Constraint-Preserving Data Generation for One-Shot Visuomotor Policy Generalization

Anonymous Author(s)
Affiliation
Address
email

Abstract: Large-scale demonstration data has powered key breakthroughs in robot manipulation, but collecting that data remains costly and time-consuming. To this end, we present Constraint-Preserving Data Generation (CP-Gen), a method that uses a single expert trajectory to generate robot demonstrations containing novel object geometries and poses. These generated demonstrations are used to train closed-loop visuomotor policies that transfer zero-shot to the real world. Similar to prior data-generation work focused on pose variations, CP-Gen first decomposes expert demonstrations into free-space motions and robot skills. Unlike prior work, we achieve geometry-aware data generation by formulating robot skills as keypoint-trajectory constraints: keypoints on the robot or grasped object must track a reference trajectory defined relative to a task-relevant object. To generate a new demonstration, CP-Gen samples pose and geometry transforms for each task-relevant object, then applies these transforms to the object and its associated keypoints or keypoint trajectories. We optimize robot joint configurations so that the keypoints on the robot or grasped object track the transformed keypoint trajectory, and then motion plan a collision-free path to the first optimized joint configuration. Using demonstrations generated by CP-Gen, we train visuomotor policies that generalize across variations in object geometries and poses. Experiments on 16 simulation tasks and four real-world tasks, featuring multi-stage, non-prehensile and tight-tolerance manipulation, show that policies trained using our method achieve an average success rate of 77%, outperforming the best baseline which achieves an average success rate of 50%. Simulation and real-world videos can be found at: cpgen-anon.github.io.

Keywords: Imitation Learning, Data Generation, Robot Manipulation

1 Introduction

Teaching robots to act by imitating humans is one of the most powerful, yet costly, ideas in robotics. At its best, large-scale imitation learning has enabled robots to tie shoelaces, fix broken grippers, and even cook shrimp [1–4]. Further scaling promises to unlock broad generalization across complex manipulation tasks. Behind these successes, however, lies a steep cost: human labor, months of continuous robot operation, and heavy infrastructure demands. For example, ALOHA Unleashed [3] collected 26,000 demonstrations over 8 months across 10 robots and 35 operators, while DROID [5] gathered 76,000 real-world demonstrations over a year. As the field pushes forward, reducing the burden of demonstration collection will be a key challenge.

Automated data generation promises to reduce the manual effort required in data collection. One line of work [6–10] generates demonstrations for scenes with diverse object poses by collecting a handful of teleoperated demonstrations and algorithmically multiplying them using pose transformations and action replay. Conceptually, these works aim to enable data efficient spatial generalization by exploiting $\mathrm{SE}(3)$ equivariance of robot actions with respect to object poses [9]. However, because they rely on $\mathrm{SE}(3)$ transformations, these works cannot generate demonstrations that adapt to geometric variations (such as different aspect ratios) with a given object instance. For example,

Submitted to the 9th Conference on Robot Learning (CoRL 2025). Do not distribute.

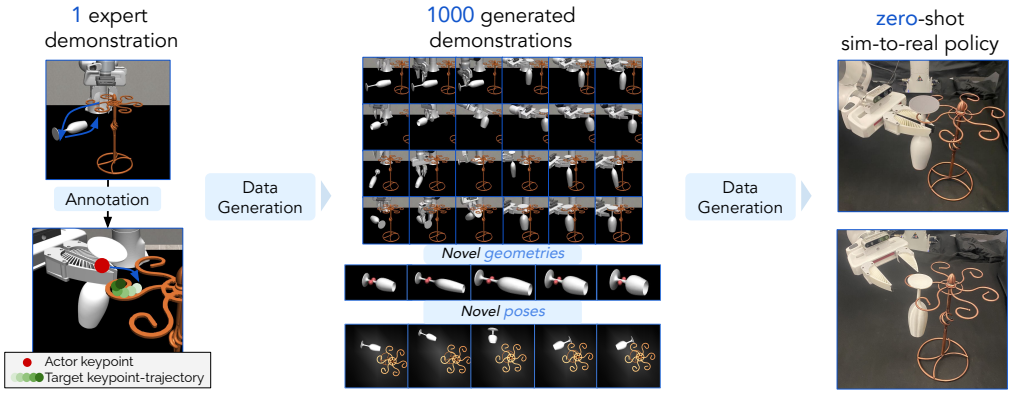


Figure 1: **CP-Gen** uses one expert demonstration and keypoint-trajectory constraints to generate diverse demonstrations involving novel object geometries and poses, enabling large-scale policy training and zero-shot sim-to-real transfer. For the Wine Glass Spiral Hanging task, the first keypoint-trajectory is anchored to the wine glass and constrains the motion of points on the end effector. The second keypoint-trajectory is anchored relative to the spiral-shaped wine glass rack, and constrains a keypoint on the wine glass stem to track a complex spiral motion.

robot actions that succeed in hanging a short, wide wine glass onto a rack may fail entirely for a tall, narrow one, even if both are aligned under the same pose transformation.

A complementary strategy to ease the burden on manual data collection is to increase learning efficiency by embedding structure—specifically, equivariance—into policy architectures [11–23]. One line of work focuses on open-loop manipulation, exploiting symmetry to generalize across different object poses [24–27]. More recent work has moved toward closed-loop policy learning, embedding equivariances directly into the policy architecture and directly predicting robot actions [11, 28–32]. However, embedding equivariance bottlenecks can limit policy expressiveness and be insufficient when dealing with scenes that involve multiple interacting objects or symmetry-breaking variations.

In this work, we propose **Constraint-Preserving Data Generation** (**CP-Gen**), a geometry-aware data generation method that takes a single expert demonstration and generates new demonstrations with both novel object poses and novel geometries. With CP-Gen, we aim to achieve the benefits of equivariant policy architectures without the bottleneck of embedding structure directly into policies.

