EquiBot: SIM(3)-Equivariant Diffusion Policy for Generalizable and Data Efficient Learning

Jingyun Yang*, **Zi-ang Cao***, **Congyue Deng**,
Rika Antonova, **Shuran Song**, **Jeannette Bohg**
Stanford University
{jingyuny, ziangcao, congyue, rika.antonova, shuran, bohg}@stanford.edu

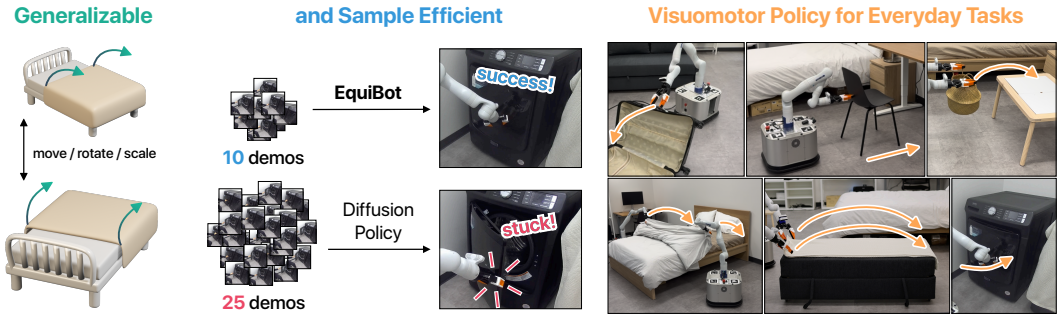

Figure 1: We propose a method for learning **generalizable** and **sample-efficient** visuomotor policies that can be applied to **everyday manipulation tasks**.

Abstract: Building effective imitation learning methods that enable robots to learn from limited data and still generalize across diverse real-world environments is a long-standing problem in robot learning. We propose EquiBot, a robust, data-efficient, and generalizable approach for robot manipulation task learning. Our approach combines SIM(3)-equivariant neural network architectures with diffusion models. This ensures that our learned policies are invariant to changes in scale, rotation, and translation, enhancing their applicability to unseen environments while retaining the benefits of diffusion-based policy learning such as multi-modality and robustness. We show in a suite of 6 simulation tasks that our proposed method reduces the data requirements and improves generalization to novel scenarios. In the real world, we show with in total 10 variations of 6 mobile manipulation tasks that our method can easily generalize to novel objects and scenes after learning from just 5 minutes of human demonstrations in each task.

Keywords: Imitation Learning, Equivariance, Data Efficiency

1 Introduction

The quest for fully autonomous robotic agents has often stumbled over the unpredictability and variability of real-world environments. While many existing visuomotor policies learned via imitation learning [1] are effective in controlled settings, they require a significant amount of data to train and even then struggle to adapt when evaluated with unfamiliar objects in unfamiliar environments. These limitations restrict the practical deployment of autonomous robots into the real world and amplify the need for policies that can learn from limited data yet perform robustly across diverse scenarios.

* These authors contributed equally.

In this work, we introduce EquiBot, an equivariant policy learning architecture based on diffusion models. With an equivariant neural network architecture, the model output is **guaranteed** to scale, translate, and rotate with the inputs, even if the model is not fully trained. This allows a learned policy to generalize to unseen scenarios that depart drastically from the scenarios in which it is trained. The equivariant architecture also brings data efficiency benefits, since an equivariant policy can infer how to react to object placements and poses that are in distribution but not sufficiently demonstrated, thus bringing better performance than a non-equivariant policy. Using a policy architecture based on diffusion models also offers robust policy learning performance, including support for idle actions and multi-modal behaviors. Thanks to the policy design, EquiBot can enable a wide range of household manipulation tasks from just a handful of single-view human demonstration videos. At test time, our method takes single-view scene point clouds and robot proprioception information as input, and outputs sequences of robot end-effector velocity actions as outputs.

We evaluate our method in both simulation and real-world experiments. In simulation, we evaluate in a suite of 6 tasks (box closing, cloth folding, object covering, push T, can pick-and-place, and square nut assembly). We show that compared to prior works in imitation learning and equivariant visuomotor policy learning, our method is both more data-efficient and generalizes better in unseen scenarios. In real robot experiments, we quantitatively test our method in 6 real-world mobile manipulation tasks in real bedroom and living room settings. Our experiments cover a wide range of everyday tasks, including pushing a chair towards a desk, closing the door of a laundry machine, folding towels, making the bed, and closing a suitcase. These tasks range from single-robot pick-and-place tasks to multi-robot tasks involving deformable or articulated objects. We show that our method can successfully perform the learned tasks with unseen objects and scenes from just 5 minutes of human demonstrations, outperforming competing baselines that rely on augmentations to achieve generalization or do not utilize a diffusion process to predict actions.

