a'_i = a_i + \delta_a \epsilon_i, \quad \epsilon_i \sim \mathcal{N}(0,1) \end{aligned} \tag{1}$$

In this equation, we divide the original action trajectory τ with length N into M segments, each segment with size $K = N/M$. Then we perturb the action within a segment seg by adding Gaussian noise of scale δ_a to the original action $\{a_m, a_{n+1}, ..., a_{n+k-1}\}$ while keeping all the actions outside of this segment unchanged to generate perturbed trajectory τ'. We assume the action space is already normalized to $[-1, 1]$ and $\mathbf{eval}$ is a binary function indicating whether an action trajectory can successfully solve the task. Intuitively, a demonstration segment is more sensitive if a smaller perturbation can cause it to fail. This sensitivity guides us on how to adjust the action to handle a new object pose. In practice, we incrementally escalate the noise scale δ_a applied to the original action trajectory until the task fails to determine $\max \delta_a$

(ii) **New End Effector Pose Trajectory.** To accommodate the new object pose, the total pose change of the end effector should be the same as the change in the object pose ΔT. Each action contributes a small part to this change. We distribute this "task" to each step based on sensitivity:

$$\begin{aligned} \overline{\psi}_{seg_j} &= \frac{\psi_{seg_j}}{\sum_{j=1}^{M} \psi_{seg_j}}, \quad \forall seg_j \\ \Delta T_j &= exp(\overline{\psi}_{seg_j} \log(\Delta T)/K) \\ a_i^{new} &= a_i f_i(\Delta T_j) \end{aligned} \tag{2}$$

In this equation, $\overline{\psi}_{seg_j}$ is the normalized robustness, ΔT_j represents the pose modification for each step, with all states in the same segment being responsible for the same amount of modification "task" to compute new action a_i^{new}. f_i is a similarity transformation in $SE(3)$ space that converts the motion from the world frame to the current end effector frame. Intuitively, segments with higher robustness are tasked with more significant changes.

Please note that all the actions discussed above pertain solely to the 6D delta pose of the end effector and do not include the finger motion of the dexterous hand. For tasks such as pick-and-place or pouring, which also involve a target pose (e.g., the plate pose in pick-and-place or the bowl pose in pouring), we can apply the same augmentation strategy to the target pose (as illustrated in Level 3 of Fig. 2).

3 .3. Learning Sim2Real Policy

Given an augmented simulation dataset, we train a visual manipulation policy that takes images and robot proprioception as input to predict the robot's actions. In human teleoperation demonstrations, robot movements are neither Moravian nor temporally correlated. To deal with this issue, our policy is trained to predict action chunks rather than per-step actions, using Action Chunking with Transformers(ACT) [85]. This approach produces smoother trajectories and reduces compounding errors.

Despite our data augmentation's capacity to accommodate diverse visual and dynamic conditions, a sim2real gap remains for the robot controller. This gap becomes more challenging in our tasks, where the end effector is a high-DoF multi-finger dexterous hand. This controller gap can significantly impact non-quasi-static tasks like rotating a valve, as shown in the second row of Figure 1. To close this gap, we fine-tune our network using a small set of real-world demonstrations (3-minute trajectory). However, due to the discrepancies in data collection patterns of human demos between simulation and reality, direct fine-tuning on real data risks overfitting. To ensure a smoother sim2real transfer, we employ several techniques, which will be discussed subsequently.

Automatic Curriculum Learning. Curriculum learning and data augmentation techniques are often used together to provide a smoother training process. Following the spirit of curriculum design in previous reinforcement learning work [1, 24], we devise a curriculum learning strategy applicable to our imitation learning context. Prior to training, we group the augmentations in Section 3 .2 into four levels of increasing complexity, as depicted in Figure 2. As per Algorithm 1, we begin training from the simplest level, $L = 0$, signifying no augmentation, and then evaluate the task success rate after several steps of training. The evaluation difficulty aligns with the current level of L. When the success rate surpasses a pre-defined threshold, we advance to the next level, which brings greater augmentation and harder evaluation. If the success rate fails to reach the threshold, we create additional augmented training data and stays at the current level. We continue this iterative process until all levels are completed. To prevent endless training, we introduce a fail-safe N_{max}: if the policy repeatedly fails during evaluation for N_{max} times, we also progress to the next level. This curriculum learning approach significantly depends on data augmentation techniques to generate training data dynamically with suitable levels of randomization. This concept stands in contrast to typical supervised learning scenarios, where data is pre-established prior to training. This on-demand data generation and customization highlights the advantage of simulation data over real-world demonstrations.

Action Aggregation for Small Motion. Human demonstrations often include noise, especially during operations involving a dexterous hand. For example, minor shaking and unintentional halting can occur within the demonstration trajectory, potentially undermining the training process. To solve this, we aggregate steps characterized by small motions, merging these actions into a single action. In practice, we set thresholds for both end-effector and finger motions to discern whether a given motion qualifies as small. Through the aggregation process, we can eliminate small operational noises from human actions, enabling the imitation learning policy to extract meaningful information from the state-action trajectory.

4 . Experiment Setups

Our experimental design aims to address the following key queries:

(i) How does simulation-based data augmentation compare to learning from real demonstrations in terms of both robustness and generalizability?

(ii) How does our automatic curriculum learning contribute to improved policy learning?

(iii) What is the ideal ratio between simulated and real data to train an effective policy for a real-world robot?

4 .1. Dexterous Manipulation Tasks

We have designed three types of manipulation tasks in both real-world and simulated environments, including two quasi-static tasks (pick and place, pour) and one non-quasi-static task (rotate). For the experiments, we utilize an Allegro hand attached to an XArm6. The action space comprises a 6-dim delta end effector pose of the robot arm and a 16-dim finger joint position of the dexterous hand, with PD control employed for both arm and hand.

Pick and Place. This task requires the robot to lift an object from the table and position it on a plate (first row of Figure 1). Success is achieved when the object is properly placed onto the red plate. We select two objects during data collection and testing on multiple different objects.

Rotate. This task requires the robot to rotate a valve on the table (second row of Figure 1). The valve is constructed with a fixed base and a moving valve geometry, connected via a revolute joint. The task is successful when the robot rotates the valve to 720 degrees. We use a tri-valve in data collection and test on tetra-valves and penta-valves.

Pour. This task requires the robot to pour small boxes from a bottle into a bowl (third row of Figure 1). It involves three steps: (i) Lift the bottle; (ii) Move it close to the bowl; (iii) Rotate the bottle to dispense the small boxes into the bowl. Success is achieved when all four boxes have been poured into the bowl.