Our key insight is to bring geometry awareness into the data generation process through a **keypoint-trajectory constraint** formulation. Starting from a *single* expert demonstration, we segment the trajectory into *free-space motions* (segments that can be replaced with point-to-point collision-free motion planning) and *robot skills* (segments requiring some interaction with task-relevant objects that cannot be completed through pure collision-free motion planning). We then formulate each skill as a keypoint-trajectory constraint: selected robot or grasped-object keypoints must track a reference trajectory defined in the frame of a task-relevant object. For example, in the *Wine Glass Spiral Hanging* task (Figure 1), the grasping skill is formulated as a trajectory-tracking constraint between keypoints on the gripper tips and a target trajectory in the wine-glass frame, where the target trajectory includes the pre-grasp segment up to the grasp pose. The subsequent *spiral* insertion phase introduces a second constraint: keypoints on the glass stem must track a spiral-like trajectory defined relative to the rack. By anchoring keypoints to the reference frame of task-relevant objects, we can sample new object geometries and poses, adapt the keypoint-trajectory constraints according to the transforms applied to the original object geometry, and generate thousands of demonstrations covering diverse shapes and geometries. Using these demonstrations, we can train a visuomotor policy that naturally generalizes robustly across changes in object pose and geometry, without requiring additional human demonstrations.

Our contributions are threefold. First, we propose CP-Gen, a data generation method that produces robot demonstrations from a single expert demonstration using a keypoint-trajectory constraint formulation. Unlike prior work, and due to this formulation, CP-Gen generates data in a geometry aware manner. Second, we validate our keypoint-trajectory data generation formulation on the MimicGen simulation benchmark. We achieve state-of-the-art performance (85% vs. 63%) on the default

task settings, as well as on our proposed simulation benchmark featuring novel object geometries (70% vs. 37%). Third, we demonstrate successful zero-shot sim-to-real on a set of four real-world tasks featuring multi-stage, non-prehensile and tight-tolerance manipulation.

2 Related Work

Data Generation for Robotics. One approach to addressing data scarcity in robotics is to generate new demonstrations from teleoperated source data [6–8, 10, 33]. MimicGen [6] generates demonstrations by resetting object poses and replaying pose-transformed versions of the source trajectory. IntervenGen [33] extends MimicGen with human interventions mid-trajectory to correct failures, while DexMimicGen [8] adapts MimicGen for bimanual and dexterous settings. SkillGen [7] segments demonstrations into skills and motions, alternating between learned policies and motion planning to improve robustness. At the core of these methods is a relative-pose formulation and the exploitation of $\mathrm{SE}(3)$-equivariance [9]: given an $\mathrm{SE}(3)$ transformation applied to an object, they apply the same $\mathrm{SE}(3)$ transformation to robot actions to replicate the original effect. As a result, they particularly struggle to generate demonstrations with new object geometries for tasks requiring tight-tolerance manipulation. CP-Gen addresses this limitation by introducing a keypoint-trajectory constraint formulation, representing skills in terms of robot or grasped-object keypoints tracking a reference trajectory. This formulation enables demonstration generation not only across novel object poses, but also across significantly different object geometries.

Equivariance for Robotic Manipulation. Another strategy to improve generalization is to embed geometric structure into policy inference and architecture. Open-loop methods design $\mathrm{SE}(2)$ or $\mathrm{SE}(3)$ equivariant feature representations on point clouds or images [11–23], achieving strong sample efficiency but often relying on expensive inference-time optimization [24–27] and lacking reactivity to dynamic changes. To address these limitations, recent work has moved toward closed-loop policies, beginning with planar $\mathrm{SO}(2)$ symmetries and extending to full $\mathrm{SIM}(3)$ equivariance [11, 28–32], where $\mathrm{SIM}(3)$ captures rotation, translation, and uniform scaling transformations. While $\mathrm{SIM}(3)$-equivariant policies enable more flexible closed-loop control compared to $\mathrm{SE}(3)$-equivariant policies, they still fundamentally assume that object geometry variations can be captured by uniform scaling, and for larger geometry variations, they rely on policy generalization. These closed-loop equivariance works [30–32] also tend to focus on single-object to robot equivariance, instead of the multi-object scenario. In CP-Gen, rather than using architectural equivariance, we leverage data generation to generalize to both novel poses and arbitrary geometry variations via a keypoint-trajectory constraint formulation.

3 Problem Setting

We are given a single expert demonstration in the form of a trajectory $\tau_{\text{src}} = \{o_t, a_t\}_{t=1}^{H}$ consisting of H observations o, and H actions a. Actions $a_t = \{a_{\text{eef}}, a_{\text{grip}}\}$ contain end-effector poses $a_{\text{eef}} \in \mathrm{SE}(3)$ along with gripper actions a_{grip} for a given manipulation task. Observations $o_t = \{I_{\text{hand}}, I_{\text{3pv}}, q_t\}$ at each time step consist of a wrist-mounted camera image I_{hand}, a third-person view camera image I_{3pv}, and robot proprioception data q_t. We aim to train a visuomotor policy $\pi_\theta(a|o)$ mapping observations $o \in O$ to actions $a \in A$.

Similar to prior data generation methods using source demonstrations [6, 7, 33], we assume that an expert demonstration τ_{src} can be decomposed into a sequence of alternating free-space motion segments $\tau_{\text{motion}}^{(i)}$ and robot skill segments $\tau_{\text{skill}}^{(i)}$: $\tau_{\text{src}} = [\tau_{\text{motion}}^{(1)}, \tau_{\text{skill}}^{(1)}, \tau_{\text{motion}}^{(2)}, \tau_{\text{skill}}^{(2)}, \ldots]$. For each skill segment, we additionally assume access to a set of annotated keypoints on the robot gripper or a grasped object that are relevant to the completion of the skill. We refer to these as *actor keypoints* and denote them by $\mathscr{K}_{\text{actor}}(t) = \{k_1^{\text{actor}}(t), \ldots, k_N^{\text{actor}}(t)\}$, where each keypoint $k_i^{\text{actor}}(t) \in \mathbb{R}^3$ is defined in the local frame of the robot gripper or the grasped object at time t. Although in this work we manually selected these keypoints, in future work, keypoints can be predicted using vision-language models [34–36], or inferred via task-specific keypoint detectors [37–40].