2 Related Work

Data-efficient imitation learning. Recent imitation learning approaches for imitation learning assume large quantities of data to be available for learning manipulation policies [2, 1]. To reduce the amount of data that policy training requires per task, some works [3, 4, 5, 6] formulate a one-shot or few-shot imitation learning setup where the policy is trained on multi-task demonstration data and can then output actions in a novel task after seeing one or more demonstrations as well as the current state. While this approach improves the data efficiency of policy learning per task, it requires the availability of multi-task data in a task domain. Some other works [7] achieve imitation learning from small demonstration datasets with sampling-based optimization methods like Bayesian Optimization, but these methods are often limited to small action spaces and open-loop settings. In contrast to prior works, we show that embedding equivariance into the policy architecture can effectively improve the data efficiency of the imitation learning algorithm, allowing a robust manipulation policy to be learned with just a handful of demonstrations.

Equivariance in robot manipulation. To help learned robot policies generalize to unseen environments and object placements, prior works [8, 9, 10] use data augmentation to improve cross-domain transfer of the learned policy. However, these methods increase training time significantly and do not guarantee generalization to visual appearances, object scales, and poses that are unseen in the training data distribution even after the augmentation. In contrast, utilizing equivariant representations in policy learning allows the learned policies to generalize to objects and initial conditions not previously seen at training time. Prior works have explored the use of equivariance in robot manipulation in several different setups [11, 12, 13, 14, 15, 16], but most of them either focus on only simple pick-and-place like tasks, do not support closed-loop policies, or do not support scale equivariance. Closely related to our work, [17] developed a SIM(3)-equivariant visuomotor policy learning method that can handle non-pick-and-place tasks with deformable and articulated objects. However, we show that this method displays unstable training performance. Our method combines SIM(3)-equivariant neural network architectures with Diffusion Policy and thus can be robustly trained and generalizes to unseen object appearance, initial states, scales, and poses. Similar to

our method, Chen et al. [18] also combines equivariance with diffusion policy, but it can only handle SO(2)-equivariance with simple 2D trajectories, which is insufficient for most 3D manipulation tasks.

3 Method

In this section, we describe the design of EquiBot. Our method builds upon recent advances that use a diffusion process to represent visuomotor policies, and incorporates a series of important modifications so the resulting architecture is equivariant to 3D translations, rotations, and uniform scaling. In Section 3.1, we introduce relevant concepts related to Diffusion Policy and equivariance; in Section 3.2, we describe in detail the design of our architecture; in Section 3.3, we describe important details related to the implementation of our method.

3.1 Preliminaries

Problem setup. We assume an imitation learning setup, where our method receives a demonstration dataset $\mathcal{D} = \{\tau^n\}_{n=1}^N$, which consists of N demonstration trajectories τ^n. Each demonstration trajectory consists of sequences of observation-action pairs $(\mathbf{O}_t, \mathbf{A}_t)$. The goal of the policy is to learn a mapping π from observation $\mathbf{O}_t$ to either the next action $\mathbf{A}_t$ or the next set of actions $\mathbf{A}_{t:t+T_p}$, where T_p is the prediction horizon. At evaluation time, the policy receives state $\mathbf{O}_t$ and predicts the next one or more actions to be executed in the environment. In this work, we assume the observations $\mathbf{O}_t = (\mathbf{X}_t, \mathbf{S}_t)$ is composed of the scene point cloud $\mathbf{X}_t$ and robot proprioception $\mathbf{S}_t$.

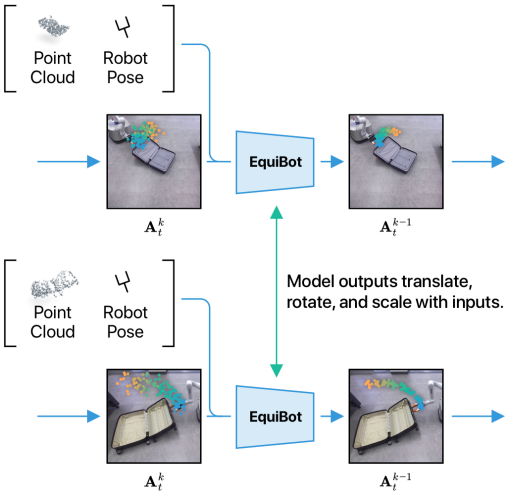

Figure 2: **Overview of our method.** Given input scene point cloud and robot proprioception, our method performs a series of diffusion steps to obtain denoised actions with SIM(3)-equivariance, meaning that when the inputs translate, rotate, and scale, its outputs are guaranteed to translate, rotate, and scale accordingly.