	Pick and Place Mustard Bottle (Single Object)				Pick and Place Tomato Soup Can (Single Object)				Pouring				Rotating			
	Level 1	Level 2	Level 3	Level 4	Level 1	Level 2	Level 3	Level 4	Level 1	Level 2	Level 3	Level 4	Level 1	Level 2	Level 3	Level 4
R3M	2 / 20	0 / 20	0 / 20	0 / 20	7 / 20	3 / 20	4 / 20	0 / 20	3 / 20	0 / 20	0 / 20	0 / 20	11 / 20	2 / 20	6 / 20	2 / 20
PVR	4 / 20	0 / 20	0 / 20	0 / 20	4 / 20	0 / 20	3 / 20	0 / 20	2 / 20	0 / 20	1 / 20	0 / 20	8 / 20	3 / 20	5 / 20	1 / 20
MVP	2 / 20	0 / 20	3 / 20	1 / 20	7 / 20	2 / 20	4 / 20	2 / 20	1 / 20	1 / 20	3 / 20	2 / 20	8 / 20	4 / 20	10 / 20	6 / 20
Ours	**7 / 20**	**6 / 20**	**8 / 20**	**5 / 20**	**14 / 20**	**11 / 20**	**13 / 20**	**13 / 20**	**9 / 20**	**4 / 20**	**10 / 20**	**7 / 20**	**15 / 20**	**10 / 20**	**17 / 20**	**13 / 20**

Table 1. **Main Comparison on Real Robot.** In our study, we compare the performance across four distinct tasks: (a) Pick and Place Bottle, (b) Pick and Place Can (exploring different grasping approaches), (c) Pouring (grasping a bottle and pouring its contents into a bowl), and (d) Rotating the tri-valve. We perform evaluations of the models in four levels of real-world scenarios. These levels included: (a) Level 1: In Domain, (b) Level 2: Out of Position, (c) Level 3: Random Light, and (d) Level 4: Out of Position and Random Light.

For each task, we have designed levels for both data augmentation (Section 3 .2) and curriculum learning (Section 3 .3). More details regarding the design of task levels can be found in the supplementary material.

4 .2. Baselines

Our approach can be interpreted as an initial pretraining phase using augmented simulation demonstrations followed by fine-tuning with a limited set of real data. It is natural to compare our method with other pre-training models for robotic manipulation. We have chosen three representative vision pre-training models. For all of them, we utilize the pre-trained model provided by the author and then fine-tune it using our real-world demonstration dataset.

PVR is built on MoCo-v2 [13], using a ResNet50 backbone [25] trained on ImageNet [57].

MVP employs self-supervised learning from a Masked Autoencoder [27] to train visual representation on individual frames from an extensive human interaction dataset compiled from multiple existing datasets. MVP integrates a Vision Transformer [19] backbone that segments frames into 16x16 patches.

R3M proposes a pre-training approach where a ResNet50 backbone is trained using a mix of time-contrastive learning, video-language alignment, and L1 regularization. This model is trained on a large-scale human interaction videos dataset from Ego4D [22].

5 . Results

5 .1. Main Comparison

Augmented simulation data markedly boosts real-world dexterous manipulation. As depicted in Table 1, our methodology outperforms the baselines trained exclusively on real data in the in-domain setting (Level 1), exhibiting an average performance boost of 31.67% averaged on all tasks. Additionally, in other settings like random lighting (Level 2), out of position (Level 3), combined random lighting and out of position (Level 3) R3M, and MVP, the baselines exhibit a significant drop in success rates. In contrast, our method shows resilience to these variations, underscoring the efficacy of simulation data augmentation. Not only does this approach bridge the sim2real gap and amplify performance in in-domain real-world tasks, but it also significantly improves manipulations in out-of-domain real-world scenarios. By integrating augmentation for both visual and dynamic variations, our method successfully navigates challenges and delivers impressive results.

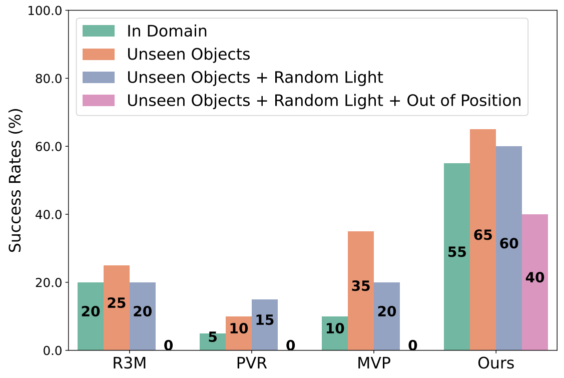


Figure 4. **Generalization to Novel Objects for Pick and Place.** We compare our approach with the baselines in scenarios involving novel objects, random light disturbances, and random object positions.

5 .2. Generalization to Novel Objects

By incorporating data augmentation techniques, such as including diverse objects in simulation, our model can effectively manipulate unfamiliar objects, even when transitioning to a real-world context. As shown in Figure 4 and 5, the baseline methods grapple with more complex real-world situations. In the most challenging scenario, rotating novel objects under random light conditions and new object positions, only one baseline method manages to solve it by chance with a 2.5% success rate. In contrast, our method still accomplishes the task with a success rate of 30%.

5 .3. Ablation on Data Augmentation

To evaluate the effectiveness of the data augmentation techniques, we perform an ablation study where our policy is trained with four levels of augmentation. As depicted in Ta-

Set of Levels / Num of demos	Test in Sim Level 1	Level 2	Level 3	Level 4	Test in Real In Domain	Random Light	Out of Position	Out of Position + Random Light
[1] / 100	78%	0%	0%	0%	20%	5%	0%	0%
[1, 2] / 330	73%	75%	10.5%	7.5%	15%	25%	0%	0%
[1,2,3] / 550	58%	66.5%	43.5%	21%	15%	15%	5%	15%
[1,2,3,4] / 810	92.5%	81%	63%	49%	35%	30%	30%	40%

Table 2. **Ablation on Data Augmentation.** To demonstrate the benefits of data augmentation, we employed auto-curriculum learning on various sets of levels. We performed 200 simulations to test its impact in a simulated environment and conducted 20 real-world tests to evaluate its effectiveness in a practical setting.

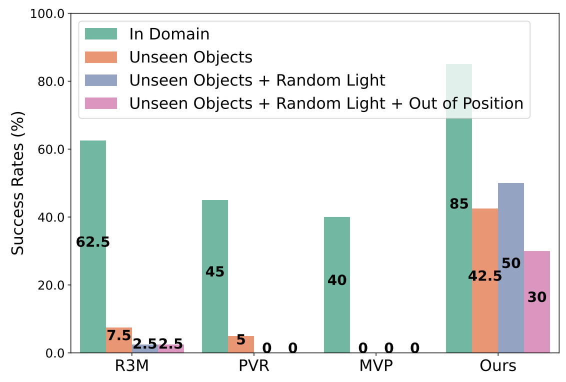


Figure 5. **Generalization to Novel Objects for Rotating.** The experimental setup for this task mirrors that of the "Generalization to Novel Objects for Pick and Place" experiments.

ble 2, the policy performs better in both the simulation and real-world settings with increased data augmentation, and the policy trained on all four levels excels in all metrics. Interestingly, the policy manages to solve simpler settings more effectively even in simulation when more randomness is introduced in the training data. These experiments underscore the importance of simulator-based data augmentation.

