

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;
- Comparison 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

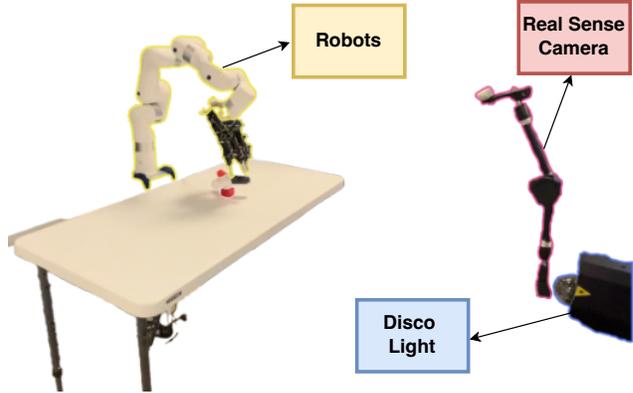


Figure 6. Details of System Setups in Real World

gathered, yielding a rich and varied collection of behaviors for examination and study.

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 4, whereas Table 5 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 4. **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 5. **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 demon-

strations 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:

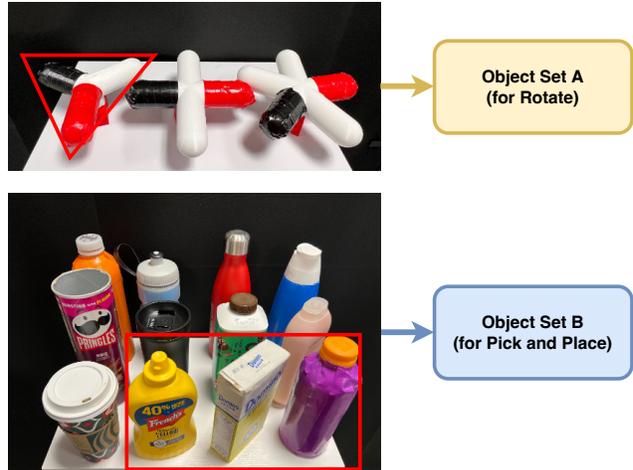


Figure 7. **Object Sets in Real World.** The objects located within the red frame are allocated for training, while the remaining objects are set aside for testing on previously unseen objects.

- 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 Mimic-Gen [44] and then fine-tuned with real-world teleoperation data, a sim2real transfer not included in the original Mimic-Gen framework. As shown in Figure 6, 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 6. **Comparison 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.

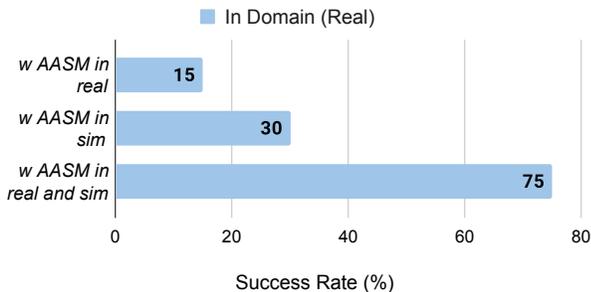
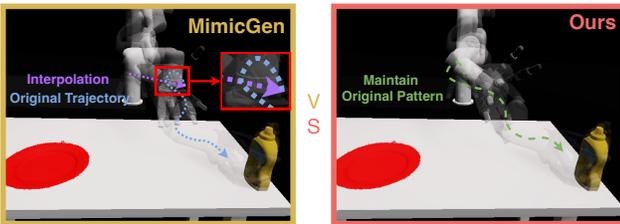


Figure 9. **Success Rate on Action Aggregation**



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 Augmentation**

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

Dataset	Test in Sim				Test in Real	
	Level 1	Level 2	Level 3	Level 4	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 7. **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.

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.

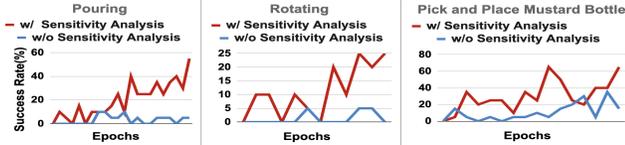


Figure 12. **Augmentation Comparison** We apply kinematic augmentation to one selected original demo, adapting it to a new position utilizing both the MimicGen method and ours.

C.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 7. We observe that training solely on 50 real demonstrations results in poor performance, and the policy overfit to joint positions rather than utilizing the visual information in images. The best results were obtained with a combination of 15 simulation 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.

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:

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

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 $\Delta T = T_W^{Onew}(T_W^{Oold})^{-1} = T_{Oold}^{Onew}$, 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 $f_i(\Delta T) = T_{R_i}^{Oold} \Delta T (T_{R_i}^{Oold})^{-1}$, where T_{R_i} is the robot end-effector pose in frame i .