SIM(3)-equivariant network architectures. Let f be a function that takes a point cloud $\mathbf{X} \in \mathbb{R}^{N\times 3}$ as input. This function is considered SIM(3)-equivariant if $f(\mathbf{TX}) = \mathbf{T}f(\mathbf{X})$ for any rigid 3D transformation $\mathbf{T} := (\mathbf{R}, \mathbf{t}, s) \in \text{SIM}(3)$, where $\mathbf{R}$, $\mathbf{t}$, and s denote rotation, translation, and scale respectively. In this work, we use the same SIM(3)-equivariant encoder architecture and network layers as [17].

Diffusion process as policy representation. Our method uses Denoising Diffusion Probabilistic Models (DDPMs) to model the conditional distribution $p(\mathbf{A}_t|\mathbf{O}_t)$ similar to [1]. Starting from Gaussian noise $\mathbf{A}_t^K$, where K is the number of diffusion steps, DDPM performs K iterations of denoising to predict actions with decreasing levels of noise, $\mathbf{A}_t^{K-1}, \ldots, \mathbf{A}_t^0$. This process follows

$$\mathbf{A}_t^{k-1} = \alpha(\mathbf{A}_t^k - \gamma\epsilon_\theta(\mathbf{O}_t, \mathbf{A}_t^k, k) + \mathcal{N}(0, \sigma^2, I)).$$

The policy outputs $\mathbf{A}_t^0$ as its inference output. In this work, we use the CNN-based Diffusion Policy variant specified in [1] as the starting point of our architecture design. Note that although our method uses point clouds as visual input format rather than RGB images, most architectural details of this policy can be adopted. The original CNN-based Diffusion Policy architecture uses a noise prediction network that takes observation $\mathbf{O}_t$, diffusion timestep k, and noisy action $\mathbf{A}_t$ as input, and predicts the gradient $\nabla\mathbf{E}(\mathbf{A}_t)$ for denoising the noisy action input. The network first uses an encoder to encode the visual observations. The encoded visual features and positional embeddings of the time step parameter are passed into FiLM layers [19] so that the encoded visual inputs are integrated into the network. Then, the policy network uses a convolutional U-net [20] to process the input noisy

actions $\mathbf{A}_t$, the conditioned observations, and diffusion timestep k to predict the output denoising gradients.

Assumptions to observation and action spaces. To make an architecture SIM(3)-equivariant, we need to make a few assumptions about the structure of the input and output of the policy. First, our policy has to use point clouds, not RGB or depth images as input observations. This is because other forms of visual information do not have the necessary 3D information to make it equivariant to translation, rotation, and scaling. Second, we assume that the proprioceptive information $\mathbf{S}_t = (\mathbf{S}_t^{(x)}, \mathbf{S}_t^{(d)}, \mathbf{S}_t^{(s)})$ can be represented by 3D positions $\mathbf{S}_t^{(x)}$, normalized directions $\mathbf{S}_t^{(d)}$, and scalars $\mathbf{S}_t^{(s)}$. This is not a strong assumption since most proprioceptive information can be converted into a format that uses only positions, velocities, offsets, and scalars: end-effector positions go to $\mathbf{S}_t^{(x)}$; end-effector velocities can be converted to position targets and go to $\mathbf{S}_t^{(x)}$; end-effector orientations can be converted into rotation matrices and placed in $\mathbf{S}_t^{(v)}$; gripper open-close states go to $\mathbf{S}_t^{(s)}$. Similarly, we assume that the executed actions $\mathbf{A}_t = (\mathbf{A}_t^{(v)}, \mathbf{A}_t^{(d)}, \mathbf{A}_t^{(s)})$ consist of 3D offsets or velocities $\mathbf{A}_t^{(v)}$, normalized directions $\mathbf{A}_t^{(d)}$, and scalars $\mathbf{S}_t^{(s)}$. Similar to proprioceptive information, most existing action spaces can also be converted into this format.

3.2 SIM(3)-Equivariant Diffusion Policy

To design a SIM(3)-equivariant model architecture, our approach is to modify each part of the CNN-bsed Diffusion Policy architecture [1] to make them individually equivariant architectures. First, we design our point cloud encoder to be SIM(3)-equivariant and additionally output a centroid vector Θ_c as well as a scalar Θ_s quantifying object scale. Θ_c and Θ_s are then used to scale the inputs to subsequent layers of the network so that they are invariant to positions and scales. We then modify the FiLM layers, the convolutional U-net architecture, and other connecting layers to be SO(3)-equivariant (aka. equivariant to 3D rotations). Finally, before producing the output actions, we scale relevant part of the action back using Θ_c and Θ_s so the output is SIM(3)-equivariant to the input observation.