5 .4. Ablation on Auto-Curriculum Learning

In this experiment, we evaluate the policy's effectiveness by testing it 200 times in simulations and conducting 20 real-world tests. As shown in Table 3, employing curriculum learning with auto-domain randomization solely based on the data generation rate yields inferior results compared to the approach based on model performance.

5 .5. Ablation on Ratio of Sim and Real Demos

To determine the optimal ratio between sim and real demos, we conducted tests using different combinations of sim and real demonstrations, as shown in Table 4. We observe that training solely on 50 real demonstrations results in poor performance, and the policy overfits to joint positions rather than utilizing the visual information in images. The best results were obtained with a combination of 15 simulation

Method	Test in Sim Level 1	Level 2	Level 3	Level 4	Test in Real Out of Position + Random Light
ACL (Task)	**80%**	**61%**	43.5%	57%	**35%**
ACL (Data)	19.5%	30%	**75%**	**66%**	20%
ACL wo CL(Data)	20%	22%	32.5%	15%	5%

Table 3. **Ablation on Auto-Curriculum Learning**. We compare three different settings: (1) Auto Curriculum Learning based on the success rate. (2) Auto Curriculum Learning based on Data Generation Rate(the ratio of successfully generated trajectories to the total number of attempts). (3) Automatic Domain Randomization only based on Data Generation Rate.

Dataset	Test in Sim Level 1	Level 2	Level 3	Level 4	Test in Real In Domain	Out of Position
50 sim	73.5%	**80%**	**63.5%**	36%	0%	0%
35 sim + 15 real	**77%**	70.5%	61%	**45.5%**	25%	**35%**
15 sim + 35 real	63%	74%	55%	33.5%	**50%**	15%
50 real	0%	0%	0%	0%	10%	0%

Table 4. **Ablation on Ratio of Sim and Real Demos.** We compare the performance resulting from various quantities of simulated and real demonstrations, keeping the total number of demonstrations constant.

demonstrations and 35 real demonstrations. These results highlight that collecting simulation data can be exceptionally valuable, even more so considering the significantly lower data collection costs.

6 . Discussion

We propose CyberDemo, a novel pipeline for imitation learning in robotic manipulation, leveraging demonstrations collected in simulation. While the common belief suggests that real-world demonstrations are the optimal way to solve real-world problems, we challenge this notion by demonstrating that extensive data augmentation can make simulation data even more valuable than real-world demonstrations, a fact also supported by our experiments. One limitation is the necessity to design a simulated environment for each real-world task, thereby increasing the human effort involved. However, since our method doesn't demand the design of specific rewards as in reinforcement learning tasks, which is often the most challenging aspect, the overall effort required is not as significant.

7 . Acknowledgement

We express our gratitude to Haichuan Che and Chengzhe Jia for their insightful conversations regarding the repair of the Allegro hand hardware. We are also thankful to Nicklas Hansen and Jiarui Xu for their valuable input on visual pretraining. Special thanks are due to Tony Zhao for his generous contribution to the community through his open-source projects on Action Chunking with Transformers.

References

[1] Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019. 3, 6

[2] Sridhar Pandian Arunachalam, Irmak Güzey, Soumith Chintala, and Lerrel Pinto. Holo-dex: Teaching dexterity with immersive mixed reality. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5962–5969. IEEE, 2023. 2

[3] Sridhar Pandian Arunachalam, Sneha Silwal, Ben Evans, and Lerrel Pinto. Dexterous imitation made easy: A learning-based framework for efficient dexterous manipulation. In *2023 ieee international conference on robotics and automation (icra)*, pages 5954–5961. IEEE, 2023. 2

[4] Yusuf Aytar, Tobias Pfaff, David Budden, Thomas Paine, Ziyu Wang, and Nando De Freitas. Playing hard exploration games by watching youtube. *Advances in neural information processing systems*, 31, 2018. 3

[5] Homanga Bharadhwaj, Abhinav Gupta, and Shubham Tulsiani. Visual affordance prediction for guiding robot exploration. *arXiv preprint arXiv:2305.17783*, 2023. 2

[6] Homanga Bharadhwaj, Jay Vakil, Mohit Sharma, Abhinav Gupta, Shubham Tulsiani, and Vikash Kumar. Roboagent: Generalization and efficiency in robot manipulation via semantic augmentations and action chunking. *arXiv preprint arXiv:2309.01918*, 2023. 1, 2

[7] Kevin Black, Mitsuhiko Nakamoto, Pranav Atreya, Homer Walke, Chelsea Finn, Aviral Kumar, and Sergey Levine. Zero-shot robotic manipulation with pretrained image-editing diffusion models. *arXiv preprint arXiv:2310.10639*, 2023. 2

[8] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3722–3731, 2017. 3

[9] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022. 1, 2

[10] Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021. 2

[11] Yevgen Chebotar, Ankur Handa, Viktor Makoviychuk, Miles Macklin, Jan Issac, Nathan Ratliff, and Dieter 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. 3

[12] Tao Chen, Megha Tippur, Siyang Wu, Vikash Kumar, Edward Adelson, and Pulkit Agrawal. Visual dexterity: In-hand reorientation of novel and complex object shapes. *Science Robotics*, 8(84):eadc9244, 2023. 3

[13] Xinlei Chen, Haoqi Fan, Ross Girshick, and Kaiming He. Improved baselines with momentum contrastive learning. *arXiv preprint arXiv:2003.04297*, 2020. 7

[14] Zoey Chen, Karl Van Wyk, Yu-Wei Chao, Wei Yang, Arsalan Mousavian, Abhishek Gupta, and Dieter Fox. Learning robust real-world dexterous grasping policies via implicit shape augmentation. *arXiv preprint arXiv:2210.13638*, 2022. 2

[15] Zoey Chen, Sho Kiami, Abhishek Gupta, and Vikash Kumar. Genaug: Retargeting behaviors to unseen situations via generative augmentation. *arXiv preprint arXiv:2302.06671*, 2023. 2

[16] Ekin D Cubuk, Barret Zoph, Dandelion Mane, Vijay Vasudevan, and Quoc V Le. Autoaugment: Learning augmentation policies from data. arxiv 2018. *arXiv preprint arXiv:1805.09501*, 1805. 2

[17] Murtaza Dalal, Ajay Mandlekar, Caelan Garrett, Ankur Handa, Ruslan Salakhutdinov, and Dieter Fox. Imitating task and motion planning with visuomotor transformers. *arXiv preprint arXiv:2305.16309*, 2023. 2

[18] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 2

[19] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 7

[20] Frederik Ebert, Yanlai Yang, Karl Schmeckpeper, Bernadette Bucher, Georgios Georgakis, Kostas Daniilidis, Chelsea Finn, and Sergey Levine. Bridge data: Boosting generalization of robotic skills with cross-domain datasets. *arXiv preprint arXiv:2109.13396*, 2021. 2

