

# AffordGen: Generating Diverse Demonstrations for Generalizable Object Manipulation with Affordance Correspondence

## Supplementary Material

### 6. Hyperparameters

#### 6.1. Point Cloud Processing

We follow the preprocessing pipeline for point cloud observations as outlined in DP3 [36]. For simulation tasks, we directly apply Farthest Point Sampling (FPS) to downsample the point cloud to 1024 points. For real-world tasks, we collect point clouds using a RealSense L515 camera at a depth image resolution of  $1024 \times 768$  pixels. The collected point cloud is first clustered using DBSCAN to mitigate the influence of outliers, then downsampled to 1024 points via FPS. In real-robot experiments, an  $\epsilon$  of 1.0 cm was used for the teapot, mug, and knife experiments, while an  $\epsilon$  of 2.0 cm was used for the shoe experiment.

#### 6.2. Policy Training and Evaluation

For all simulation and real tasks, we train the policy for 1000 epochs with a batch size of 512. To stabilize the training process, we use the Adam optimizer and cosine learning rate scheduler with 500 warmup steps. We set the observation horizon  $T_o = 2$ , action prediction horizon  $T_p = 16$ , and action execution horizon  $T_a = 6$  for the teapot, mug, and knife tasks. For long-horizon tasks like the shoe, we set  $T_o = 2$ ,  $T_p = 16$ , and  $T_a = 16$  to avoid getting stuck.

### 7. Experiment Details

#### 7.1. Task Description

We summarize the four tasks (which are the same for both simulation and real-world setups) as follows:

- **Teapot Pouring:** Grasp the handle, position the spout above the cup, and tilt beyond a threshold angle.
- **Mug Hanging:** Grasp the handle and hang the mug by threading its handle onto the rack.
- **Knife Cutting:** Grasp the handle and bring the blade into perpendicular contact with the object.
- **Shoe Aligning:** Grasp the heel and place both shoes on the rack in the same orientation.

For simulation tasks, we summarize the randomization range during evaluation as follows:

	Teapot	Mug	Knife	Shoe
<b>Position (cm)</b>	$20 \times 20$	$10 \times 15$	$20 \times 20$	$15 \times 25$
<b>Orientation (<math>^\circ</math>)</b>	$[0, 180]$	$[0, 180]$	$[0, 90]$	$[0, 120]$

For the real teapot, mug, and knife tasks, during testing, we placed each tested instance in a  $3 \times 3$  grid, evaluating

three orientations at each position in a total of 27 rollouts for each eval instance as illustrated in Figure 11.

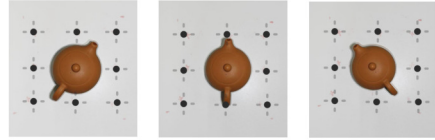


Figure 11.  $3 \times 3$  grid with three different orientations during real teapot, mug and knife evaluation.

For the real shoe task, we designed 5 initial pose configurations, as shown in Figure 12, for the two shoes to be evaluated twice at varied positions, resulting in 10 rollouts per instance. Note that the success times we report refer to the total number of successfully placed shoes, not the number of fully successful instances. Since each instance comprises both left and right shoes, the total number of times is 20, which is twice the number of evaluation rollouts.



Figure 12. Five pose configurations for real shoe evaluation.

#### 7.2. Detailed Real-world Results

We summarize the detailed success rates on each tested instance in the following sections. Note that each instance labeled 'a' serves as the source object for this task.

##### 7.2.1. Teapot Pouring



Figure 13. Teapot evaluation instances

	a	b	c	d	e	f	g
<b>AffordGen</b>	13/27	14/27	6/27	<b>18/27</b>	<b>9/27</b>	<b>11/27</b>	<b>16/27</b>
<b>DemoGen</b>	<b>14/27</b>	0/27	2/27	0/27	0/27	0/27	0/27
<b>CPGen</b>	10/27	2/27	6/27	3/27	0/27	2/27	2/27
<b>FUNCTO</b>	10/27	<b>18/27</b>	<b>8/27</b>	9/27	<b>9/27</b>	5/27	1/27

### 7.2.2. Mug Hanging



Figure 14. Mug evaluation instances

	a	b	c	d	e	f
<b>AffordGen</b>	19/27	<b>17/27</b>	16/27	<b>20/27</b>	<b>19/27</b>	16/27
<b>DemoGen</b>	<b>20/27</b>	9/27	<b>17/27</b>	0/27	9/27	<b>19/27</b>
<b>CPGen</b>	19/27	4/27	13/27	12/27	5/27	16/27
<b>FUNCTO</b>	7/27	6/27	9/27	10/27	9/27	7/27