Encoder. We use a PointNet-based [21] encoder in this work. For SIM(3)-equivariance, we reuse the encoder Φ introduced in [17]. This encoder takes a point cloud $\mathbf{X}$ as input and outputs a latent code $\Theta = \Phi(\mathbf{X})$, comprised of four components: $\Theta := (\Theta_R, \Theta_{\text{inv}}, \Theta_c, \Theta_s)$, where Θ_R is a rotation equivariant latent representation, Θ_{inv} is an invariant latent representation, scalar Θ_s is the computed object scale, and vector Θ_c denotes the object centroid. For more details on this encoder, we refer to [17]. While [17] pre-trains the encoder using generated simulation data, we do not perform pre-training on the encoder and learn it from scratch. This eliminates the need to build task-specific simulation environments and collect custom pre-training data in these environments.

Routing input observations and actions into a conditional U-net. The conditional U-net takes two inputs: action representation $\mathbf{Z}_a$ and conditioning information $\mathbf{Z}_c$. To construct these inputs from point cloud encoding Θ, proprioception $\mathbf{S}_t$, and noisy action $\mathbf{A}_t$, we need to first translate and scale $\mathbf{S}_t$ and $\mathbf{A}_t$ using Θ_c and Θ_s so the resulting values are invariant to scale and position, and then merge relevant inputs. More concretely, we define the action representation as

$$\mathbf{Z}_a = f_{\text{fuse}}\big([\mathbf{A}_t^{(v)}/\Theta_s, \mathbf{A}_t^{(d)}], \mathbf{A}_t^{(s)}\big), \tag{1}$$

where f_{fuse} is a Vector Neuron layer that takes vector information as its first and scalar information as its second input argument. We define conditioning information as

$$\mathbf{Z}_c = (\mathbf{Z}_c^{(\text{vector})}, \mathbf{Z}_c^{(\text{scalar})}) = \Big(\big[\Theta_{\text{inv}}, (\mathbf{S}_t^{(x)} - \Theta_c)/\Theta_s, \mathbf{S}_t^{(d)}\big], \big[\mathbf{S}_t^{(s)}, \texttt{pos\_emb}(k)\big]\Big), \tag{2}$$

where $\mathbf{Z}_c^{\text{vector}}$ and $\mathbf{Z}_c^{\text{scalar}}$ are vector and scalar conditioning used as input to the FiLM layers [19] in the conditional U-net, and $\texttt{pos\_emb}(k)$ is the positional embedding of the diffusion timestep k.

SO(3)-equivariant conditional U-net. A conditional U-net is composed of 1D convolution layers, upsampling layers, and FiLM layers. We make this network SO(3)-equivariant by converting every layer of this network to an SO(3)-equivariant layer.

To make 1D convolution layers SO(3)-equivariant, we treat vector channels of the layer inputs as batch dimensions and perform the original convolution operations. We find that this simple change makes the convolution layer SO(3)-equivariant. We do not make any modifications to the upsampling layer, as it is naturally SO(3)-equivariant. To make the FiLM layer SO(3)-equivariant, we substitute vanilla linear layers with vectorized linear layers introduced in [22]. More formally, a FiLM layer is formulated as $\text{FiLM}(\mathbf{F}|\gamma,\beta) = \gamma\mathbf{F} + \beta$, where $\gamma = f(\mathbf{x})$ and $\beta = h(\mathbf{x})$ are parameters predicted from learned functions used to modulate a neural network layer's activations $\mathbf{F}$, and $\mathbf{x}$ is this neural network layer's input. We replace non-equivariant layers f and h with Vector Neuron layers, achieving rotation equivariance.

Output. The conditional U-net with SO(3)-equivariant layers processes $\mathbf{Z}_a$ and $\mathbf{Z}_c$ and outputs translation and scale invariant actions $\hat{\mathbf{A}}_{\text{inv}}$. To process this value into the final output of the policy, we assemble the final output action as $\hat{\mathbf{A}}_t = (\hat{\mathbf{A}}^{(v)}_{\text{inv}} \cdot \Theta_s, \hat{\mathbf{A}}^{(d)}_{\text{inv}}, \hat{\mathbf{A}}^{(s)}_{\text{inv}})$, where $\hat{\mathbf{A}}^{(x)}_{\text{inv}}$, $\hat{\mathbf{A}}^{(d)}_{\text{inv}}$, and $\hat{\mathbf{A}}^{(s)}_{\text{inv}}$ are the position, direction, and scalar components of the predicted invariant action.

