

EquivAct: SIM(3)-Equivariant Visuomotor Policies beyond Rigid Object Manipulation

Jingyun Yang^{*1}, Congyue Deng^{*1}, Jimmy Wu², Rika Antonova¹, Leonidas Guibas¹, Jeannette Bohg¹

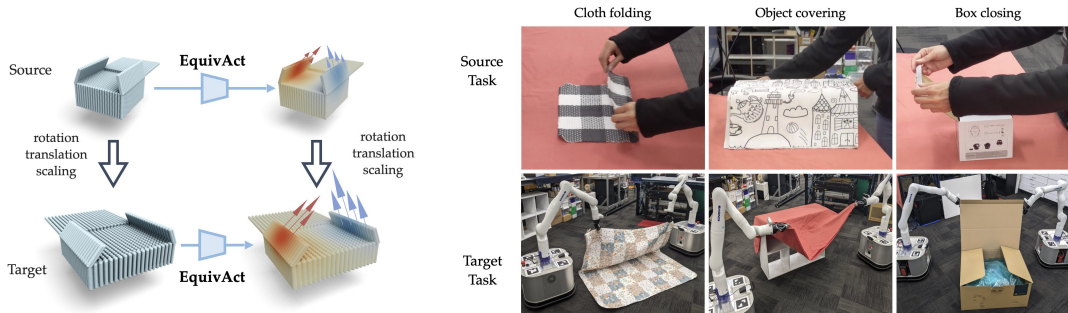


Fig. 1: **Overview.** Constructed with SIM(3)-equivariant point cloud networks, our method takes a few examples of solving a source task, then generalizes zero-shot to changes in object *appearances*, *scales*, and *poses*.

Abstract—If a robot masters folding a kitchen towel, we would expect it to master folding a large beach towel. However, existing policy learning methods that rely on data augmentation still don’t guarantee such generalization. Our insight is to add equivariance to both the visual object representation and policy architecture. We propose *EquivAct* which utilizes SIM(3)-equivariant network structures that guarantee generalization across all possible object translations, 3D rotations, and scales by construction. *EquivAct* is trained in two phases. We first pre-train a SIM(3)-equivariant visual representation on simulated scene point clouds. Then, we learn a SIM(3)-equivariant visuomotor policy using a small amount of source task demonstrations. We show that the learned policy directly transfers to objects that substantially differ from demonstrations in scale, position, and orientation. We evaluate our method in three manipulation tasks involving deformable and articulated objects, going beyond typical rigid object manipulation tasks considered in prior work. We conduct experiments both in simulation and in reality. For real robot experiments, our method uses 20 human demonstrations of a tabletop task and transfers zero-shot to a mobile manipulation task in a much larger setup. Experiments confirm that our contrastive pre-training procedure and equivariant architecture offer significant improvements over prior work. Project website: equivact.github.io

I. INTRODUCTION

Given a few examples of how to solve a manipulation task, humans can extrapolate and learn to solve variations of the same task where objects have different visual or physical properties. Existing works in robot learning still require extensive data augmentation to make the learned policies generalize to varied object scales, orientations, and visual appearances [1]–[5]. Even then, augmentations do not guarantee generalization to unseen variations.

^{*} Equal contribution. ¹ Department of Computer Science, Stanford University. ² Department of Computer Science, Princeton University.

This work was supported in part by the Toyota Research Institute and the National Science Foundation grant No.2030859 to the Computing Research Association for the CIFellows Project.

Contact: {jingyuny, rika.antonova}@stanford.edu

In this work, we focus on the problem of learning visuomotor policies that can take a few example trajectories from a single source manipulation scenario as input, then generalize zero-shot to scenarios with changes in objects’ appearances, scales, and poses. We go beyond pick-and-place of rigid objects and also handle deformable and articulated objects, such as clothes and boxes. Our insight is to *add equivariance to both the visual object representation and policy architecture*, enabling policies to generalize to novel object positions, orientations, and scales *by construction*.