### 7.2.3. Knife Cutting

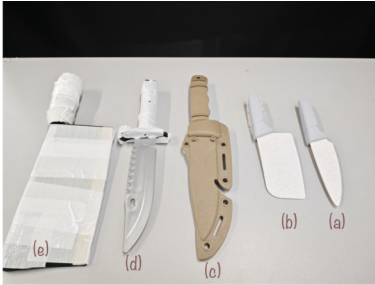


Figure 15. Knife evaluation instances

	a	b	c	d	e
<b>AffordGen</b>	23/27	<b>23/27</b>	<b>25/27</b>	23/27	<b>25/27</b>
<b>DemoGen</b>	<b>25/27</b>	20/27	10/27	16/27	1/27
<b>CPGen</b>	23/27	<b>23/27</b>	24/27	<b>24/27</b>	17/27
<b>FUNCTO</b>	21/27	20/27	20/27	10/27	11/27

### 7.2.4. Shoe Organizing



Figure 16. Shoe evaluation instances

	a	b	c	d
<b>AffordGen</b>	11/20	<b>14/20</b>	<b>15/20</b>	<b>16/20</b>
<b>DemoGen</b>	13/20	12/20	5/20	7/20
<b>CPGen</b>	<b>18/20</b>	<b>14/20</b>	8/20	8/20
<b>FUNCTO</b>	15/20	8/20	5/20	6/20

## 7.3. Baseline Implementation

### 7.3.1. DemoGen

In both simulation and real-world experiments, we compare against the DemoGen baseline. It should be noted that the original DemoGen implementation does not generate demonstrations under varying object yaw rotations relative to the camera; the object and robot arm always face the camera from the same side. In contrast, our method extends data generation to novel camera viewpoints, and we therefore introduce a simulated rendering pipeline to produce point clouds from unseen perspectives. To ensure a fair comparison, we use the same object mesh reconstructed during the AffordGen process and generate DemoGen demonstrations across diverse positions and orientations via simulation.

### 7.3.2. CPGen

We also included the CPGen baseline in both simulation and real-world experiments. The original CPGen implementation utilizes RGB-D images and segmentation masks as policy inputs to achieve sim-to-real transfer. This requires reconstructing the real environment in simulation and providing real-time segmentation masks during real-world evaluation. To ensure a fair comparison, we converted the trajectories generated by CPGen into point cloud data for both real and simulated settings, using the same digital cousin point cloud generation method as employed in AffordGen. During the implementation of CPGen, it was necessary to apply random rescaling to the original object mesh. Since the original CPGen paper does not discuss the selection of a rescaling factor, we set CPGen’s rescaling factor such

that the bounding box of the original mesh could cover the bounding box of the test meshes.

### 7.3.3. FUNCTO

We included the FUNCTO baseline in our real-world experiments. The original FUNCTO baseline, as an online planning algorithm, incorporates a complex keypoint selection pipeline that involves fine-grained adjustments of planning angle alignment using large language models. To ensure a fair comparison, we simplified FUNCTO’s keypoint selection process by directly employing DINOv2 for keypoint correspondence, which aligns with AffordGen’s approach of selecting keypoints from 2D camera views. It should be noted that the AffordGen pipeline does not inherently constrain the methodology for semantic keypoint selection; the sophisticated selection process used in the original FUNCTO implementation, albeit at the cost of annotation efficiency, is equally applicable to our AffordGen data generation pipeline.

## 7.4. Real World Experiment Demos

Please refer to the associated videos to see the performance of policies trained on AffordGen-generated data in the real world. With a small number of expert demonstrations on a single object, the data generated by AffordGen can train policies that perform well both on objects within the same category and across different categories. The learned policy is able to execute consecutively on different objects, showcasing the superior generalization ability of AffordGen.