3.3 Implementation Details

Data normalization can be important to the performance of diffusion models. Vanilla diffusion policy normalizes the observations and actions separately. We instead normalize all 3D-vector inputs (including positions and velocities) together due to our SIM(3)-equivariance assumptions. Implementation-wise, we take a subset of training data and compute the mean point cloud scale and mean action scale as s_{pc} and s_{ac}. Then, all position- and velocity-related information is divided by $s_{\text{pc}}/s_{\text{ac}}$ at the start of the network forward pass and multiplied back before the output is returned. This normalization factor ensures that the diffusion process always works with actions with values within the -1 to 1 range. We do not apply offsets to position and velocity information in this work. We normalize scalar information in the same way as the vanilla diffusion policy.

4 Experiments

Through our experiments, we want to answer the following questions: (1) does our method generalize to unseen scenarios better than imitation learning methods that do not leverage equivariance; (2) does our method demonstrate more robust performance than prior methods for equivariant visuomotor policy learning that do not leverage diffusion models; (3) does our method achieve better data efficiency when there is little training data compared to prior imitation learning methods? To answer these three questions, we perform quantitative experiments in both simulation (Section 4.1) and the real world (Section 4.2).

4.1 Simulation Experiments

4.1.1 Comparisons to Vanilla Diffusion Policies and Other Equivariant Policy Architectures

In this experiment, we test if our method outperforms prior methods in out-of-distribution generalization.

Comparisons. We compare our method to three baselines. (1) *Diffusion Policy (DP)* [1]: Vanilla diffusion policy using a point cloud as input. We substitute the imaged-based encoder to a PointNet++ encoder [21] similar to what EquiBot is using. (2) *Diffusion Policy with Augmentations (DP+Aug)*: This baseline uses the same architecture as the vanilla diffusion policy baseline, but trains with synthetically generated data augmentation. (3) *EquivAct* [17]: A reimplementation of [17] that drops the pre-training phase that requires task-specific simulated data.

Tasks. We use four simulated tasks: cloth folding, object covering, box closing, and push T (see Figure 3). The first three tasks involve two mobile robots manipulating various deformable and articulated objects. In these tasks, a simulated depth camera records point clouds of relevant objects in the scene from a third-person viewpoint. The policy takes these point clouds as input and commands the end-effector position, rotation, and gripper open-close actions of both robots. The push

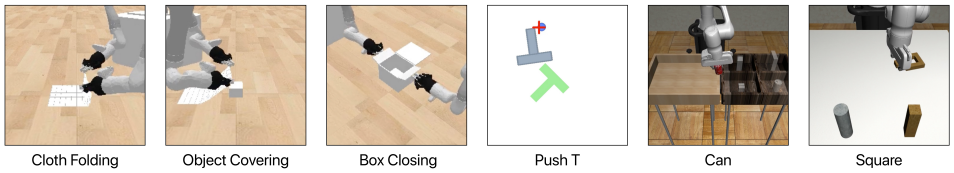

Figure 3: **Visualizations of simulation environments.** The three mobile manipulation tasks feature varied rigid, deformable, and articulated objects. The push T task features multi-modal demonstration data that challenge the learning algorithms. The Can and Square tasks from the Robomimic benchmark require precise position and orientation movements to successfully complete the tasks.

T benchmark task is a simulated 2D T-shape pushing game developed in [1] to showcase learning from multimodal demonstrations. To make this task compatible with our setup, we assume the agent and object are placed on the ground plane ($z = 0$) in 3D space. In this task, the policy receives the eight corners of the T-shape as input and outputs the velocity command to the pushing operator.

Augmentations. The *DP+Aug* baseline requires augmentations in the training phase. In all four environments, we augment the training data to (1) rotate the observation up to 360 degrees around the z-axis, (2) uniformly scale the observation within the range $0.5 \times -1.5\times$, and (3) apply a random Gaussian offset to the observation with standard deviation equal to 0.1 times the approximate workspace size.

Training and evaluation. We train all methods for 2,000 epochs. For every training run, we save a checkpoint every 50 epochs and evaluate the last 5 checkpoints saved at the end of training. For every evaluation, we run the policy in a randomly initialized environment for 10 episodes and record the mean final reward achieved by the policy. For all results we show, we run training over 3 random seeds and report the mean and standard deviation of evaluation results over these seeds.

