

# Supplementary Material for: Constraint-Preserving Data Generation for One-Shot Visuomotor Policy Generalization

Anonymous Author(s)

Affiliation

Address

email

## 1 Method Details

### 1.1 Source Demonstration Collection

For all CP-Gen data generation, we collect one expert demonstration per task using a custom tele-operation system based on the Gello framework [1]. We use the Gello ‘leader’ arm to control the end-effector pose of the Franka Emika Panda robot in simulation. The underlying controller is a Whole-Body Inverse Kinematics controller provided by Robosuite [2].

### 1.2 Keypoint Annotation

We annotate “actor keypoints”—task-relevant 3D points on the gripper and/or the manipulated object—to extract keypoint-trajectory constraints that enable geometry aware data generation. When the actor keypoints are located on the gripper (e.g., fingertips or center of grasp), we reuse the same set of predefined gripper keypoints across all environments. When the keypoints are located on the object, we use the MuJoCo [3] viewer interface to select 3D keypoints on the task relevant object’s geometry. As described in the main text, though we opt for manual annotation of keypoints in this paper, automatic methods are also possible [4].

### 1.3 Policy Training Details

For both simulation and real-world experiments, we leverage the DDPM implementation from the Diffusion Policy [5] paper. We use the U-Net architecture conditioned on a sequence of RGB images to predict future actions. We adopt a fixed diffusion horizon and use default policy hyperparameters across all tasks unless otherwise stated. Key training settings are summarized in Table 1.

Table 1: Training details for diffusion policy.

Category	Detail
Horizon	16 steps (2 observation steps, 8 action steps)
Backbone	ResNet-18 (GroupNorm, no pretrained weights)
Crop Size	$76 \times 76$ (random crop from fixed center crop)
Diffusion Steps	100 (DDPM with <code>squaredcos_cap_v2</code> schedule)
U-Net Config	Downsampling dimensions: [512, 1024, 2048]; kernel size: 5; 8 GroupNorm groups
Optimizer	AdamW (learning rate $1e-4$ , betas = [0.95, 0.999], weight decay $1e-6$ )
Batch Size	64
Learning Rate Schedule	Cosine decay with 500-step warmup
EMA	Enabled (inv_gamma = 1.0, power = 0.75)