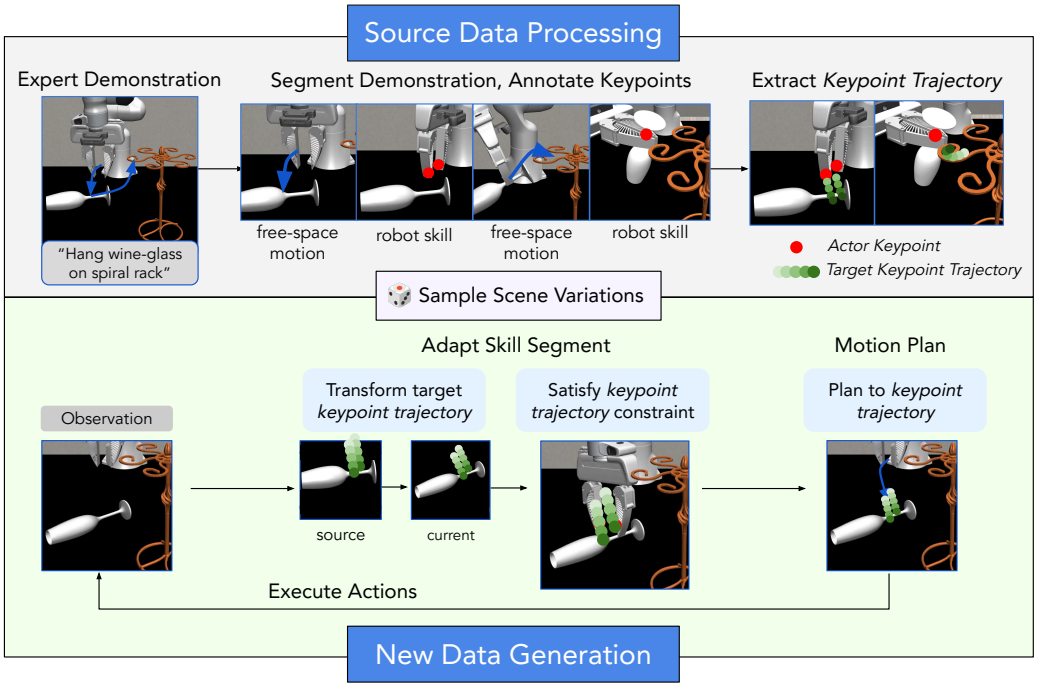


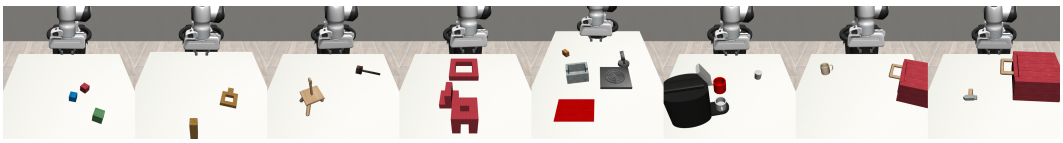
Figure 2: **CP-Gen Method**. In the *Source Data Processing* stage (top), starting from an expert demonstration τ_{src}, we (a) segment the trajectory into free-space motion and skill segments, (b) annotate keypoints on the robot or grasped object ● (dubbed *actor* keypoints), and (c) convert each skill segment into a *keypoint-trajectory constraint* by extracting a keypoint-trajectory expressed in the frame of a task-relevant object. After processing the source demonstration, in the *New Data Generation* stage (bottom), we *sample scene variations* by applying geometry and pose transforms to every task-relevant object. For each sampled scene, we (a) adapt our source skill segment to the current observation by transforming the extracted target keypoint-trajectory with the current object transforms and solving for robot configurations that satisfy the updated keypoint-trajectory constraint, and (b) motion plan a collision-free path that from the current robot joint configuration to the first configuration in the solved configuration trajectory. The process iterates over all segments to generate a demonstration for the new scene, which can subsequently be used to train a visuomotor policy $\pi_\theta(a \mid o)$.

4 CP-Gen: Constraint-Preserving Data Generation

We present **Constraint-Preserving Data Generation (CP-Gen)**, a data generation method that enables one-shot visual imitation learning by adapting a source demonstration trajectory to scenes with novel object geometries and poses (see Figure 2). Our approach begins by decomposing the original demonstration trajectory τ_{src} into a sequence of skill segments τ_{skill} interleaved with collision-free motion segments τ_{motion}, such that $\tau_{\text{src}} = [\tau_{\text{motion}}, \tau_{\text{skill}}, \ldots, \tau_{\text{motion}}, \tau_{\text{skill}}]$. Each skill segment τ_{skill} is formulated as a *keypoint-trajectory constraint*: keypoints on the robot or grasped object, *actor keypoints* ${}^{O}\mathcal{K}_{\text{actor}}(t)$, must track a reference trajectory defined relative to a task-relevant object. This reference trajectory is represented as a time-varying sequence of *target keypoints* ${}^{O}\mathcal{K}_{\text{target}}(t)$ expressed in the local frame of the object O. We extract this reference keypoint-trajectory (Section 4.1) by transforming the actor keypoints into the object frame at each timestep. We assume that adapting and preserving this keypoint-trajectory constraint to a new scene leads to successful execution. To generate a new demonstration in a novel scene, we first sample *pose* and *geometry* transforms for each task-relevant object. We then apply the sampled transformations to the objects (and their associated keypoint trajectories) to produce a new scene reset. For the new scene, we iterate through two phases of skill segment adaptation (Section 4.2) and motion planning (Section 4.3). Finally, after filtering out any failed demonstrations using a success detector (which we assume exists for each task), we use imitation learning to train a visuomotor policy on the generated demonstrations.