We propose *EquivAct*, a novel visuomotor policy learning method that can learn closed-loop policies for 3D robot manipulation tasks using demonstrations from a single source manipulation scenario, then generalize zero-shot to new variations. A resulting policy takes a partial point cloud of the scene and robot end-effector poses as input, then outputs robot actions, which include end-effector velocity and gripper commands. Different from neural network architectures used in most prior works, we employ SIM(3)-equivariant network structures. This implies that when the input point cloud and end-effector positions are translated, rotated and scaled, the output end-effector velocities transform accordingly. With this, our method can learn from demonstrations of small-scale tabletop tasks, then offer zero-shot generalization to mobile manipulation tasks that involve larger variants of objects with different appearances.

Our method is composed of two phases: a representation learning phase and a policy learning phase. The representation learning phase serves two purposes. First, although the proposed architecture is equivariant to uniform scaling, we need to pre-train the representation to handle non-uniform scaling. Second, although rotation is handled via equivariance, we need robustness to nonlinear changes caused by objects’ self-occlusions when altering camera view angles. Hence, for the representation learning phase we collect a set of simulated point clouds of objects with randomized

non-uniform re-scaling. This data is not a target task demo and does not include robot actions. We train a SIM(3)-equivariant encoder-decoder architecture [6] that takes a single-view partial scene point cloud as input, then outputs both global and local features. We use a contrastive learning loss on paired point clouds, which ensures that local features for corresponding object parts in similar poses are closer than disparate parts. Next, in the policy learning phase, we assume access to a small set of demonstrations of a human solving a source task. Using these, we train a closed-loop policy that takes a partial point cloud of the scene as input, extracts global and local features using the encoder from the representation learning phase, and then passes the features through a SIM(3)-equivariant action prediction network to obtain end-effector actions.

We evaluate our method in three challenging scenarios that go beyond typical rigid object manipulation tasks of prior work [7]–[11]: comforter folding, container covering, and box closing (see Fig. 1). For each scenario, we show that our approach learns a closed-loop policy from source demonstrations, and then completes the target task without any further adaptation. We also compare with methods that rely on extensive augmentations and demonstrate that our approach of leveraging equivariance is more efficient for ensuring generalization to out-of-distribution object poses and scales.

II. RELATED WORK

A. Equivariant 3D Learning

A number of 3D deep learning works have studied the construction of neural network architectures that are equivariant to 3D transformations such as rigid transformations [12]–[19], multi-part motions [6], [20]–[22], or more general and arbitrary group actions [23]–[27]. Such network designs enable natural and robust generalizations to out-of-distribution inputs by construction without additional data augmentation in the training process. In our work, we leverage the implicit shape representation from [6] powered by the network structures from [15] to facilitate more geometrically interpretable and generalizable feature learning.

B. Equivariant Representations for Robot Manipulation

Prior works studied equivariant features for robot manipulation [7]–[11], but had several limitations: (1) focused on pick-and-place tasks and didn’t handle harder tasks with articulated and deformable objects; (2) mostly based on open-loop policies or hand-designed pick-and-place primitives, lacking closed-loop feedback; (3) only handled equivariances in translation and rotation and didn’t consider scale equivariance. [28] combines SO(2)-equivariant closed-loop policy learning with additional simulated augmentations for a suite of tabletop rigid object manipulation tasks, but cannot handle out-of-distribution objects with unseen uniform and non-uniform scaling. Compared to prior works, our work learns a robot manipulation policy that supports closed-loop feedback, can handle deformable and articulated objects, is equivariant to translation, rotation, and uniform scaling by

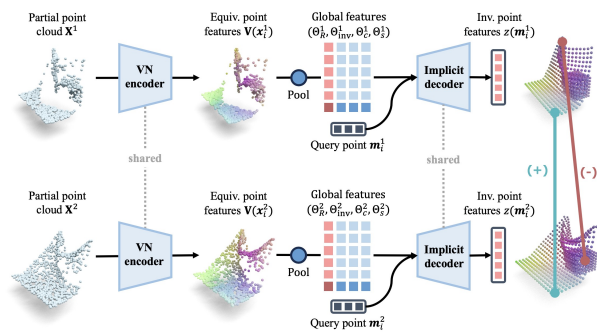


Fig. 2: **Representation learning pipeline.** This phase takes paired partial point clouds as inputs, processes them through an equivariant encoder-decoder architecture, then employs a contrastive loss based on invariant point features, yielding equivariant global and local features as output.

construction, and achieves generalization for objects with non-uniform scaling with a representation learning phase.