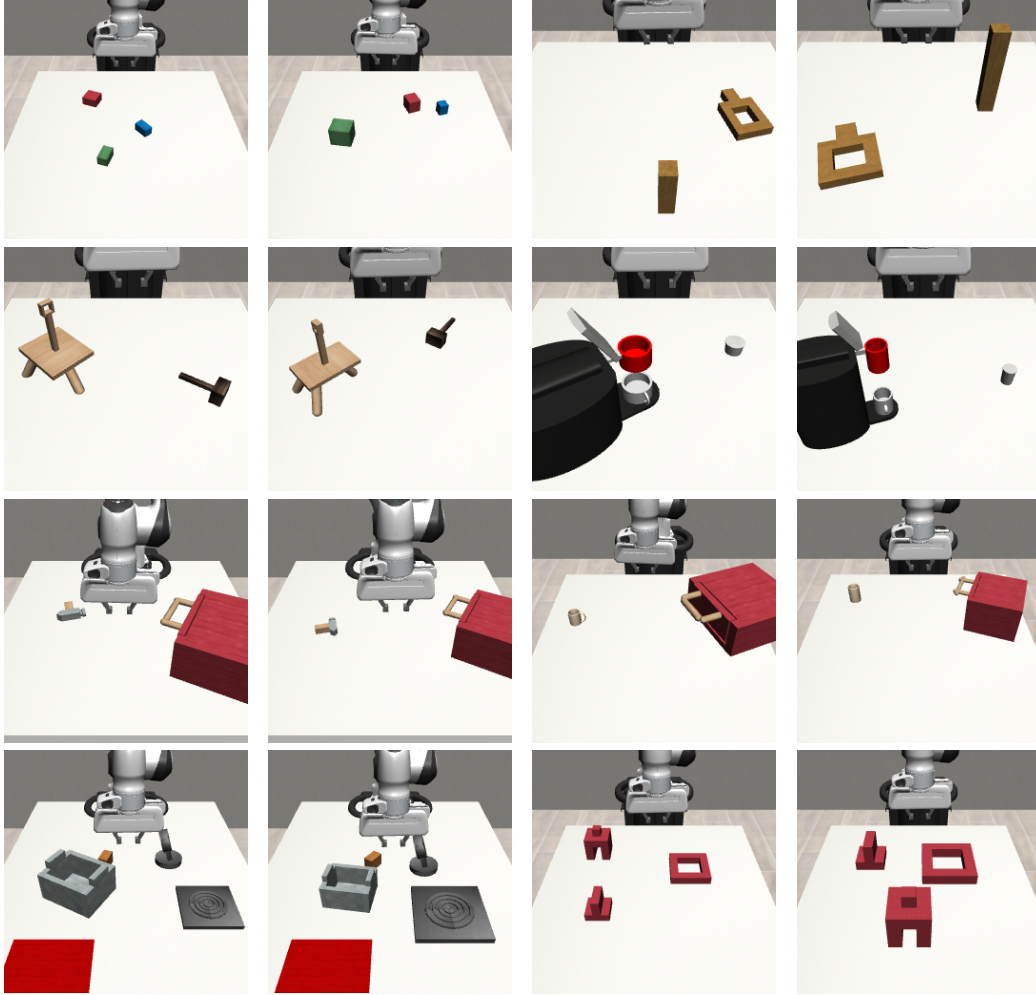


Figure 1: **Examples of Geometry Generalization.** For each task, we show two examples of scale factors applied to each axis on the task-relevant objects.

For simulation tasks that include depth and segmentation inputs, we use  $84 \times 84$  image resolutions and provide both depth and segmentation masks from both robot arm cameras. To mimic the real-world setup, we do not use depth from the hand-mounted camera.

For real-world experiments, raw observations are first resized to  $90 \times 160$ , followed by a center crop to  $90 \times 90$ . We then apply the same random cropping procedure as in simulation to produce the final  $76 \times 76$  input.

## 2 Simulation Experiment Details

We evaluate CP-Gen on eight simulated manipulation tasks introduced in MimicGen [6], covering short and long-horizon behaviors such as stacking, insertion, drawer manipulation, and object assembly. All tasks are executed in MuJoCo [3] using the Robosuite [2] framework with an inverse kinematics controller at 20 Hz. The action space is an absolute end-effector pose and gripper joint.

**Default Task Setting.** For all tasks, we use the MimicGen-defined  $D_1$  variant, which introduces modest spatial variation in object placement and top-down rotation. Below we describe each of the eight MimicGen [6] tasks used in our study:

- **Stack Three:** The robot must sequentially stack a red block onto a green block and a blue block onto the red one. There are 4 subtasks in total.
- **Square:** The robot picks up a square nut and places it onto a peg. There are 2 subtasks: grasping the nut and inserting it onto the peg.
- **Threading:** The robot must grasp a needle and thread it through a tripod hole. The task requires precise control and involves 2 subtasks.
- **Coffee:** The robot picks up a coffee pod, inserts it into a coffee machine, and closes the hinge. This task involves 2 subtasks.
- **Three Piece Assembly:** The robot picks and inserts two separate pieces into a base structure to assemble a composite object. The task contains 4 subtasks.
- **Hammer Cleanup:** The robot opens a drawer, picks up a hammer, places it inside the drawer, and closes the drawer. There are 4 subtasks in total.
- **Mug Cleanup:** Similar to Hammer Cleanup, but with a mug object. The task includes object-centric variations involving different mug geometries. It contains 4 subtasks.
- **Kitchen:** The robot must switch the stove on, place a pot onto the stove, pick and place bread into the pot, move the pot to the serving region, and then turn the stove off. This long-horizon task involves 7 sequential subtasks.

**Geometry Generalization Benchmark.** To evaluate the robustness of CP-Gen to changes in object geometry, we introduce a new benchmark involving geometry generalization. Specifically, we keep the same pose reset distributions from the default tasks settings, but apply non-uniform scaling transforms to task relevant objects. We show samples scene resets for our custom benchmark in Figure 1.