4.1 Keypoint-Trajectory Constraint Extraction

To extract the target keypoint-trajectory for a skill segment τ_{skill} from the original demonstration τ_{src}, we transform the actor keypoints ${}^{A}\mathcal{K}_{\text{actor}} = \{{}^{A}k_1, \ldots, {}^{A}k_N\}$, defined in the actor's local frame A



Property	Stack Three	Square	Threading	Assembly	Kitchen	Coffee	Mug Cleanup	Hammer Cleanup
Object(s)	block1, block2, block3	nut, peg	needle, tripod	base, piece_1, piece_2	bread, pot, stove, button	pod, machine	drawer, object	hammer, drawer
Scale Range	[0.6,1.4]	[1.0,1.0]	[0.9,1.1]	[0.7,1.3]	[0.7,1.4]	[0.6,1.1]	[0.7,1.3]	[0.7,1.2]

Figure 3: **Simulation Tasks and Geometry Generalization Variants**. *Top*: Tasks from the MimicGen benchmark [6] and object geometries sampled from our proposed *Geometry Generalization* task variants. *Bottom*: Uniform scale sampling ranges applied in *Geometry Generalization* task variants.

(robot gripper or grasped object), into the task-relevant object frame O via a frame transform from actor to world frame, then world to task-relevant object frame:

$$ {}^{O}k_i(t) = \underbrace{{}^{O}T_W(t)}_{\text{object pose}^{-1}} \cdot \underbrace{{}^{W}T_A(t)}_{\text{actor pose}} \cdot {}^{A}k_i, \tag{1} $$

Applying Eq. 1 across all t yields target keypoint-trajectory: ${}^{O}\mathcal{K}_{\text{target}}(t) = \{{}^{O}k_1(t), \ldots, {}^{O}k_N(t)\}$.

4.2 Skill Segment Adaptation

Given a skill segment τ_{skill} and a current scene observation, our goal is to adapt the original keypoint-trajectory constraint to new task-relevant object geometries and poses. Doing so requires updating both the actor keypoints and the target keypoint-trajectory.

Update Keypoint-Trajectory Constraint for New Geometry. Suppose that the task-relevant object undergoes a geometric transformation $\mathbf{X}$ (e.g., non-uniform scaling) in its own local frame O. Let the original local-frame target keypoint-trajectory be ${}^{O}\mathcal{K}_{\text{target}}(t) = \{{}^{O}k_1(t), \ldots, {}^{O}k_N(t)\}$. We therefore apply $\mathbf{X}$ to transform the target keypoint-trajectory:

$$ {}^{O}k'_i(t) = \mathbf{X} \cdot {}^{O}k_i(t), \quad \forall t. $$

If the actor keypoints ${}^{A}\mathcal{K}_{\text{actor}}(t) = \{{}^{A}k_1(t), \ldots, {}^{A}k_N(t)\}$ are anchored to a grasped object that has been geometrically transformed, we similarly apply the object's local-frame geometric transformation $\mathbf{X}_A$ to the actor keypoints: ${}^{A}k'_i = \mathbf{X}_A \cdot {}^{A}k_i$. These updated keypoints ${}^{A}k'_i$ are then used in the optimization process to match the newly transformed target trajectory.

Solve for Robot Configuration via Keypoint Matching. Given the updated actor keypoints ${}^{A}k'_i$ (in the actor's local frame) and the updated target keypoint-trajectory ${}^{O}k'_i(t)$ (in the object's local frame), our goal is to solve for the robot joint configuration q_t^* at each timestep t such that the transformed actor keypoints match the target keypoints in the world frame.

First, we compute the world-frame position of each actor keypoint (recall that actor keypoints are defined on the robot or a grasped object, and that we assume a fixed end-effector to grasped object frame transformation) via forward kinematics: ${}^{W}k'_i(q) = f_{\text{FK}}(q, {}^{A}k'_i)$. Next, we compute the world-frame position of each target keypoint by transforming the updated local-frame keypoints using the current object pose ${}^{W}T_O(t)$ relative to the world frame: ${}^{W}k_i^{\text{target}}(t) = {}^{W}T_O(t) \cdot {}^{O}k'_i(t)$.

Method	StackThree	Square	Thread	Assembly	Kitchen	Coffee	Mug	Hammer	Avg
CP-Gen [RGB]	0.82±0.00	**0.87±0.03**	**0.87±0.03**	**0.85±0.03**	0.96±0.01	**1.00±0.00**	**0.86±0.03**	**0.8±0.03**	**0.88**
MimicGen [RGB]	**0.98±0.01**	0.72±0.04	0.38±0.04	0.32±0.04	**0.98±0.01**	0.58±0.04	0.78±0.03	0.64±0.04	0.67
CP-Gen [D+S]	**0.56±0.00**	**0.91±0.02**	**0.85±0.03**	**0.79±0.03**	**0.92±0.02**	**0.99±0.01**	**0.72±0.00**	**0.79±0.03**	**0.82**
MimicGen [D+S]	0.52±0.04	0.76±0.03	0.44±0.04	0.40±0.04	0.94±0.02	0.56±0.04	0.46±0.04	0.64±0.04	0.59