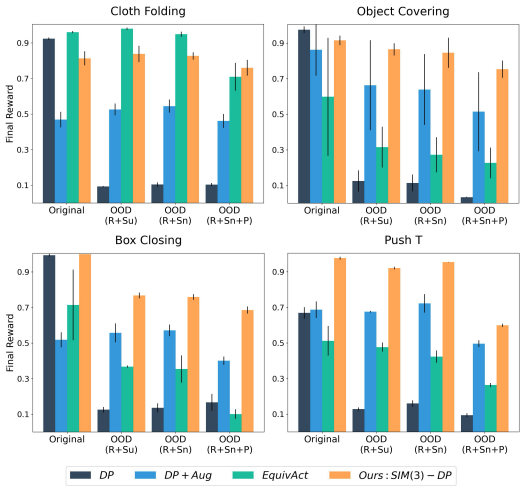

Figure 4: **Results of out-of-distribution generalization experiments.** We show that our method achieves more robust out-of-distribution generalization performance than methods that do not use diffusion processes to model policies and ones that do not utilize equivariance. Error bars show the mean and standard deviation over 3 seeds.

Evaluation setups. To gain insight into the generalizability of competing methods, we design four different evaluation setups to test our trained policies. The *Original* setup evaluates the trained policy at the exact same initial poses and goals as in the demos; the *OOD (R+Su)* setup randomizes initial object rotation with randomized object uniform scaling from $1\times$ to $2\times$; the *OOD (R+Sn)* setup randomizes rotation and scaling as in *OOD (R+Su)*, but additionally adds non-uniform scaling up to a 1.33 aspect ratio change; the *OOD (R+Sn+P)* setup adds dramatic position randomization on top of the *OOD (R+Sn)* setup.

Results. We show the results of this experiment in Figure 4. The *DP* baseline performs very well in the *Original* setup, but its performance drops significantly when it comes to any of the *OOD* setups. The *DP+Aug* baseline has slightly worse performance compared to *DP* in the distribution setup. This is because *DP+Aug* needs to account for more synthetically augmented scenarios and thus overfits less to the training data. When it comes to *OOD* setups, *DP+Aug* performs much better than *DP*, but still suffers from performance drops. The *EquivAct* baseline performs very well in the Cloth Folding task but displays subpar performance in Object Covering and Box Closing tasks due to unstable training performance among checkpoints. It also cannot perform well in the Push T task because it cannot handle multi-modal training data well. Our method performs stably in all four

tasks and suffers from the least amount of performance drop compared to all baselines. This shows that our method indeed outperforms prior methods in out-of-distribution generalization.

4.1.2 Data Efficiency Experiments

In this experiment, we aim to test if our method outperforms prior methods in a low-data regime, even if it is evaluated in distribution. Because we only care about in-distribution performance in this experiment, we only compare our method with the *DP* baseline. We adopt two Robomimic environments [23] for this experiment: Can and Square.

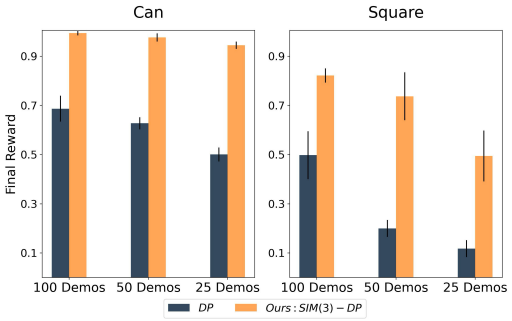

Figure 5: **Results of data efficiency experiments.** Our method achieves better data efficiency than Diffusion Policy when evaluated in-distribution in two benchmark tasks.

Setup. In each task, we train all methods in three setups: learning from 100 demos, 50 demos, and 25 demos. We run 2,000 epochs for each method with 3 random seeds. As in the previous experiment, each evaluation job computes the average performance over the last five checkpoints in the training run with 50-epoch intervals. The results plot the mean and standard deviation over seeded runs.

Results. The results of this experiment can be found in Figure 5. In both tasks, the performance of *DP* drops dramatically when the number of demos decreases from 100 to 25, while our method retains relatively higher performance when the amount of data drops. This is because when training data size is small, the training data does not cover the whole distribution of initial poses to allow the *DP* baseline to learn how to complete the task with all possible initial poses during evaluation. Our method, on the other hand, leverages equivariance to generalize to initial object poses that the training data does not cover.

4.2 Real Robot Experiments

In our real robot experiments, we show a series of experiments where we train mobile robots to perform everyday manipulation tasks from 5 minutes of single-view human demonstration videos. We select a suite of 6 tasks that involve diverse everyday objects, including rigid, articulated, and deformable objects (see Figure 6): (1) *Push Chair:* A robot pushes a chair towards a desk; (2) *Luggage Packing:* A robot picks up a pack of clothes and places it inside an open suitcase; (3) *Luggage Closing:* A robot closes an open suitcase on the floor; (4) *Laundry Door Closing:* A robot pushes the door of a laundry machine to close it; (5) *Bimanual Folding:* Two robots collaboratively fold a piece of cloth on a couch; (6) *Bimanual Make Bed:* Two robots unfold a comforter to make it cover the bed completely.