[21] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030, 2016. 3

[22] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18995–19012, 2022. 2, 7

[23] Jiayuan Gu, Fanbo Xiang, Xuanlin Li, Zhan Ling, Xiqiang Liu, Tongzhou Mu, Yihe Tang, Stone Tao, Xinyue Wei, Yunchao Yao, et al. Maniskill2: A unified benchmark for generalizable manipulation skills. *arXiv preprint arXiv:2302.04659*, 2023. 2

[24] Ankur Handa, Arthur Allshire, Viktor Makoviychuk, Aleksei Petrenko, Ritvik Singh, Jingzhou Liu, Denys Makoviichuk, Karl Van Wyk, Alexander Zhurkevich, Balakumar Sundaralingam, et al. Dextreme: Transfer of agile in-hand manipulation from simulation to reality. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5977–5984. IEEE, 2023. 3, 6

[25] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 7

[26] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020. 2

[27] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022. 2, 7

[28] Daniel Ho, Kanishka Rao, Zhuo Xu, Eric Jang, Mohi Khansari, and Yunfei Bai. Retinagan: An object-aware approach to sim-to-real transfer. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10920–10926. IEEE, 2021. 2, 3

[29] Binghao Huang, Yuanpei Chen, Tianyu Wang, Yuzhe Qin, Yaodong Yang, Nikolay Atanasov, and Xiaolong Wang. Dynamic handover: Throw and catch with bimanual hands. *arXiv preprint arXiv:2309.05655*, 2023. 3

[30] Wenlong Huang, Chen Wang, Ruohan Zhang, Yunzhu Li, Jiajun Wu, and Li Fei-Fei. Voxposer: Composable 3d value maps for robotic manipulation with language models. *arXiv preprint arXiv:2307.05973*, 2023. 2

[31] Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 5(2):3019–3026, 2020. 2

[32] Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. Bc-z: Zero-shot task generalization with robotic imitation learning. In *Conference on Robot Learning*, pages 991–1002. PMLR, 2022. 2

[33] Yunfan Jiang, Agrim Gupta, Zichen Zhang, Guanzhi Wang, Yongqiang Dou, Yanjun Chen, Li Fei-Fei, Anima Anandkumar, Yuke Zhu, and Linxi Fan. Vima: General robot manipulation with multimodal prompts. *arXiv*, 2022. 2

[34] Amlan Kar, Aayush Prakash, Ming-Yu Liu, Eric Cameracci, Justin Yuan, Matt Rusiniak, David Acuna, Antonio Torralba, and Sanja Fidler. Meta-sim: Learning to generate synthetic datasets. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4551–4560, 2019. 2

[35] Manuel Kaspar, Juan D Muñoz Osorio, and Jürgen Bock. Sim2real transfer for reinforcement learning without dynamics randomization. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4383–4388. IEEE, 2020. 3

[36] Sateesh Kumar, Jonathan Zamora, Nicklas Hansen, Rishabh Jangir, and Xiaolong Wang. Graph inverse reinforcement learning from diverse videos. In *Conference on Robot Learning*, pages 55–66. PMLR, 2023. 3

[37] Chengshu Li, Ruohan Zhang, Josiah Wong, Cem Gokmen, Sanjana Srivastava, Roberto Martín-Martín, Chen Wang, Gabrael Levine, Michael Lingelbach, Jiankai Sun, et al. Behavior-1k: A benchmark for embodied ai with 1,000 everyday activities and realistic simulation. In *Conference on Robot Learning*, pages 80–93. PMLR, 2023. 2

[38] Jacky Liang, Saumya Saxena, and Oliver Kroemer. Learning active task-oriented exploration policies for bridging the sim-to-real gap. *arXiv preprint arXiv:2006.01952*, 2020. 3

[39] Corey Lynch, Ayzaan Wahid, Jonathan Tompson, Tianli Ding, James Betker, Robert Baruch, Travis Armstrong, and Pete Florence. Interactive language: Talking to robots in real time. *IEEE Robotics and Automation Letters*, 2023. 2

[40] Zhao Mandi, Homanga Bharadhwaj, Vincent Moens, Shuran Song, Aravind Rajeswaran, and Vikash Kumar. Cacti: A framework for scalable multi-task multi-scene visual imitation learning. *arXiv preprint arXiv:2212.05711*, 2022. 2

[41] Priyanka Mandikal and Kristen Grauman. Dexvip: Learning dexterous grasping with human hand pose priors from video. In *Conference on Robot Learning*, pages 651–661. PMLR, 2022. 3

[42] Ajay Mandlekar, Yuke Zhu, Animesh Garg, Jonathan Booher, Max Spero, Albert Tung, Julian Gao, John Emmons, Anchit Gupta, Emre Orbay, et al. Roboturk: A crowdsourcing platform for robotic skill learning through imitation. In *Conference on Robot Learning*, pages 879–893. PMLR, 2018. 2

[43] Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. *arXiv preprint arXiv:2108.03298*, 2021. 1, 2

[44] Ajay Mandlekar, Soroush Nasiriany, Bowen Wen, Iretiayo Akinola, Yashraj Narang, Linxi Fan, Yuke Zhu, and Dieter Fox. Mimicgen: A data generation system for scalable robot learning using human demonstrations. *arXiv preprint arXiv:2310.17596*, 2023. 2, 15

[45] Oier Mees, Markus Merklinger, Gabriel Kalweit, and Wolfram Burgard. Adversarial skill networks: Unsupervised robot skill learning from video. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4188–4194. IEEE, 2020. 3

[46] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022. 2

[47] Anh Nguyen, Dimitrios Kanoulas, Luca Muratore, Darwin G Caldwell, and Nikos G Tsagarakis. Translating videos to

commands for robotic manipulation with deep recurrent neural networks. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3782–3788. IEEE, 2018. 2

[48] Abhishek Padalkar, Acorn Pooley, Ajinkya Jain, Alex Bewley, Alex Herzog, Alex Irpan, Alexander Khazatsky, Anant Rai, Anikait Singh, Anthony Brohan, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023. 1, 2

[49] Simone Parisi, Aravind Rajeswaran, Senthil Purushwalkam, and Abhinav Gupta. The unsurprising effectiveness of pre-trained vision models for control. In *International Conference on Machine Learning*, pages 17359–17371. PMLR, 2022. 2

[50] Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter 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. 3

[51] Haozhi Qi, Ashish Kumar, Roberto Calandra, Yi Ma, and Jitendra Malik. In-hand object rotation via rapid motor adaptation. In *Conference on Robot Learning*, pages 1722–1732. PMLR, 2023. 3

[52] Haozhi Qi, Brent Yi, Sudharshan Suresh, Mike Lambeta, Yi Ma, Roberto Calandra, and Jitendra Malik. General in-hand object rotation with vision and touch. In *Conference on Robot Learning*, pages 2549–2564. PMLR, 2023. 3

[53] Yuzhe Qin, Hao Su, and Xiaolong Wang. From one hand to multiple hands: Imitation learning for dexterous manipulation from single-camera teleoperation. *RA-L*, 7(4):10873–10881, 2022. 2