Method	StackThreeG	SquareG	ThreadG	AssemblyG	KitchenG	CoffeeG	MugG	HammerG	Avg
CP-Gen [RGB]	**0.68±0.04**	**0.88±0.03**	**0.79±0.03**	**0.55±0.04**	**0.83±0.03**	**0.58±0.04**	**0.82±0.03**	**0.67±0.04**	**0.73**
MimicGen [RGB]	0.60±0.04	0.42±0.04	0.10±0.02	0.16±0.03	0.74±0.04	0.00±0.00	0.38±0.04	0.40±0.04	0.35
CP-Gen [D+S]	**0.36±0.00**	**0.83±0.03**	**0.73±0.04**	**0.79±0.03**	**0.86±0.03**	**0.63±0.04**	**0.76±0.04**	**0.74±0.04**	**0.67**
MimicGen [D+S]	0.22±0.03	0.64±0.04	0.05±0.02	0.38±0.04	0.84±0.03	0.00±0.00	0.50±0.04	0.45±0.04	0.39

Table 1: **CP-Gen achieves state-of-the-art results on the MimicGen simulation benchmark**. On the original MimicGen benchmark (default task variants), CP-Gen achieves an average success rate of 88% compared to MimicGen's 67%. On our custom benchmark containing *Geometry Generalization* task variants which feature novel object geometries (denoted as TaskG), CP-Gen achieves an average success rate of 70%, outperforming MimicGen's 37% by a margin of 33%. These results highlight CP-Gen's strong generalization not only to pose variations but also to challenging geometric variations. Bolded numbers indicate the best-performing method within each modality group.

Finally, we solve the following optimization problem for each timestep of the robot skill segment:

$$q_t^* = \arg\min_q \underbrace{\sum_{i=1}^{N} \left\| f_{\text{FK}}(q, {}^A k_i') - \left({}^W T_O(t) \cdot {}^O k_i'(t) \right) \right\|_2^2}_{\text{match keypoints}} + \underbrace{\lambda \left\| q - q_{t-1}^* \right\|_2^2}_{\text{temporal smoothness penalty}} \tag{2}$$

where $\lambda \geq 0$ trades off keypoint matching and temporal smoothness in joint space. This optimization encourages a robot trajectory where the transformed actor keypoints ${}^A k_i'$ track the updated keypoint-trajectory ${}^O k_i'(t)$ in the world frame under new object geometries and object poses. We solve the optimization problem using the L-BFGS-B [41] gradient-based optimizer from SciPy [42].

4.3 Motion Planning

After obtaining the optimized robot configurations $Q^* = [q_1^*, \ldots, q_H^*]$ from the skill segment adaptation described above, we plan a collision-free trajectory from the robot's current joint configuration to the first joint configuration of the upcoming skill segment q_1^*. We query a collision-free motion planner [43, 44] with the robot's start configuration, the first joint configuration of the upcoming skill, and provide collision geometries extracted directly from our simulation environment. As our action space uses end-effector poses, we use an inverse kinematics controller to track the generated trajectory.

5 Experiments

Through our experiments, we wish to answer the following questions:

Q1: Can a policy trained on data generated from a *single demonstration* using CP-Gen generalize to *unseen object geometries and poses*?

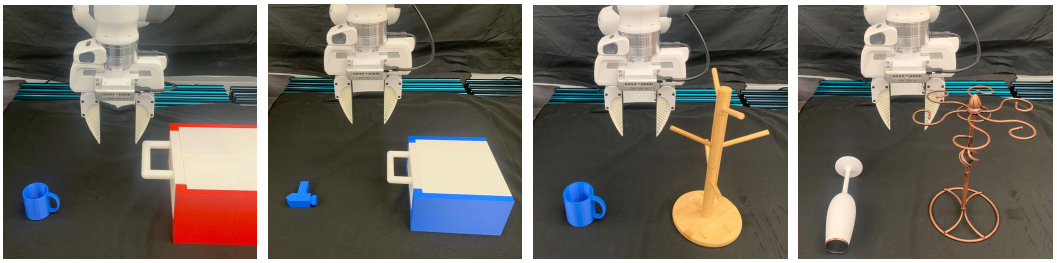
Q2: How does CP-Gen's *keypoint-trajectory formulation* affect *data generation* success rates on tasks with *novel object geometries*?

Q3: How does training policies on *diverse object geometries* affect performance in various settings?

Q4: Can *pixel-based policies* trained with CP-Gen in simulation *transfer zero-shot* to the real world?

5.1 Simulation Experiments

Tasks. We evaluate all methods on single-arm Franka Panda tasks from the MimicGen [6] benchmark. We evaluate under two environment reset distributions. The default, fixed geometry distribution varies only object poses, and corresponds to distribution "D1" in the MimicGen benchmark. The custom *Geometry Generalization* distribution introduces shape variations via non-uniform scaling and retains the same pose reset range as the default distribution (Figure 3).



(a) Mug Cleanup (b) Hammer Cleanup (c) Mug Hanging (d) Wine Glass Spiral Hanging

Figure 4: **Real World Tasks**. We evaluate on four challenging real-world tasks and show that policies trained on CP-Gen generated simulation datasets can transfer zero-shot to the real world.

Evaluation Metrics. We report the average success rate achieved by a particular policy network on the given task, along with the standard error of this metric. For computing standard errors, we use the best checkpoint rolled out for three seeds of 50 trials each.

Baselines. We compare CP-Gen to MimicGen [6], a data generation method that produces demonstrations in which object poses differ from those seen in source demonstrations. MimicGen first parses each source demonstration into object-centric subtask segments. To generate a new demonstration, it selects a reference segment, transforms it to match the new scene's object pose, runs linear interpolation to reach the start of the transformed reference segment and executes the resulting end-effector trajectory using a controller. Finally, MimicGen filters out failed runs using a custom success checker, in the same way as CP-Gen. The authors additionally add Gaussian action noise to increase state coverage at the cost of the number of successfully generated demonstrations. In our case, we use 0.01 noise by default, but set noise to 0 if the number of successfully generated demonstrations drops below 10% for a given task. We use MimicGen to generate 1000 successful demonstrations per task from 10 human demonstrations.