Data collection. We collect 15 human demonstration videos for each real robot task. We use a ZED 2 stereo camera to record the movement of a human operator using their fingers to manipulate the objects of interest at 15 Hz. After data collection, we use an off-the-shelf hand detection model, an object segmentation model, and a stereo-to-depth model to parse out the human hand poses and object point clouds in each frame of the collected demos. We then subsample this data to 3 Hz and convert it into a format supported by our policy training algorithm.

Mobile robot setup. In all real robot experiments, we use holonomic mobile bases with Kinova Gen3 7 DoF arms mounted on top. We use a single ZED 2 camera mounted at a fixed position in the workspace to obtain visual observations for the robot policy. After getting stereo images from the camera, we first use a learned stereo-to-depth model [24] to obtain the single-view point cloud of the scene, and then use the Grounded Segment Anything Model [25] to segment out the relevant objects in the scene based on a natural language description of objects involved in the task. The segmented single view point cloud is then used as the input visual observation for the policy.

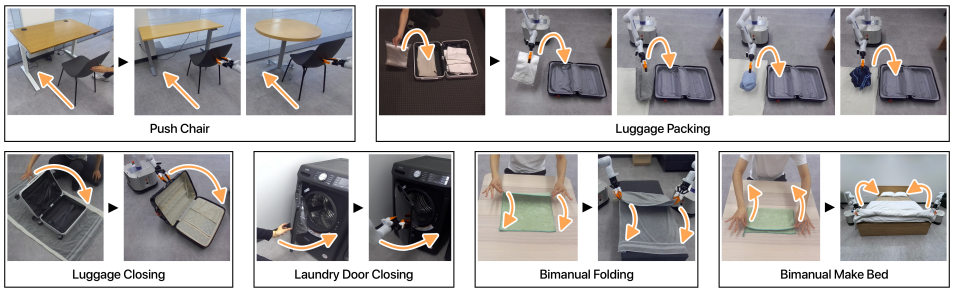

Figure 6: **Real robot evaluation setups.** Each block represents one task. In each block, we show sample training demonstrations collected by a human on the left and evaluation scenarios on the right.

	Push Chair		**Luggage Packing**				**Luggage Closing**
Unseen Poses →	–		Translate + Rotate				–
Object Variations →	Long Desk	Round Table	T-shirt	Towel Roll	Cap	Shorts	Large Luggage
DP	0/10	0/10	0/10	0/10	0/10	0/10	0/10
DP+Aug	0/10	0/10	0/10	0/10	0/10	0/10	2/10
Ours	8/10	10/10	7/10	3/10	8/10	8/10	6/10

	Laundry Door Closing	**Bimanual Folding**	**Bimanual Make Bed**
Unseen Poses →	–	Translate + Rotate	–
Object Variations →	–	Long Bath Towel	Comforter
DP	3/10	0/10	0/10
DP+Aug	1/10	0/10	0/10
Ours	8/10	6/10	8/10

Table 1: **Results of real robot experiments.** In a suite of 6 mobile manipulation tasks, we show that our method can learn from just 5 minutes of human demonstration, outperforming the Diffusion Policy and the Diffusion Policy with Augmentation baselines by a large margin.

We utilize the motion capture system in the room to track the pose of the mobile base. During one evaluation, the learned policy reads parsed point clouds and sends 8 actions at a time to the robot to execute. The mobile robot then moves the arm to achieve the commanded actions, moving the mobile base when the arm is too close or too far away from the base.

Training and evaluation. We train all methods for 1,000 epochs. After training, we evaluate each method for 10 episodes and record the success rate of the method. We vary the evaluation scenarios from the training scenarios differently in each task. In *Laundry Door Closing*, we perform evaluations in-distribution. In *Push Chair*, *Luggage Closing*, and *Bimanual Make Bed*, we evaluate with novel objects to make the evaluation out-of-distribution to the training data. In *Luggage Packing* and *Bimanual Folding*, we not only switch to novel objects but also translate and rotate the layout of the scene.

Results. The results of real robot experiments are shown in Table 1. The evaluation shows that our method can generalize to diverse unseen objects, outperforming the *DP* baseline in both in-distribution and out-of-distribution scenarios with novel objects and unseen object poses.

5 Limitations and Conclusion

Although our method is equivariant to translation, rotation, and scale, it does not handle changes in environment dynamics. It also does not handle variations in the relative positioning of objects when multiple objects are present. Resolving these limitations might involve explicitly modeling scene dynamics and individual objects.