C. 3D Representations for Deformable and Articulated Object Manipulation

The problem of 3D manipulation of deformable and articulated objects has been studied by several prior works, including FlingBot [29], GarmentNets [30], FabricFlowNet [31], and ACID [32]. However, these works often are very focused on specific manipulation problems. For example, FlingBot focuses only on the task of flinging, FabricFlowNet is designed for pick-and-place tasks with cloth, and ACID is designed to manipulate volumetric deformable objects. In contrast, we propose a framework capable of learning a variety of 3D manipulation tasks, including ones that involve deformable and articulated objects.

III. METHOD

A. Preliminaries

SIM(3)-equivariance. Consider a function f that takes a point cloud $\mathbf{X} \in \mathbb{R}^{N \times 3}$ as input. We say it is SIM(3)-equivariant if, for any rigid 3D transformation $\mathbf{T} := (\mathbf{R}, \mathbf{t}, s) \in \text{SIM}(3)$ with rotation \mathbf{R} , translation \mathbf{t} , and scale s , the output of f transforms coherently with the input, that is $f(\mathbf{T}\mathbf{X}) = \mathbf{T}f(\mathbf{X})$. In [15], the authors introduced ‘Vector Neurons’ (VN) for constructing rotation-equivariant point cloud networks. In [6], VN building blocks were used to obtain a SIM(3)-equivariant shape encoding using implicit representations. Given a partial point cloud \mathbf{X} , a VN encoder Φ encodes the input into a latent code $\Theta = \Phi(\mathbf{X})$ comprised of four components $\Theta := (\Theta_R, \Theta_{\text{inv}}, \Theta_c, \Theta_s)$. Here, $\Theta_R \in \mathbb{R}^{C \times 3}$ is a rotation equivariant latent representation, $\Theta_{\text{inv}} \in \mathbb{R}^C$ is an invariant latent representation, scalar $\Theta_s \in \mathbb{R}$ represents object scale, and $\Theta_c \in \mathbb{R}^3$ represents the object centroid. For any $\mathbf{T} = (\mathbf{R}, \mathbf{t}, s) \in \text{SIM}(3)$, the equivariance of encoder Φ can be written as:

$$\mathbf{T}\Theta = (\Theta_R \mathbf{R}, \Theta_{\text{inv}}, s\Theta_c \mathbf{R} + \mathbf{t}, s\Theta_s) = \Phi(s\mathbf{X}\mathbf{R} + \mathbf{t}). \quad (1)$$

The latent code can be decoded by an implicit decoder Ψ that takes a query position $\mathbf{x} \in \mathbb{R}^3$ as input and outputs a per-point feature $z(\mathbf{x})$ with:

$$z(\mathbf{x}) = \Psi(\Theta_{\text{inv}}, \langle \Theta_R, (\mathbf{x} - \Theta_c) / \Theta_s \rangle). \quad (2)$$

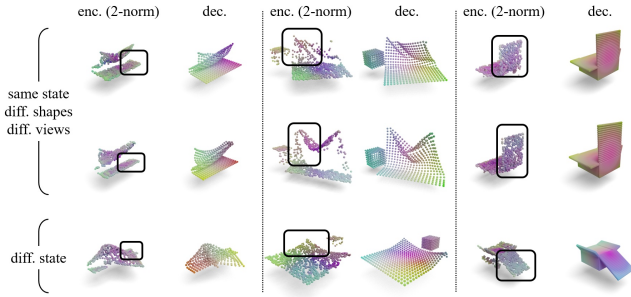


Fig. 3: **Visualizations of per-point features.** The encoder features are equivariant vector-valued features on the partial point cloud observations and the visualizations are done on their invariant components (channel-wise 2-norms). The decoder features are invariant scalar-valued features on the complete objects. The RGB values are computed via PCA within each task. All point clouds are aligned to the canonical pose for visualization. **Top two rows:** Objects of different shapes viewed from different camera angles but at the same poses. Both encoder and decoder features show strong correspondences within each state due to the contrastive learning. **Bottom row:** Objects in a different state (features become different from the two top rows).