Policy Training. For each method and task, we train a Diffusion Policy [45], specifically the DDPM variant, on the corresponding generated dataset that contains 1000 successful demonstrations. We use the same success classifier for both CP-Gen and MimicGen to filter for successful demonstrations. For policy inputs, we use two types of modalities from the third-person view and hand camera: (1) RGB images or (2) depth map and segmentation masks. We also provide proprioception information (end-effector pose and gripper width).

Results. From Table 1, we see that policies trained with data generated by CP-Gen (which takes a single source demonstration) outperform those trained using data generated using MimicGen across both observation input modalities. On the default task variants, we see that CP-Gen outperforms policies trained using MimicGen data (85% vs. 63%). We hypothesize that the policy performance drop is due to dataset quality, and hypothesize two reasons. First, CP-Gen uses a collision-free motion planner, and thus contains collision-avoidance behaviors, while MimicGen data does not have explicit collision-avoidance behaviors. Second, CP-Gen uses *actor keypoints* (keypoints on the robot or grasped object) during data generation which enables the robot to adapt actions based on how an object is grasped, while MimicGen only preserves the relative $\mathrm{SE}(3)$ transformation between the robot and a reference object. For example, on the Threading task, regardless of how the robot grasps the needle, MimicGen would generate the same end-effector actions for the robot in the threading phase based on the pose of the tripod object. In contrast, CP-Gen would adjust the robot's actions to align the actor keypoints on the needle tip with the tripod. On the *Geometry Generalization* task variants, CP-Gen's advantage is even more pronounced (70% vs. 37%). This finding is unsurprising, since CP-Gen generated demonstrations are adapted to the novel object geometries, while MimicGen does not consider changes to object geometry when generating demonstrations.

Method	Mug Cleanup	Hammer Cleanup	Mug Hanging	Wine Glass Spiral Hanging	**Average**
CP-Gen	**0.80 ± 0.13**	**0.80 ± 0.13**	**1.00 ± 0.00**	**0.70 ± 0.15**	**0.83**
MimicGen	0.20 ± 0.13	0.00 ± 0.00	0.80 ± 0.13	0.60 ± 0.16	0.40

Table 2: **CP-Gen policies successfully transfer zero-shot sim2real**. Policies take in depth and segmentation masks, and tasks feature both object-geometry and object pose variations.

(a) Data Generation			(b) Policy Evaluation		
Method	**Default**	**Geometry**	**Method**	**Default**	**Geometry**
MimicGen	0.58	0.23	CP-Gen (Default)	0.88	0.45
CP-Gen	0.89	0.62	CP-Gen (Geometry)	0.72	0.73

Table 3: **Ablations**. (a) CP-Gen outperforms MimicGen in terms of data generation success rate under both fixed (*Default*) and varied object geometry (*Geometry Generalization)* settings. (b) Using CP-Gen to generate diverse object geometries, denoted by *CP-Gen (Geometry)*, substantially improves policy generalization to novel geometries (73% vs. 45%). Furthermore, given a fixed training dataset budget of 1000 demonstrations, training and evaluating on the same distribution yields the best policy evaluation performance.

5.2 Real World Experiments

Tasks. We evaluate CP-Gen and baselines on four real-world manipulation tasks (Figure 4): (1) *Mug Cleanup*: Open a drawer, pick and place mug into a drawer, and close the drawer. (2) *Hammer Cleanup*: Same as *Mug Cleanup*, except with a small hammer, introducing grasping challenges. Cleanup tasks test multi-stage, non-prehensile manipulation involving articulated objects. (3) *Mug Hanging*: Pick and insert a mug onto a branch-like hook. This task tests constrained insertion through a small opening. (4) *Wine Glass Spiral Hanging*: Pick, reorient and hang a wine glass by threading the stem through a spiral-shaped rack. This task tests tight tolerance path following and orientation control. For each task, we manually construct a digital twin in simulation, generate 1000 successful demonstrations using a single source demonstration, and train a Diffusion Policy on the dataset. Policies are evaluated on two novel object geometries for each task.

Policy Inputs. To reduce the sim-to-real gap [46], we use depth maps and segmentation masks as input observations. Specifically, we use depth and segmentation masks for the third-person-view camera, and segmentation masks for the hand camera.

Results. From Table 2, we see that CP-Gen achieves strong zero-shot sim-to-real transfer on all four manipulation tasks. In contrast, MimicGen struggles with novel geometries. In the *Cleanup* tasks, failures arise from misalignment during drawer opening. In the *Hanging* tasks, failures are caused by suboptimal grasp positioning, which prevents proper insertion.

5.3 Ablations

We conduct a series of ablations to study two aspects of CP-Gen: the use of the keypoint-trajectory constraint formulation, and the effect of training on diverse object geometries. First, to assess the impact of the keypoint-trajectory constraint formulation, we compare data generation success rates between CP-Gen and MimicGen (which uses a relative pose data generation formulation) across both default and *Geometry Generalization* task variants on eight simulation tasks. For each task, we run 50 data generation trials. As shown in Table 3(a), CP-Gen achieves consistently higher success rates (89% vs. 58%, 62% vs. 23%), especially on the *Geometry Generalization* reset distribution, highlighting the value of the geometry-aware data generation that our keypoint-trajectory constraint formulation enables. Second, we evaluate how training on diverse object geometries affects policy generalization. In Table 3(b), we report policy success rates when training with and without geometry diversity, and evaluated in the *Default* and *Geometry Generalization* environment reset settings. As expected, training with diverse object geometries substantially improves policy generalization to novel geometries (73% vs. 45%). We also observe that training and evaluating on the same distribution yields the best evaluation performance (73% vs. 45% and 88% vs. 72%).

6 Conclusion

In this work, we propose CP-Gen, a data generation framework for generalizable visual imitation learning given just one expert demonstration. CP-Gen generates geometry-aware demonstrations using the insight that robot skills can be formulated as keypoint-trajectory constraints: keypoints on the robot or grasped object must track a reference trajectory defined relative to a task-relevant object. We demonstrate that CP-Gen's approach to data generation from just one source demonstration achieves state-of-the-art success rates on the MimicGen simulation benchmark, and outperforms baselines on a custom benchmark featuring diverse object geometries. Finally, we show successful zero-shot sim-to-real transfer of policies trained with CP-Gen data to challenging real-world tasks.