[54] Yuzhe Qin, Yueh-Hua Wu, Shaowei Liu, Hanwen Jiang, Ruihan Yang, Yang Fu, and Xiaolong Wang. Dexmv: Imitation learning for dexterous manipulation from human videos. In *European Conference on Computer Vision*, pages 570–587. Springer, 2022. 2

[55] Yuzhe Qin, Wei Yang, Binghao Huang, Karl Van Wyk, Hao Su, Xiaolong Wang, Yu-Wei Chao, and Dietor Fox. Anyteleop: A general vision-based dexterous robot arm-hand teleoperation system. *arXiv preprint arXiv:2307.04577*, 2023. 4

[56] Kanishka Rao, Chris Harris, Alex Irpan, Sergey Levine, Julian Ibarz, and Mohi Khansari. Rl-cyclegan: Reinforcement learning aware simulation-to-real. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11157–11166, 2020. 2

[57] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *International journal of computer vision*, 115:211–252, 2015. 7

[58] Manolis Savva, Abhishek Kadian, Oleksandr Maksymets, Yili Zhao, Erik Wijmans, Bhavana Jain, Julian Straub, Jia Liu, Vladlen Koltun, Jitendra Malik, et al. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9339–9347, 2019. 2

[59] Karl Schmeckpeper, Oleh Rybkin, Kostas Daniilidis, Sergey Levine, and Chelsea Finn. Reinforcement learning with videos: Combining offline observations with interaction. *arXiv preprint arXiv:2011.06507*, 2020. 2

[60] Max Schwarzer, Nitarshan Rajkumar, Michael Noukhovitch, Ankesh Anand, Laurent Charlin, R Devon Hjelm, Philip Bachman, and Aaron C Courville. Pretraining representations for data-efficient reinforcement learning. *Advances in Neural Information Processing Systems*, 34:12686–12699, 2021. 2

[61] Pierre Sermanet, Corey Lynch, Yevgen Chebotar, Jasmine Hsu, Eric Jang, Stefan Schaal, Sergey Levine, and Google Brain. Time-contrastive networks: Self-supervised learning from video. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 1134–1141. IEEE, 2018. 2

[62] Rutav Shah and Vikash Kumar. Rrl: Resnet as representation for reinforcement learning. *arXiv preprint arXiv:2107.03380*, 2021. 2

[63] Lin Shao, Toki Migimatsu, Qiang Zhang, Karen Yang, and Jeannette Bohg. Concept2robot: Learning manipulation concepts from instructions and human demonstrations. *The International Journal of Robotics Research*, 40(12-14):1419–1434, 2021. 2

[64] Kenneth Shaw, Shikhar Bahl, and Deepak Pathak. Videodex: Learning dexterity from internet videos. In *Conference on Robot Learning*, pages 654–665. PMLR, 2023. 2

[65] Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019. 2

[66] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Cliport: What and where pathways for robotic manipulation. In *Conference on Robot Learning*, pages 894–906. PMLR, 2022. 2

[67] Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. Progprompt: Generating situated robot task plans using large language models. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 11523–11530. IEEE, 2023. 2

[68] Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. *Advances in neural information processing systems*, 28, 2015. 13

[69] Simon Stepputtis, Joseph Campbell, Mariano Phielipp, Stefan Lee, Chitta Baral, and Heni Ben Amor. Language-conditioned imitation learning for robot manipulation tasks. *Advances in Neural Information Processing Systems*, 33: 13139–13150, 2020. 2

[70] Adam Stooke, Kimin Lee, Pieter Abbeel, and Michael Laskin. Decoupling representation learning from reinforcement learning. In *International Conference on Machine Learning*, pages 9870–9879. PMLR, 2021. 2

[71] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter 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. 3

[72] Joanne Truong, Sonia Chernova, and Dhruv Batra. Bi-directional domain adaptation for sim2real transfer of embodied navigation agents. *IEEE Robotics and Automation Letters*, 6(2):2634–2641, 2021. 3

[73] Luobin Wang, Runlin Guo, Quan Vuong, Yuzhe Qin, Hao Su, and Henrik Christensen. A real2sim2real method for robust object grasping with neural surface reconstruction. In *2023 IEEE 19th International Conference on Automation Science and Engineering (CASE)*, pages 1–8. IEEE, 2023. 3

[74] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, et al. Sapien: A simulated part-based interactive environment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11097–11107, 2020. 4

[75] Tete Xiao, Ilija Radosavovic, Trevor Darrell, and Jitendra Malik. Masked visual pre-training for motor control. *arXiv preprint arXiv:2203.06173*, 2022. 2

[76] Lin Yen-Chen, Andy Zeng, Shuran Song, Phillip Isola, and Tsung-Yi Lin. Learning to see before learning to act: Visual pre-training for manipulation. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7286–7293. IEEE, 2020. 2

[77] Zhao-Heng Yin, Binghao Huang, Yuzhe Qin, Qifeng Chen, and Xiaolong Wang. Rotating without seeing: Towards in-hand dexterity through touch. *arXiv preprint arXiv:2303.10880*, 2023. 3

[78] Tianhe Yu, Ted Xiao, Austin Stone, Jonathan Tompson, Anthony Brohan, Su Wang, Jaspiar Singh, Clayton Tan, Jodilyn Peralta, Brian Ichter, et al. Scaling robot learning with semantically imagined experience. *arXiv preprint arXiv:2302.11550*, 2023. 2

[79] Ying Yuan, Haichuan Che, Yuzhe Qin, Binghao Huang, Zhao-Heng Yin, Kang-Won Lee, Yi Wu, Soo-Chul Lim, and Xiaolong Wang. Robot synesthesia: In-hand manipulation with visuotactile sensing. *arXiv preprint arXiv:2312.01853*, 2023. 3

[80] Zhecheng Yuan, Zhengrong Xue, Bo Yuan, Xueqian Wang, Yi Wu, Yang Gao, and Huazhe Xu. Pre-trained image encoder for generalizable visual reinforcement learning. *Advances in Neural Information Processing Systems*, 35:13022–13037, 2022. 2

[81] Sergey Zakharov, Wadim Kehl, and Slobodan Ilic. Deceptionnet: Network-driven domain randomization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 532–541, 2019. 3

[82] Kevin Zakka, Andy Zeng, Pete Florence, Jonathan Tompson, Jeannette Bohg, and Debidatta Dwibedi. Xirl: Cross-embodiment inverse reinforcement learning. In *Conference on Robot Learning*, pages 537–546. PMLR, 2022. 3

[83] Yanjie Ze, Nicklas Hansen, Yinbo Chen, Mohit Jain, and Xiaolong Wang. Visual reinforcement learning with self-supervised 3d representations. *IEEE Robotics and Automation Letters*, 8(5):2890–2897, 2023. 2

[84] Andy Zeng, Pete Florence, Jonathan Tompson, Stefan Welker, Jonathan Chien, Maria Attarian, Travis Armstrong, Ivan Krasin, Dan Duong, Vikas Sindhwani, et al. Transporter networks: Rearranging the visual world for robotic manipulation. In *Conference on Robot Learning*, pages 726–747. PMLR, 2021. 2

[85] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023. 6, 13

[86] Zhengbang Zhu, Hanye Zhao, Haoran He, Yichao Zhong, Shenyu Zhang, Yong Yu, and Weinan Zhang. Diffusion models for reinforcement learning: A survey. *arXiv preprint arXiv:2311.01223*, 2023. 2

CyberDemo: Augmenting Simulated Human Demonstration for Real-World Dexterous Manipulation

Supplementary Material

A . Overview

This supplementary document offers further information, results, and visualizations to complement the primary paper. Specifically, we encompass:

- Details on data collection;
- Details on training and testing procedures;
- Details on the design of evaluation levels;
- Comparision to other data generation methods;
- More ablation studies;
- Additional details on the derivation of data augmentation for randomizing object poses.