## 8. 3D Mesh Dataset

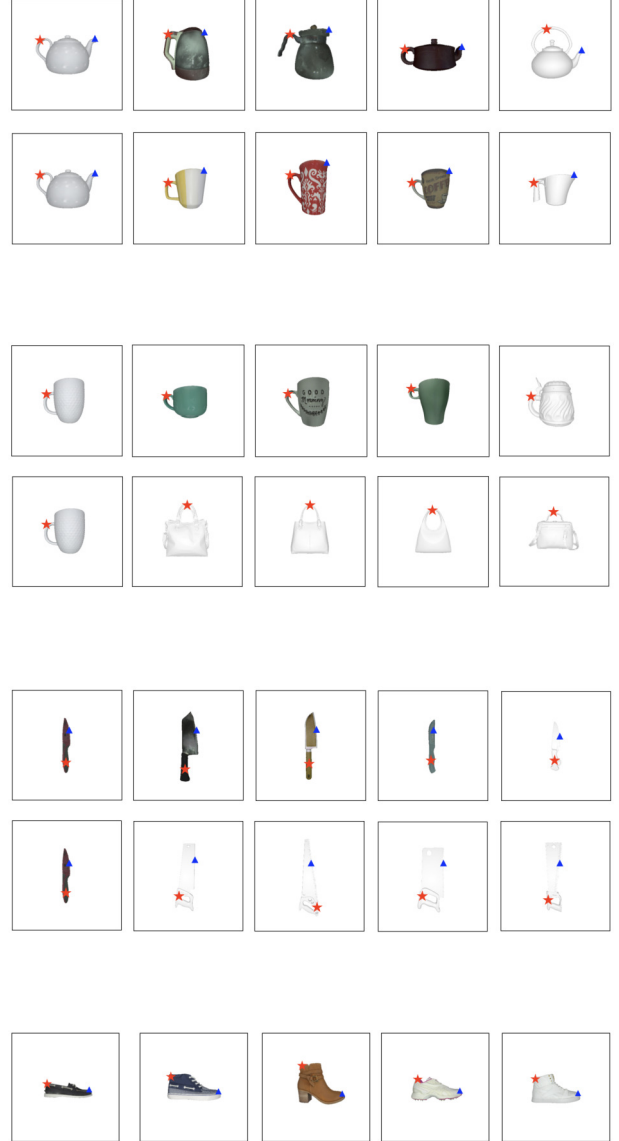
### 8.1. 3D Mesh Dataset Pre-processing

To obtain a sufficient number of meshes for a specific category, we leveraged an existing 3D generative model [27]. For the teapot and mug categories, the generated meshes are almost upright, with their in-plane rotations (within the XY plane) typically aligning with one of the four cardinal angles:  $0^\circ$ ,  $90^\circ$ ,  $180^\circ$ , or  $270^\circ$ . To estimate their poses, we employ the off-the-shelf 6D pose estimator Omni6DPose [38] by rendering images from predefined viewpoints around the object. The estimated rotation is then applied in reverse to align the mesh to a canonical pose. For the knife category, the generated meshes may have different orientations in  $SE(3)$ , which hinders the accuracy of pose canonicalization. Thus, Principal Component Analysis (PCA) is first used to extract the three principal axes of each generated knife, they are then rotated to match the axes of the reference mesh. Since the curvature of the blade in the longest and middle axes tends to introduce estimation errors, we align the models first along the shortest axis, which provides the most accurate estimate due to its shape consistency.

## 8.2. Dataset Visualization

We present the mesh dataset used for generating demonstrations for the teapot task in Figure 18.

### 8.3. Correspondence Prediction Visualization



## 9. Data Generation Details

Given that the generated trajectories may vary in length from the original ones, we sample the goal object point cloud from random timestamps (excluding the skill segment) of the source demonstration to serve as the goal object point cloud for the new demonstration. To preserve the visual authenticity of the occlusions that occur when the object interacts with the goal object during the skill

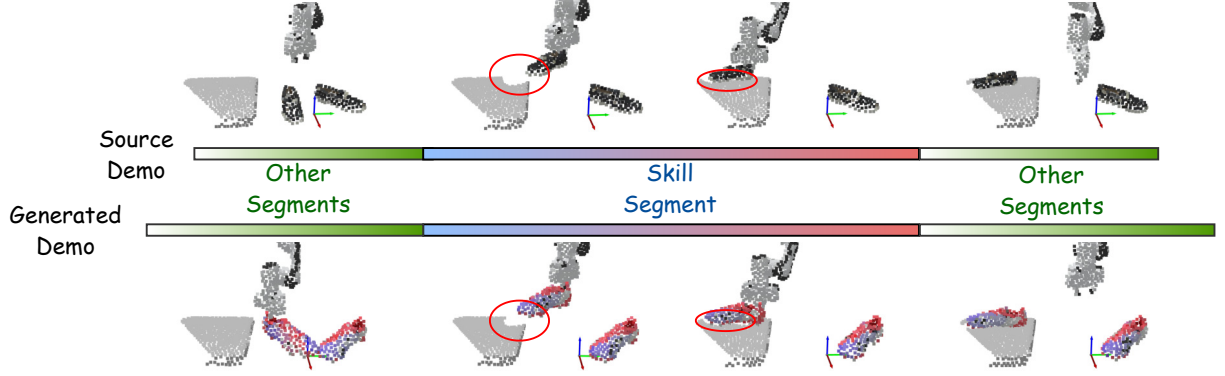


Figure 17. We preserve the occlusions of the goal object during skill segment in our point cloud generation process.