Limitations. CP-Gen requires manual annotation of robot skill segments and keypoints. The simulation environment and reward functions are also manually defined; future work could incorporate foundation model based success classifiers. The method assumes a fixed skill sequence, limiting task level generalization. Incorporating task level planning via foundation models [47–50] or task and motion planner methods [51–53] can further improve data generation success. CP-Gen is less effective for tasks with complex constraint sequences, such as dexterous in-hand manipulation, where methods such as reinforcement learning or model predictive control may be more suitable. Current results are limited to a single-arm Panda robot, though the method can be extended to multi-arm, mobile base, and legged systems [8]. CP-Gen does not handle the situation where one may want demonstrations generated for a specific object instance within the same category for which a source demonstration is available. For example, we may have a source demonstration for a mug with a square handle, but wish to generate demonstrations for a specific mug that might have a round handle. CP-Gen would need require the geometric transformation from the square handle mug to the round handle mug, which may be non-trivial to obtain. Future work may explore the use of neural descriptor fields [25] to assist in this setting.

References

[1] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, "Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware," in *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, Jul. 2023. DOI: 10.15607/RSS.2023.XIX.016.

[2] Z. Fu, T. Z. Zhao, and C. Finn, "Mobile aloha: Learning bimanual mobile manipulation with low-cost whole-body teleoperation," in *Conference on Robot Learning (CoRL)*, 2024.

[3] T. Z. Zhao *et al.*, "Aloha unleashed: A simple recipe for robot dexterity," *arXiv preprint arXiv:2410.13126*, 2024.

[4] J. Aldaco *et al.*, "Aloha 2: An enhanced low-cost hardware for bimanual teleoperation," *arXiv preprint arXiv:2405.02292*, 2024.

[5] A. Khazatsky *et al.*, "Droid: A large-scale in-the-wild robot manipulation dataset," 2024.

[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. R. Garrett, A. Mandlekar, B. Wen, and D. Fox, "Skillgen: Automated demonstration generation for efficient skill learning and deployment," in *8th Annual Conference on Robot Learning*, 2024. [Online]. Available: https://openreview.net/forum?id=YOFrRTDC6d.

[8] Z. Jiang *et al.*, "Dexmimicgen: Automated data generation for bimanual dexterous manipulation via imitation learning," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*, 2025.

[9] E. Ameperosa, J. A. Collins, M. Jain, and A. Garg, "Rocoda: Counterfactual data augmentation for data-efficient robot learning from demonstrations," *arXiv preprint arXiv:2411.16959*, 2024.

[10] Z. Xue, S. Deng, Z. Chen, Y. Wang, Z. Yuan, and H. Xu, "Demogen: Synthetic demonstration generation for data-efficient visuomotor policy learning," *arXiv preprint arXiv:2502.16932*, 2025.

[11] D. Wang, R. Walters, and R. Platt, "SO(2)-Equivariant Reinforcement Learning," in *International Conference on Learning Representations*, 2022.

[12] C. Kohler, A. S. Srikanth, E. Arora, and R. Platt, "Symmetric models for visual force policy learning," *arXiv preprint arXiv:2308.14670*, 2023.

[13] H. H. Nguyen, A. Baisero, D. Klee, D. Wang, R. Platt, and C. Amato, "Equivariant reinforcement learning under partial observability," in *Conference on Robot Learning*, PMLR, 2023, pp. 3309–3320.

[14] H. Nguyen, T. Kozuno, C. C. Beltran-Hernandez, and M. Hamaya, "Symmetry-aware reinforcement learning for robotic assembly under partial observability with a soft wrist," *arXiv preprint arXiv:2402.18002*, 2024.

[15] B. Eisner, Y. Yang, T. Davchev, M. Vecerik, J. Scholz, and D. Held, "Deep SE(3)-equivariant geometric reasoning for precise placement tasks," in *The Twelfth International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=2inBuwTyL2.

[16] D. Wang, R. Walters, X. Zhu, and R. Platt, "Equivariant Q Learning in Spatial Action Spaces," in *5th Annual Conference on Robot Learning*, 2021.

[17] H. Huang, D. Wang, A. Tangri, R. Walters, and R. Platt, "Leveraging Symmetries in Pick and Place," *The International Journal of Robotics Research*, 2023.

[18] A. Simeonov *et al.*, "SE(3)-Equivariant Relational Rearrangement with Neural Descriptor Fields," in *Conference on Robot Learning*, PMLR, 2023, pp. 835–846.

[19] C. Pan, B. Okorn, H. Zhang, B. Eisner, and D. Held, "TAX-Pose: Task-Specific Cross-Pose Estimation for Robot Manipulation," in *Conference on Robot Learning*, PMLR, 2023, pp. 1783–1792.

[20] H. Huang, D. Wang, X. Zhu, R. Walters, and R. Platt, "Edge Grasp Network: A Graph-Based SE(3)-invariant Approach to Grasp Detection," in *International Conference on Robotics and Automation (ICRA)*, 2023.

[21] S. Liu *et al.*, "Continual Vision-based Reinforcement Learning with Group Symmetries," in *Conference on Robot Learning*, PMLR, 2023, pp. 222–240.

[22] M. Jia *et al.*, "Seil: Simulation-augmented equivariant imitation learning," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2023, pp. 1845–1851.

[23] S. Kim, B. Lim, Y. Lee, and F. C. Park, "Se (2)-equivariant pushing dynamics models for tabletop object manipulations," in *Conference on Robot Learning*, PMLR, 2023, pp. 427–436.