We proposed a visuomotor policy learning method that is capable of generalizable and data-efficient policy learning in a wide range of robot manipulation tasks. Extending the proposed framework to multi-task setups and evaluating it in tasks that have longer horizons and larger action spaces are clear directions for future work.

References

[1] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[2] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[3] Y. Duan, M. Andrychowicz, B. Stadie, O. Jonathan Ho, J. Schneider, I. Sutskever, P. Abbeel, and W. Zaremba. One-shot imitation learning. *Advances in neural information processing systems*, 30, 2017.

[4] C. Finn, T. Yu, T. Zhang, P. Abbeel, and S. Levine. One-shot visual imitation learning via meta-learning. In *Conference on robot learning*, pages 357–368. PMLR, 2017.

[5] S. James, M. Bloesch, and A. J. Davison. Task-embedded control networks for few-shot imitation learning. In *Conference on robot learning*, pages 783–795. PMLR, 2018.

[6] Z. Mandi, F. Liu, K. Lee, and P. Abbeel. Towards more generalizable one-shot visual imitation learning. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2434–2444. IEEE, 2022.

[7] J. Yang, J. Zhang, C. Settle, A. Rai, R. Antonova, and J. Bohg. Learning periodic tasks from human demonstrations. *IEEE International Conference on Robotics and Automation (ICRA)*, 2022.

[8] S. James, P. Wohlhart, M. Kalakrishnan, D. Kalashnikov, A. Irpan, J. Ibarz, S. Levine, R. Hadsell, and K. Bousmalis. Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12627–12637, 2019.

[9] B. Mehta, M. Diaz, F. Golemo, C. J. Pal, and L. Paull. Active domain randomization. In *Conference on Robot Learning*, pages 1162–1176. PMLR, 2020.

[10] T. Yu, T. Xiao, A. Stone, J. Tompson, A. Brohan, S. Wang, J. Singh, C. Tan, J. Peralta, B. Ichter, et al. Scaling robot learning with semantically imagined experience. *arXiv preprint arXiv:2302.11550*, 2023.

[11] A. Simeonov, Y. Du, A. Tagliasacchi, J. B. Tenenbaum, A. Rodriguez, P. Agrawal, and V. Sitzmann. Neural descriptor fields: Se (3)-equivariant object representations for manipulation. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 6394–6400. IEEE, 2022.

[12] A. Simeonov, Y. Du, Y.-C. Lin, A. R. Garcia, L. P. Kaelbling, T. Lozano-Pérez, and P. Agrawal. Se (3)-equivariant relational rearrangement with neural descriptor fields. In *Conference on Robot Learning*, pages 835–846. PMLR, 2023.

[13] Z. Xue, Z. Yuan, J. Wang, X. Wang, Y. Gao, and H. Xu. Useek: Unsupervised se (3)-equivariant 3d keypoints for generalizable manipulation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1715–1722. IEEE, 2023.

[14] 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. *arXiv preprint arXiv:2206.08321*, 2022.

[15] T. Weng, D. Held, F. Meier, and M. Mukadam. Neural grasp distance fields for robot manipulation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1814–1821. IEEE, 2023.

[16] J. Seo, N. P. S. Prakash, X. Zhang, C. Wang, J. Choi, M. Tomizuka, and R. Horowitz. Robot manipulation task learning by leveraging se (3) group invariance and equivariance. *arXiv preprint arXiv:2308.14984*, 2023.

[17] J. Yang, C. Deng, J. Wu, R. Antonova, L. Guibas, and J. Bohg. Equivact: Sim(3)-equivariant visuomotor policies beyond rigid object manipulation, 2023.

[18] K. Chen, X. Chen, Z. Yu, M. Zhu, and H. Yang. Equidiff: A conditional equivariant diffusion model for trajectory prediction. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, pages 746–751. IEEE, 2023.

[19] E. Perez, F. Strub, H. De Vries, V. Dumoulin, and A. Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.

[20] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015.

[21] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.

[22] C. Deng, O. Litany, Y. Duan, A. Poulenard, A. Tagliasacchi, and L. J. Guibas. Vector neurons: A general framework for so (3)-equivariant networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12200–12209, 2021.

[23] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. In *arXiv preprint arXiv:2108.03298*, 2021.

[24] V. Guizilini, I. Vasiljevic, D. Chen, R. Ambruș, and A. Gaidon. Towards zero-shot scale-aware monocular depth estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9233–9243, 2023.

[25] T. Ren, S. Liu, A. Zeng, J. Lin, K. Li, H. Cao, J. Chen, X. Huang, Y. Chen, F. Yan, et al. Grounded sam: Assembling open-world models for diverse visual tasks. *arXiv preprint arXiv:2401.14159*, 2024.