Problem formulation. We consider a problem where an agent is given a set of demonstrations $\mathcal{D}_{\text{demo}} = \{\tau_d\}_{d=1}^{N_{\text{demo}}}$. Each demo τ_d consists of a sequence of tuples $\{(o_t, a_t)\}_{t=1}^T$, where $o_t = (\mathbf{X}_t, \{\mathbf{y}_t^e\}_{e=1}^E)$ is an observation that includes a point cloud of the scene \mathbf{X}_t and end-effector poses $\{\mathbf{y}_t^e\}_{e=1}^E$, while a_t is an action that consists of velocity and gripper commands for all end-effectors. Using $\mathcal{D}_{\text{demo}}$ dataset from a source scenario, we train a policy $\pi(a|o)$ to take a point cloud of the target scene as input and output an action a . We then deploy the learned policy in a target scenario that involves objects with different appearances, scales, and poses.

B. Learning SIM(3)-equivariant 3D Visual Representations from Simulation Data

The first phase of our method is a representation learning phase that addresses two issues. First, although architecture based on vector neurons is equivariant to uniform scaling, we need to also handle non-uniform re-scaling of objects. Second, although rotation is handled via equivariance in principle, in practice we need robustness to nonlinear changes caused by objects' self-occlusions when altering camera view angles. Hence, we start with the representation learning phase, illustrated in Fig. 2, which helps ensure generalization to varied objects in the same category (e.g. non-uniform re-scaling) and ensures robustness to nonlinear effects of changing viewpoints. We assume access to a dataset $\mathcal{D}_{\text{sim}} = \{(\mathbf{X}^q \in \mathbb{R}^{N \times 3}, \mathbf{M}^q \in \mathbb{R}^{M \times 3})\}_{q=1}^{|\mathcal{D}_{\text{sim}}|}$ containing point clouds of simulated scenes that include the objects of interest. In each sample, we record the partial point cloud of the scene unprojected from one camera view \mathbf{X}^q and the ground-truth mesh of the objects in the scene \mathbf{M}^q . Since simulated scenes include full information about object meshes, we can assume that all ground-truth meshes \mathbf{M}^q have the same number of vertices, and the points in two meshes correspond to each other. This canonicalization simplifies constructing effective losses for this learning phase.

We train the visual representation using a contrastive loss

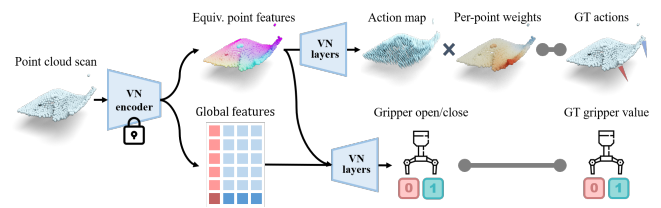


Fig. 4: **Policy learning architecture.** We first pass a point cloud captured during policy execution through the frozen encoder from the representation learning phase to get local and global equivariant features. These are passed to two VN heads to get target end-effector velocities and open/close actions.

similar to [33]. Specifically, we sample a simulated data pair of object point clouds $(\mathbf{X}^1, \mathbf{M}^1)$ and $(\mathbf{X}^2, \mathbf{M}^2)$ where two different object instances share a similar pose, articulation, or deformation. Then, we aim to learn per-point features $\{z(\mathbf{m}_i^1)\}_{i=1}^M$ and $\{z(\mathbf{m}_i^2)\}_{i=1}^M$ at ground-truth mesh points. Since we know the two ground-truth meshes have one-to-one correspondences, we know that $z(\mathbf{m}_i^1)$ and $z(\mathbf{m}_i^2)$ correspond to the same points on the mesh and should have similar latent features. We then apply a PointInfoNCE loss [33] to train the latent representation:

$$\mathcal{L} = - \sum_{i=1}^M \log \frac{\exp(z(\mathbf{m}_i^1) \cdot z(\mathbf{m}_i^2) / \tau)}{\sum_{j \neq i} \exp(z(\mathbf{m}_i^1) \cdot z(\mathbf{m}_j^2) / \tau)}. \quad (3)$$

Instead of the SDF decoder in [6] which is incompatible with thin-shell deformable objects with highly varying geometries (like cloth), or the contrastive loss on the multi-views of the same object as originally introduced in [33], we sample positive point pairs across different objects at comparable poses. This encourages representation to learn similar features for corresponding points of varied object instances (see Fig. 3).