[24] H. Huang, D. Wang, R. Walters, and R. Platt, "Equivariant Transporter Network," in *Robotics: Science and Systems*, 2022.

[25] A. Simeonov *et al.*, "Neural descriptor fields: Se (3)-equivariant object representations for manipulation," in *ICRA*, 2022.

[26] H. Ryu, H.-i. Lee, J.-H. Lee, and J. Choi, "Equivariant Descriptor Fields: SE(3)-Equivariant Energy-Based Models for End-to-End Visual Robotic Manipulation Learning," in *The Eleventh International Conference on Learning Representations*, 2023.

[27] H. Huang, O. Howell, X. Zhu, D. Wang, R. Walters, and R. Platt, "Fourier Transporter: Bi-Equivariant Robotic Manipulation in 3D," *arXiv preprint arXiv:2401.12046*, 2024.

[28] X. Zhu, D. Wang, O. Biza, G. Su, R. Walters, and R. Platt, "Sample Efficient Grasp Learning Using Equivariant Models," in *Robotics: Science and Systems*, 2022.

[29] X. Zhu, D. Wang, G. Su, O. Biza, R. Walters, and R. Platt, "On robot grasp learning using equivariant models," *Autonomous Robots*, vol. 47, no. 8, pp. 1175–1193, 2023.

[30] D. Wang *et al.*, "Equivariant diffusion policy," in *8th Annual Conference on Robot Learning*, 2024. [Online]. Available: https://openreview.net/forum?id=wD2kUVLT1g.

[31] J. Yang, Z.-a. Cao, C. Deng, R. Antonova, S. Song, and J. Bohg, "Equibot: Sim(3)-equivariant diffusion policy for generalizable and data efficient learning," in *8th Annual Conference on Robot Learning*, 2024.

[32] J. Yang, C. Deng, J. Wu, R. Antonova, L. Guibas, and J. Bohg, "Equivact: Sim(3)-equivariant visuomotor policies beyond rigid object manipulation," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2024, pp. 9249–9255.

[33] R. Hoque, A. Mandlekar, C. Garrett, K. Goldberg, and D. Fox, *Intervengen: Interventional data generation for robust and data-efficient robot imitation learning*, 2024. arXiv: 2405.01472 [cs.RO].

[34] 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.

[35] M. Oquab *et al.*, "Dinov2: Learning robust visual features without supervision," *arXiv preprint arXiv:2304.07193*, 2023.

[36] OpenAI, *Gpt-4 technical report*, 2023. arXiv: 2303.08774 [cs.CL].

[37] L. Manuelli, W. Gao, P. Florence, and R. Tedrake, "Kpam: Keypoint affordances for category-level robotic manipulation," in *ISRR*, 2019.

[38] Z. Qin, K. Fang, Y. Zhu, F.-F. Li, and S. Savarese, "Keto: Learning keypoint representations for tool manipulation," in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2020, pp. 7278–7285.

[39] P. Sundaresan *et al.*, "Learning rope manipulation policies using dense object descriptors trained on synthetic depth data," in *ICRA*, 2020.

[40] L. Manuelli, Y. Li, P. Florence, and R. Tedrake, "Keypoints into the future: Self-supervised correspondence in model-based reinforcement learning," *arXiv preprint arXiv:2009.05085*, 2020.

[41] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, "A limited memory algorithm for bound constrained optimization," *SIAM Journal on Scientific Computing*, vol. 16, no. 5, pp. 1190–1208, 1995. DOI: 10.1137/0916069. eprint: https://doi.org/10.1137/0916069. [Online]. Available: https://doi.org/10.1137/0916069.

[42] P. Virtanen *et al.*, "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python," *Nature Methods*, vol. 17, pp. 261–272, 2020. DOI: 10.1038/s41592-019-0686-2.

[43] B. Sundaralingam *et al.*, *Curobo: Parallelized collision-free minimum-jerk robot motion generation*, 2023. arXiv: 2310.17274 [cs.RO].

[44] W. Thomason, Z. Kingston, and L. E. Kavraki, "Motions in microseconds via vectorized sampling-based planning," in *IEEE International Conference on Robotics and Automation*, 2024. [Online]. Available: http://arxiv.org/abs/2309.14545.

[45] C. Chi *et al.*, "Diffusion policy: Visuomotor policy learning via action diffusion," in *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[46] M. Dalal *et al.*, "Local policies enable zero-shot long-horizon manipulation," *International Conference of Robotics and Automation*, 2025.

[47] M. Ahn *et al.*, "Do as i can, not as i say: Grounding language in robotic affordances," *arXiv preprint arXiv:2204.01691*, 2022.

[48] W. Huang *et al.*, "Inner monologue: Embodied reasoning through planning with language models," *arXiv preprint arXiv:2207.05608*, 2022.

[49] K. Lin, C. Agia, T. Migimatsu, M. Pavone, and J. Bohg, "Text2motion: From natural language instructions to feasible plans," *Autonomous Robots*, vol. 47, no. 8, pp. 1345–1365, 2023.

[50] B. Liu *et al.*, "Llm+ p: Empowering large language models with optimal planning proficiency," *arXiv preprint arXiv:2304.11477*, 2023.

[51] L. P. Kaelbling and T. Lozano-Pérez, "Hierarchical task and motion planning in the now," in *2011 IEEE International Conference on Robotics and Automation*, 2011, pp. 1470–1477. DOI: 10.1109/ICRA.2011.5980391.

[52] C. R. Garrett, T. Lozano-Pérez, and L. P. Kaelbling, "Pddlstream: Integrating symbolic planners and blackbox samplers via optimistic adaptive planning," in *Proceedings of the International Conference on Automated Planning and Scheduling*, vol. 30, 2020, pp. 440–448.

[53] C. R. Garrett *et al.*, "Integrated task and motion planning," *Annual review of control, robotics, and autonomous systems*, vol. 4, pp. 265–293, 2021.