B . Implementation details

In this section, we provide an overview of the data collection, training, and testing processes.

B .1. Human Demonstration Collection

The human play data is gathered through a teleoperation setup, where a human operator controls the system using a single real-sense camera in both the simulated and real environments. The entire trajectory is recorded at a rate of 30 frames per second, with each trajectory spanning approximately 20-30 seconds.

In the real-world setting, an additional real-sense camera is used to capture RGB images, which serve as the observations in the dataset. To ensure alignment between the simulated and real environments, we perform hand-eye calibration in the real world. This calibration process allows us to determine the relative position between the camera and the robot arm, enabling us to apply this transformation in the simulation.

B .2. Real World Setup

The system design for data collection is shown in Figure 6. As represented in the figure, the collection of human play data incorporates a human operator and a camera. The camera captures video footage at a frequency of 30 frames per second. Throughout the data collection process, the human operator interacts with the scene without any defined task objective. Instead, they interact freely with the environment, motivated by curiosity and the intent to observe intriguing behaviors.

In our experiments, human play data is collected by recording 30 seconds of uninterrupted interaction in each demonstration. This timeframe permits ample data to be gathered, yielding a rich and varied collection of behaviors for examination and study.

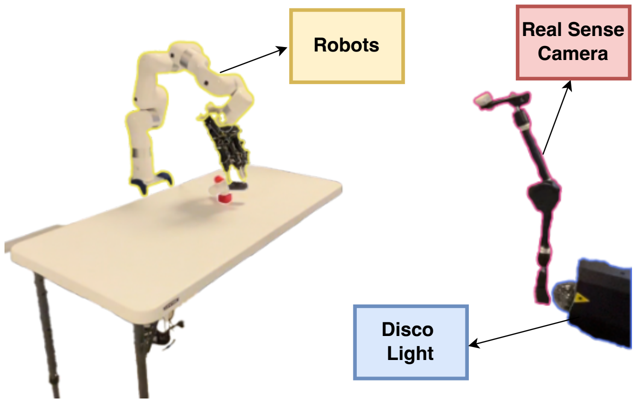


Figure 6. **Details of System Setups in Real World**

B .3. Policy Training

We use a conditional VAE [68] for training on 100 simulation demonstrations. The action chunking size was fixed at 50, in line with the methodology adopted in [85]. Following the simulation data training, the model was fine-tuned with 15 real-world demonstrations, using a smaller learning rate and distinct batch norms for the real-world data.

Hyperparameters related to policy learning are displayed in Table 5, whereas Table 6 lists the hyperparameters pertinent to auto-curriculum learning.

To infuse diversity into the augmentation process, we have incorporated a randomness scale that ranges from 0 to 10 for each augmentation. In the context of auto-curriculum learning, this randomness scale progressively rises with a constant variance throughout each testing cycle.

In the course of auto-curriculum learning, the policy's performance is assessed across all four simulation levels, and the success rate is averaged. If the success rate falls below the success rate threshold, the increase in randomness scale is halted. This strategy aids in maintaining a balance between introducing randomness and ensuring the policy consistently accomplishes its tasks.

In summary, these hyperparameters and the evaluation procedure in auto-curriculum learning allow the policy to evolve and enhance over time, gradually escalating the randomness scale while preserving a satisfactory success rate.

Hyperparameter	Default
Batch Size	128
Num of Epochs	None
Finetuning Epochs	3000
Optimizer	AdamW
Learning Rate (LR)	1e-5
Finetuning LR	1e-6
Weight Decay	1e-2
Evaluation Frequency	100 epochs
Encoder Layers	4
Decoder Layers	7
Heads	8
Feedforward Dimension	3200
Hidden Dimension	256
Chunk Size	50
Dropout	0.1

Table 5. **Hyperparameters of Policy Network**

Hyper Parameters	Default
Test Cycles	300
Evaluation Freq	100 epochs
Randomness Variance For Each Cycle	0.2
Success Rate Threshold	15%
Data Generation Rate Threshold	30%

Table 6. **Hyperparameters for Auto Curriculum Learning**

B .4. Policy Testing

During the real-world testing phase, we perform both in-domain and out-of-domain tests to evaluate the performance of the model. For out-of-domain tests, we significantly randomize the positions of objects, consciously choosing locations not included in the original data. This step guarantees that the model is examined in unfamiliar situations, evaluating its capacity to generalize and adjust to novel object arrangements.

Moreover, to introduce visual disruptions and test the robustness of the model, we incorporate a disco light. The disco light generates visual disturbances and adds an extra layer of complexity to the test environment. This approach enables us to assess the model's resilience in dealing with unexpected visual inputs and its ability to sustain performance amidst such disruptions.

In the concluding stage, we evaluate the policy's ability to generalize across a range of objects, as illustrated in Figure 7. To carry out this generalizability test, we enhance the initial 100 simulation demonstrations by introducing 10 unique objects (adding 10 additional demonstrations for each object), and then re-run our pipeline. For the pick and place task in the real-world setting, we collected 15 demonstrations involving three different objects (with five demonstrations performed for each object within the red frame). For the rotating task, the real-world dataset includes only one object, identical to the original testing case.

By conducting these assessments and incorporating a variety of objects, we aim to evaluate the policy's adaptability and performance in diverse situations, ensuring its robustness and flexibility. A selection of demos is displayed in Figure 8.

B .5. Details of Simulation Evaluation Level designs

In the Pick & Place and Pour tasks, we have defined different levels to introduce varying degrees of randomness:

- Level 1 signifies the original domain and encompasses slight randomization of the pose of the manipulated objects, including the end-effector pose and orientation.
- Level 2 includes randomization of lighting and texture.
- Level 3 incorporates minimal randomization of the target objects (plate in pick place, bowl in pouring).
- Level 4 escalates the randomness scale of both the manipulated and target objects.

For Rotate task since there is only one object, things are different for the Rotate task:

- Level 1 is set as randomizing the orientation of the objects, which is also the original domain.
- Level 2 is the same as pick place and pouring tasks.
- Level 3 is adding the randomization of the end-effector pose of the manipulated objects.
- Level 4 increases the randomness scale of the manipulated objects.