segment, the goal object point cloud in this phase is sequentially replayed from the corresponding segment of the source demonstration. The overall process can be expressed as follows:

$$\begin{cases} \text{Assemble}(\mathcal{O}_{\text{real}}[i], \mathcal{O}_{\text{sim}}[t]), & \text{if } t \notin \mathcal{T}'_s \\ \text{Assemble}(\mathcal{O}_{\text{real}}[t - t'_s + t_s], \mathcal{O}_{\text{sim}}[t]), & \text{if } t \in \mathcal{T}'_s \end{cases}$$

where  $t_s$  and  $t'_s$  are the start timesteps of  $\tau_s$  and  $\tau'_s$ , respectively, and  $i \in \mathcal{U}(0, T')$  denotes uniform sampling. The Assemble operator represents the process of concatenating the point cloud of the robot arm and the new meshes (from simulation) with the goal object point cloud (from the real world), under the constraint that the point count per frame remains consistent with the source demonstration. A visualization of the point cloud generation strategy is presented in Figure 17.

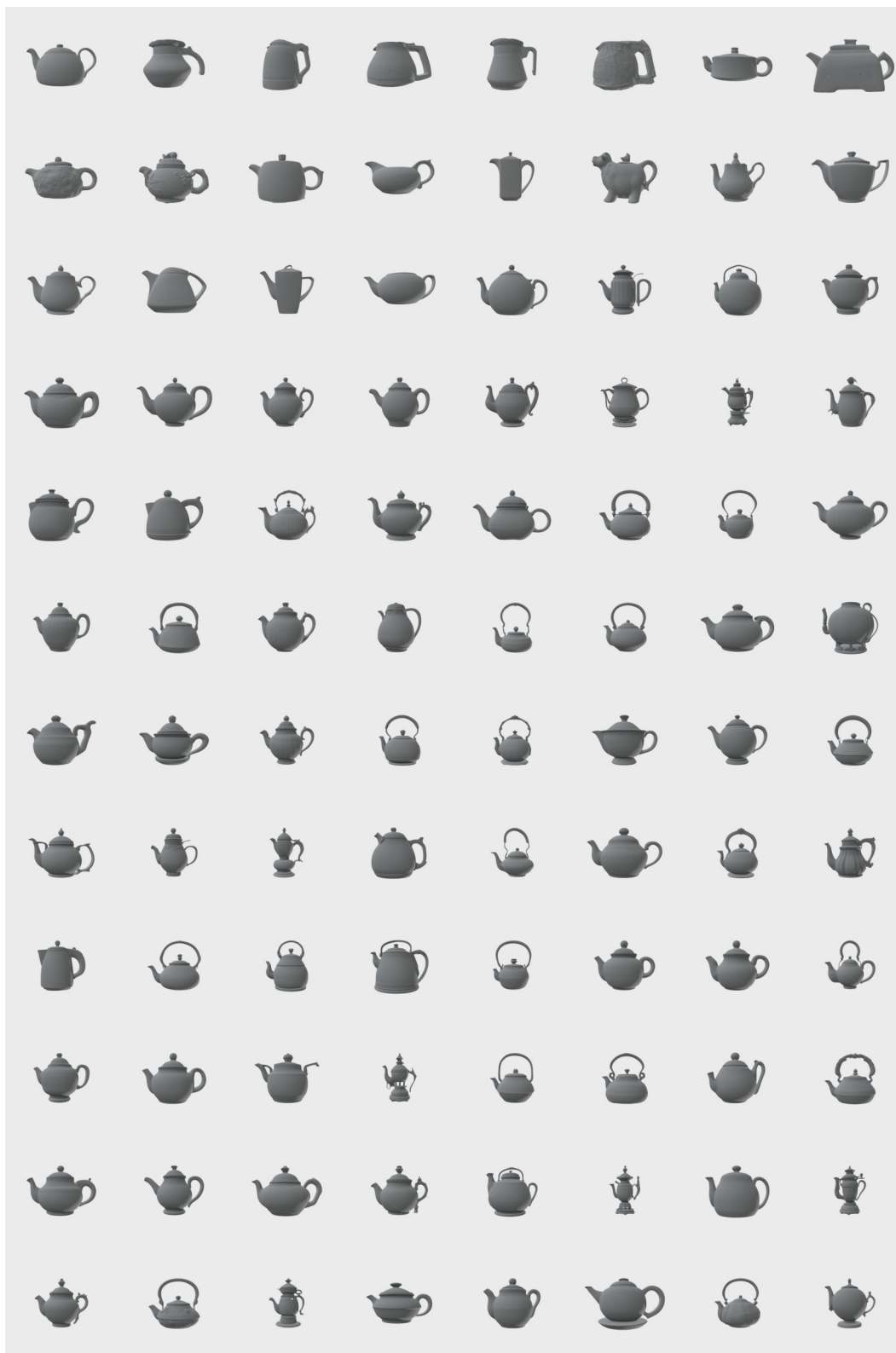


Figure 18. Part of the teapot meshes used for demonstration generation