C. Learning Generalizable Visuomotor Policies with SIM(3)-equivariant Architecture

After the representation learning phase, we freeze the encoder Φ and learn an action prediction head on top of the learned feature representation with a small number of human demonstrations. As observed in [34], the intermediate representations from the encoder can possess desirable properties even without decoding at inference time.

Architecture. The policy learning architecture is illustrated in Fig. 4. Given input observation $o = (\mathbf{X}, \{\mathbf{y}_e^e\}_{e=1}^E)$, our policy π first passes the input point cloud through the encoder Φ to obtain a global feature Θ_R and per-point local features $\{\mathbf{V}(\mathbf{x}_n)\}_{n=1}^N$. Then, the policy architecture is split into two branches. The end-effector velocity prediction branch takes the local features as input and passes them through three VN layers to obtain a velocity map $\{\hat{\mathbf{v}}_n\}_{n=1}^N$. When the policy is deployed, for each end-effector $e \in \{1, \dots, E\}$ we find a point \mathbf{x}_{k^e} in the point cloud that is closest to the current end-effector position \mathbf{y}^e ; we use the velocity map output $\hat{\mathbf{v}}_{k^e}$ as the target velocity. The other branch passes the global and local features through three other VN layers to produce a target open/close command

$\{\hat{g}^e\}_{e=1}^E$ for each end-effector. The target end-effector velocities and gripper open/close commands are then concatenated to form the output action $\hat{a} = (\{\hat{\mathbf{v}}_{k^e}\}_{e=1}^E, \{\hat{g}^e\}_{e=1}^E)$.

Loss function. We use a multi-component loss function for training the target velocities and gripper open/close commands. To supervise the outputs of the velocity map, we want to apply a loss term on every point in the point cloud. However, points that are located farther away from any end-effector in the demonstration should receive less supervision than points closer to one of the end-effectors. To achieve this, we weigh the loss term applied to each point by how far that point is from the end-effectors. Concretely, the loss is defined by:

$$\mathcal{L}_{\text{vel}} = \frac{1}{N} \sum_{n=1}^N \sum_{e=1}^E w(\mathbf{x}_n, \mathbf{y}^e) \cdot \|\hat{\mathbf{v}}_n - \mathbf{v}^e\|_2^2 \quad (4)$$

Here, \mathbf{v}^e denotes the ground truth velocity of the e -th end-effector, $\hat{\mathbf{v}}_n$ denotes the n -th point in the output velocity map, and $w(\mathbf{x}_n, \mathbf{y}^e) = \exp(-(\mathbf{x}_n - \mathbf{y}^e)^2 / (2\sigma))$ is a weighting function that is large when \mathbf{x}_n is close to \mathbf{y}^e and small when \mathbf{x}_n is far from \mathbf{y}^e . The gripper open/close loss is simply defined as the mean squared error (MSE) between the gripper command \hat{g}^e and the ground truth gripper state g^e observed during the demo: $\mathcal{L}_{\text{grip}} = \text{MSE}(\hat{g}^e, g^e)$. The policy learning loss is a weighted sum of the above loss components:

$$\mathcal{L} = \mathcal{L}_{\text{vel}} + \lambda_{\text{grip}} \mathcal{L}_{\text{grip}}, \quad (5)$$

where λ_{grip} is a weighting term for the velocity component of the loss function, which we set to 0.05 in our experiments to scale the two terms to comparable magnitudes.

IV. EXPERIMENTS

Through our experiments, we aim to answer the following questions: **(Q1)** Can our method learn from a few demonstrations in one scenario, then generalize to scenarios with unseen object sizes and poses? **(Q2)** Does our method outperform methods that rely on extensive augmentations to achieve generalization? **(Q3)** Does our method perform better than methods that don't employ equivariant architectures or representations pre-trained from simulation?

A. Comparisons

We compare our method with several baselines and also analyze ablations, as described below.