Below are the defined randomness parameters for each task, with all numbers listed in international units if without a statement. In our settings, the position (0,0) represents the center of the table.

Level Design for Pick and Place

- Random Manipulated Object Pose with a small scale:
  - The x-coordinate of the Manipulated Object ranges from -0.1 to 0.1.
  - The y-coordinate of the Manipulated Object ranges from 0.2 to 0.3.
  - The Manipulated Object's z-axis Euler degree ranges from 80 to 90.
- Random Light and Texture(The randomness scale here is fixed to be 2):
  - The direction of the light is constrained within a circular range. The radius of this circle spans from 0.5 to the randomness scale * 0.1.
  - To determine the color of each channel for the lights, a uniform sampling approach is employed. This involves selecting a value within the range [default color of that channel - randomness scale * 0.1, default color of that channel + randomness scale * 0.1].
  - The ground color and sky color of the environment map are randomized in the same way as lights.

- Random Target Object Pose with a small scale:
  - The x-coordinate of the Target Object ranges from -0.1 to 0.1.
  - The y-coordinate of the Target Object ranges from -0.3 to -0.1.
- Random Manipulated and Target Object Pose with a Large scale:
  - The x-coordinate of the Manipulated Object ranges from -0.2 to 0.2.
  - The y-coordinate of the Manipulated Object ranges from 0.1 to 0.3.
  - The Manipulated Object's z-axis Euler degree ranges from 70 to 90.
  - The Manipulated Object's z-axis Euler degree ranges from 80 to 90.
  - The x-coordinate of the Target Object ranges from -0.2 to 0.2.
  - The y-coordinate of the Target Object ranges from -0.3 to 0.

Level Design for Pour

- Random Manipulated Object Pose with a small scale:
  - The x-coordinate of the Manipulated Object ranges from -0.1 to 0.1.
  - The y-coordinate of the Manipulated Object ranges from -0.2 to -0.1.
  - The Manipulated Object's z-axis Euler degree ranges from 0 to 179.
- Random Light and Texture(The randomness scale here is fixed to be 2): same as pick and place.
- Random Target Object Pose with a small scale:
  - The x-coordinate of the Target Object ranges from -0.1 to 0.1.
  - The y-coordinate of the Target Object ranges from 0.2 to 0.3.
- Random Manipulated and Target Object Pose with a Large scale:
  - The x-coordinate of the Manipulated Object ranges from -0.1 to 0.15.
  - The y-coordinate of the Manipulated Object ranges from -0.3 to 0.
  - The x-coordinate of the Target Object ranges from -0.2 to 0.2.
  - The y-coordinate of the Target Object ranges from 0.2 to 0.4.
  - The Manipulated Object's z-axis Euler degree ranges from 0 to 359.

Level Design Rotate

- Random Manipulated Object Pose with a small scale:
  - The Manipulated Object's z-axis Euler degree ranges from 0 to 30.
- Random Light and Texture(The randomness scale here is fixed to be 2): same as pick and place.
- Random Manipulated Object Pose with a small scale:

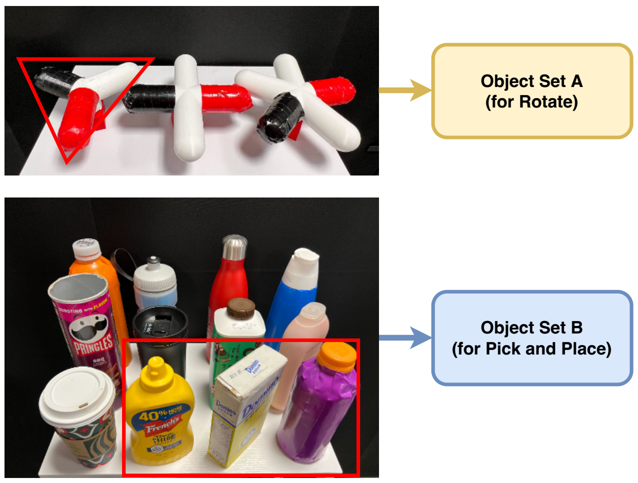


Figure 7. **Object Sets in Real World.** The objects located within the red frame are allocated for training, while the remaining objects are set aside for testing on previously unseen objects.

  - The x-coordinate of the Target Object ranges from -0.1 to 0.1.
  - The y-coordinate of the Target Object ranges from -0.15 to 0.15.
- Random Manipulated Pose with a Large scale:
  - The x-coordinate of the Manipulated Object ranges from -0.2 to 0.2.
  - The y-coordinate of the Manipulated Object ranges from -0.3 to 0.3.
  - The Manipulated Object's z-axis Euler degree ranges from 0 to 60.

C . More Experimental Results

C .1. Comparision of Data Generation Method

To test our data augmentation approach's effectiveness, we pretrained using simulation data augmented by MimicGen [44] and then fine-tuned with real-world teleoperation data, a sim2real transfer not included in the original MimicGen framework. As shown in Figure 7, MimicGen adds an interpolated trajectory (in purple) to new object poses, potentially causing abrupt transitions. Our method, however, seamlessly integrates the entire sequence, resulting in more fluid motion. The imitation learning policy trained with our data thus outperforms others, as evidenced by the improved results in simulation and reality shown in the table.

C .2. Ablation on Action Aggregation

As illustrated in Figure 9, the use of action aggregation with Small Motion extends beyond its advantages in imitation learning within a single domain. It also functions as an effective instrument in closing the gap between simulated and real environments. As illustrated in Figure 10, the success

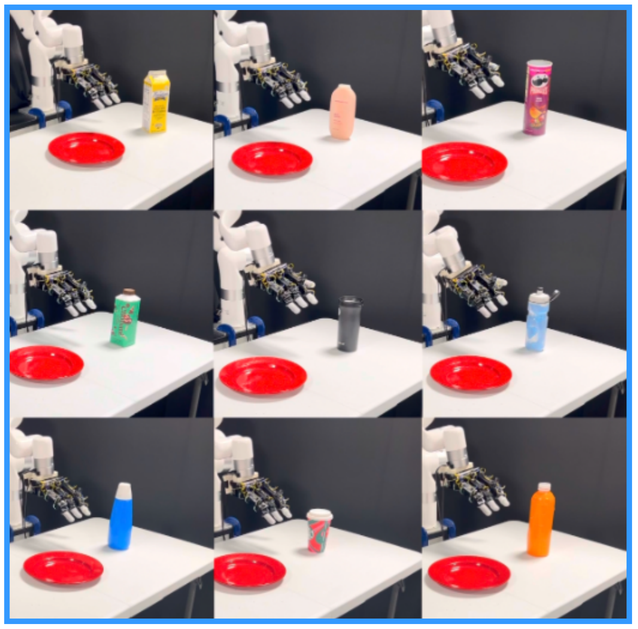
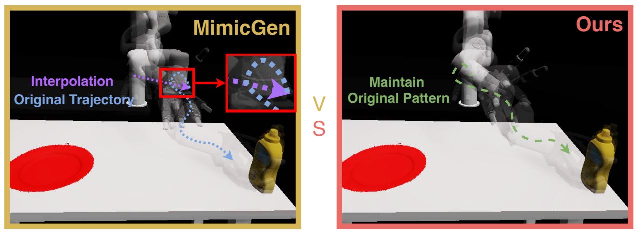


Figure 8. **Pick and Place Evaluation on diverse real-world objects.**

	Simulation				Real World		
	Level 1	Level 2	Level 3	Level 4	In Domain	Out of Position	Random Light
MimicGen	75.5%	49%	19.5%	14%	2/20	1/20	5/20
Ours	**80%**	**61%**	**43.5%**	**57%**	**7/20**	**6/20**	**8/20**

Table 7. **Comparision to MimicGen**. We compare our data augmentation method with the one used in MimicGen on the Pick and Place task. For real-world experiments, we fine-tuned it with the same real-world data as other methods.