### 3 Real-World Experiment Details

**Real-World Setup.** For our real-world robot, we use a Franka Emika Panda arm, with a UMI gripper [7] attached to the panda gripper. For our cameras, we use two Intel RealSense cameras: a 3rd-person view (3PV) camera statically mounted, and an eye-in-hand (hand) camera mounted to the wrist.

**Digital Twin Visualizations.** For each real-world task, we construct a simulation-based digital twin environment. Figure 2 shows RGB, depth map and binary segmentation mask renderings for all four real-world tasks, from both third-person and eye-in-hand perspectives.

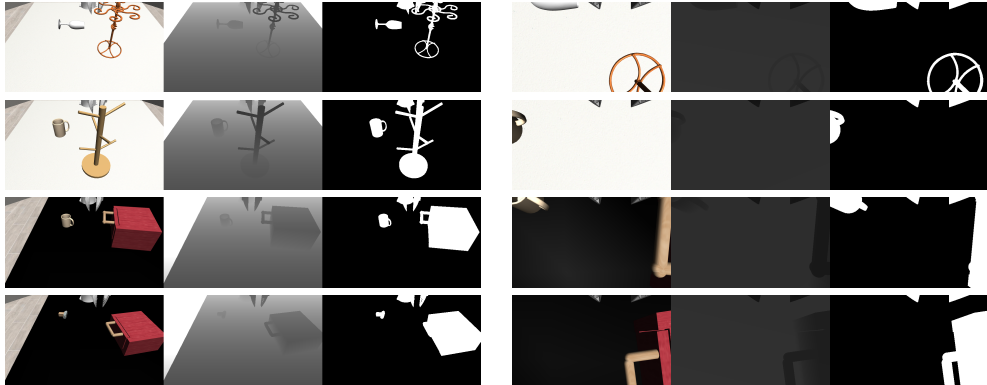


Figure 2: Digital twin environments for each real-world task. Left: third-person agentview. Right: eye-in-hand view.

**Segmentation Masks.** To obtain segmentation masks in the real world, we first use the Segment Anything Model (SAM) [8] to generate binary segmentation masks. Then, we apply adaptive color thresholding using Otsu’s method [9] within the selected SAM masks to isolate foreground objects from the black table background.

## References

- [1] P. Wu, Y. Shentu, Z. Yi, X. Lin, and P. Abbeel, *Gello: A general, low-cost, and intuitive tele-operation framework for robot manipulators*, 2023.
- [2] Y. Zhu *et al.*, “Robosuite: A modular simulation framework and benchmark for robot learning,” in *arXiv preprint arXiv:2009.12293*, 2020.
- [3] E. Todorov, T. Erez, and Y. Tassa, “Mujoco: A physics engine for model-based control,” in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2012, pp. 5026–5033. DOI: [10.1109/IROS.2012.6386109](https://doi.org/10.1109/IROS.2012.6386109).
- [4] W. Huang, C. Wang, Y. Li, R. Zhang, and L. Fei-Fei, “Rekep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation,” in *8th Annual Conference on Robot Learning*, 2024. [Online]. Available: <https://openreview.net/forum?id=9iG3SEbMnL>.
- [5] C. Chi *et al.*, “Diffusion policy: Visuomotor policy learning via action diffusion,” in *Proceedings of Robotics: Science and Systems (RSS)*, 2023.
- [6] A. Mandlekar *et al.*, “Mimicgen: A data generation system for scalable robot learning using human demonstrations,” in *7th Annual Conference on Robot Learning*, 2023.
- [7] C. Chi *et al.*, “Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots,” in *Proceedings of Robotics: Science and Systems (RSS)*, 2024.
- [8] N. Ravi *et al.*, “Sam 2: Segment anything in images and videos,” *arXiv preprint arXiv:2408.00714*, 2024. [Online]. Available: <https://arxiv.org/abs/2408.00714>.
- [9] N. Otsu, “A threshold selection method from gray-level histograms,” *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, no. 1, pp. 62–66, 1979. DOI: [10.1109/TSMC.1979.4310076](https://doi.org/10.1109/TSMC.1979.4310076).