(1) **PointNet+BC:** a baseline behavior cloning (BC) algorithm that trains a neural network architecture, which takes a partial point cloud of the scene as input, encodes the input with a PointNet [35], concatenates the point cloud feature with robot proprioception, then passes the concatenated vector through an MLP to obtain target robot actions. Since this baseline has no representation learning phase, it only uses the source task demos and does not utilize simulated data. This baseline examines a basic policy learning method one would use for processing point cloud inputs and outputting target actions without adding more complexity to handle generalization.

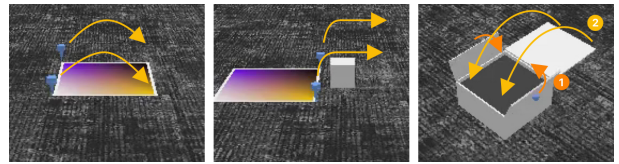


Fig. 5: **Simulation environments.** *Cloth Folding (left):* two grippers fold a piece of cloth by grasping two corners of the cloth. *Object Covering (middle):* two grippers pick up a cloth at two corners and drag it to fully cover another object. This task tests the ability to handle scenarios with several objects. *Box Closing (right):* two manipulators close a box with three flaps by first closing the side flaps and then closing the larger front/back flap. This task tests manipulation with articulated objects.

(2) **PointNet+BC_{aug}:** a variation of *PointNet+BC* that augments the demonstration data with additional object poses and scales, aiming to make test-time scenarios ‘in distribution’ with respect to the augmented data. The total amount of data used after augmentation equals the amount of data used in our method. This baseline serves to answer (Q2) from above.

(3) **PointContrast+BC:** this baseline first trains a latent representation of the 3D scene with PointContrast [33], then trains an MLP that takes the learned features and robot proprioception as input and outputs target robot actions. This baseline serves to answer question (Q3).

(4) **PointContrast_{aug}+BC:** a variation of *PointContrast+BC* that uses augmented object poses during feature training, which is a way of approximating equivariant features with Siamese training [36].

(5) **Ours w/o pre-training:** this ablation trains the whole architecture, including the visual encoder and the action heads, together without pre-training.

(6) **Ours w/o equivariance:** this ablation replaces all SIM(3)-equivariant parts in our method with non-equivariant PointNet-based components with similar sizes.

(7) **Ours w/o equivariance with aug:** this ablation is our method without SIM(3)-equivariance, but with using augmentation for object poses and scales during feature training.

B. Simulation Experiments

Tasks. We evaluate our method on three challenging robot manipulation tasks involving various deformable and articulated objects: cloth folding, object covering, and box covering. We illustrate the three environments in Fig. 5.

Representation learning data. Our simulation environments use PyBullet simulation engine [37]. For simplicity, we use floating end-effectors, which can physically push objects and attach to objects to imitate grasping. We generate randomized simulation data in each task for representation pre-training. In *Cloth Folding* and *Object Covering* tasks, we generate representation learning data by grasping two corners of the object and performing a pick-and-place motion to random target positions. We collected 200 simulated episodes for each task, varying the non-uniform scaling and camera viewpoint in each episode. Note that these episodes were not demonstrations and didn't have to include successful executions of the target task. In the *Box Closing* task, we

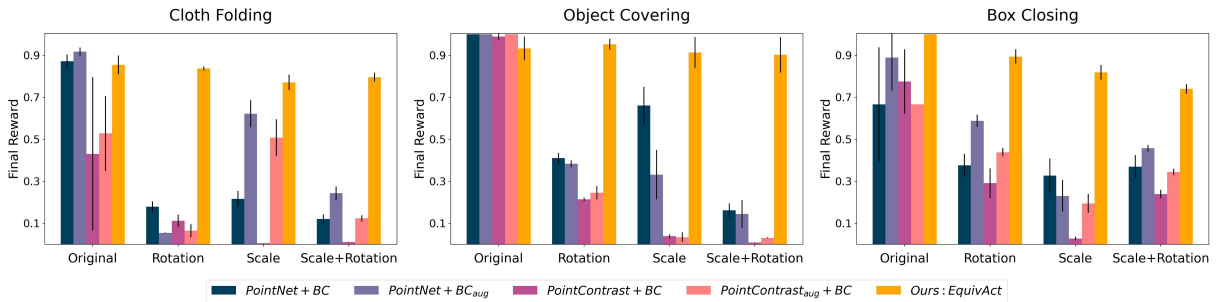


Fig. 6: **Results for simulation experiments.** We evaluate 3 manipulation tasks involving deformable and articulated objects. The comparisons with baselines show that our method outperforms prior methods that rely on augmentations to achieve generalization or utilize non-equivariant representations.