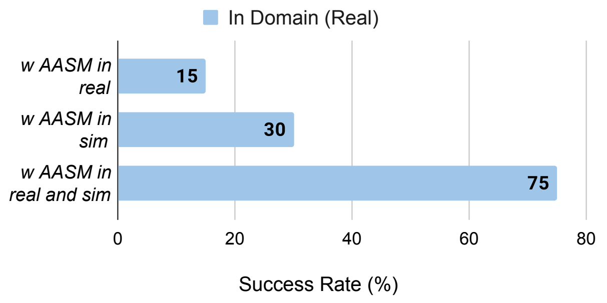


Figure 9. **Success Rate on Action Aggregation**

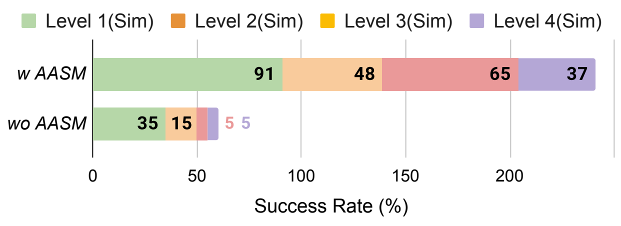


Figure 10. **Ablation on Action Aggregation with Small Motion in Simulation.** Success rate evaluate in simulator when the policy is trained on dataset with and without action aggregation.

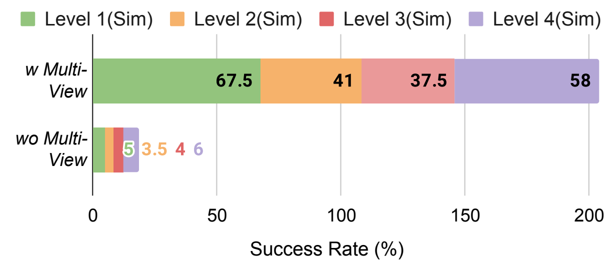


Figure 11. **Ablation on Multi-View Augmentaion**

rate of the policy becomes more pronounced as the level of difficulty escalates. This suggests that action aggregation becomes increasingly beneficial for more challenging tasks.

C .3. Ablation on Novel Camera Views

As demonstrated in Figure 11, the application of random camera views augmentation improves the policy's robustness, particularly in situations involving camera view changes. In this method, all levels remain consistent while minor alterations to the camera view are incorporated.

These changes involve a combined rotation along the y-axis and z-axis, plus a slight shift in the x, y, and z directions. The rotation Euler angle is sampled within the range of $[-15, 15]$, enabling managed variation in the camera's alignment. Additionally, the translation is sampled within the range of $[-0.05m, 0.05m]$, allowing minor adjustments in the camera's placement.

By integrating these alterations, the multi-view augmentation method introduces realistic variations in camera perspectives, thereby enhancing the policy's resilience to changes in the viewpoint. This strategy boosts the model's capability to adapt and perform efficiently, even when confronted with varied camera angles and positions.

C .4. Ablation on Kinematics Augmentation

We ablate the augmentation methods with or without sensitivity analysis (in contrast, simply relocating the end-effector to a new pose). We test both methods in an environment where object poses are extensively randomized, to verify the effectiveness of these two kinematic augmentation approaches. Figure 12 illustrates the success rate during training using datasets generated via these distinct augmentation strategies. Our findings demonstrate that, through the application of our proposed techniques, the model demonstrates consistent improvement over all three manipulation tasks. These outcomes underscore the efficacy of incorporating sensitivity analysis into pose data augmentation.

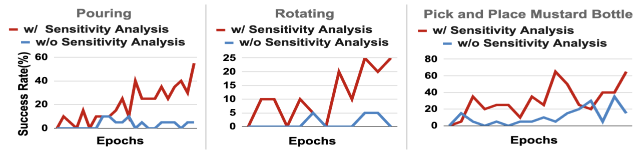


Figure 12. **Augmentation Comparison** We apply kinematic augmentation to one selected original demo, adapting it to a new position utilizing both the MimicGen method and ours.

D . Derivation of Data Augmentation

In this section, we delve deeper into the derivation of the formula applied in the data augmentation of random object pose. We reproduce Equation 2 from the main paper and provide a detailed explanation:

$$\begin{aligned}\overline{\psi}_{seg_j} &= \frac{\psi_{seg_j}}{\sum_{j=1}^{M}\psi_{seg_j}}, \quad \forall seg_j \\ \Delta T_j &= exp(\overline{\psi}_{seg_j}\log(\Delta T)/K) \\ a_i^{new} &= a_i f_i(\Delta T_j)\end{aligned} \tag{3}$$

The first line of the equation normalizes the robustness score computed from Equation 1 in the main paper, ensuring that the sum of all scores equals 1. This parameter can be interpreted as a weight for each action chunk, symbolizing the proportion of modification each chunk should undertake to guide the robot to the new pose.

The second line calculates the relative pose modification for each step in chunk j. There are K steps in chunk j, and each step is allocated the same quantity of modification within the same chunk. Here, $log()$ maps the $SE(3)$ Lie Group to its $se(3)$ Lie algebra, where exp is its inverse, mapping $se(3)$ back to $SE(3)$.

The third line of this equation computes the new action based on the pose modification. Here, f_i is a similarity transformation in the $SE(3)$ space that transitions the motion from the world frame to the current end-effector frame. We now provide a detailed derivation of f_i. Since $\Delta T = T_W^{O_{new}}(T_W^{O_{old}})^{-1} = T_{O_{old}}^{O_{new}}$, this relative pose change is a representation in the old object pose frame. To use this pose transformation to modify the action, we need to transform this relative pose into the frame corresponding to the action, which is the frame of the current end-effector pose. Considering the similarity transformation $T_B^A X (T_B^A)^{-1}$, which transforms a $SE(3)$ motion X represented in frame B to frame A, f_i can be derived as $f_i(\Delta T) = T_{R_i}^{O_{old}}\Delta T(T_{R_i}^{O_{old}})^{-1}$, where T_{R_i} is the robot end-effector pose in frame i.