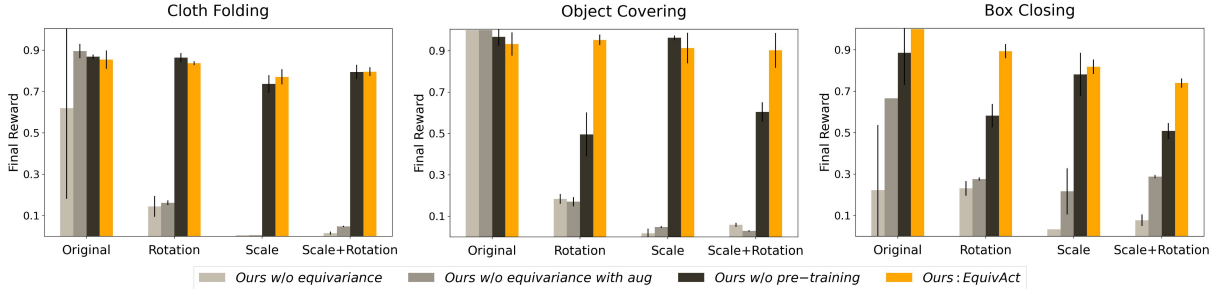


Fig. 7: **Ablation results in simulation.** Comparisons show that the pre-training and equivariance components are essential for good performance.

generated data for pre-training by taking snapshots of the box in various articulated poses (i.e. randomized positions for each box flap). We collected 20k point clouds (equivalent to 200 episodes, since episodes in *Box Closing* are roughly 100 steps) with varying non-uniform scaling and camera pose.

Demonstration data. We collect 50 noisy demonstrations for all tasks. In each task, the 50 demos include the same object at the canonical pose. This means the policy is trained using demos of only one given manipulation scenario.

Evaluation. To thoroughly examine the generalization capability of the learned policy, we evaluate it in four different setups. The *Original* setup evaluates the performance of the policy with an object that is placed and sized the same as in the demonstrated scenario. The *Rot* setup randomizes the object rotation while keeping the object scale constant. The *Scale* setup tests non-uniform re-scaling, where each object dimension is re-scaled up to twice its original size, while keeping the aspect ratio within at most 1 : 1.33 for the shortest vs. longest dimension; in this setup we keep the object in the canonical pose. The *Rot+Scale* setup randomizes both the rotation and non-uniform scaling of the object. Note that except for the *Original* setup, all other setups test out-of-distribution performance of the trained policy. In particular, in the *Box Closing* task, we add a further challenge to the trained policy by removing the contents in the box at test time, which introduces further differences in point clouds between train and test time for this task. We measure the performance of a task using a task-specific reward function scaled between 0 and 1. In the *Cloth Folding* task, the reward is measured by how close the bottom two corners of the cloth are to the top two corners. In the *Object Covering* task, the reward equals the fraction of the object volume within the convex hull of the final cloth mesh. In the *Box Closing* task,

the reward equals $\frac{\theta_{\text{left}} + \theta_{\text{right}} + \theta_{\text{top}}}{3\pi}$, where θ_{left} , θ_{right} , and θ_{top} denote the angular pose of the left, right, and top flaps (0 denotes fully open and π denotes fully closed).

Results. Fig. 6 shows experimental results in simulation. All methods perform well in the *Original* evaluation setup. This shows that all methods learn to complete the demonstrated tasks. In the other setups, the two **PointNet+BC** baselines display a significant performance drop. Even after adding augmentations in **PointNet+BC_{aug}**, the baseline is still not able to recover to the level of its ‘in-distribution’ performance for any of the three tasks. This shows that augmentation does not guarantee effective generalization to varied object scales and poses. Similarly, the **Point-Contrast+BC** baseline performs well in the *Rot* setup but struggles to perform well in the evaluation setups that involve scaling. In contrast, our pre-training phase that utilizes paired data with non-uniform re-scaling helps our method perform well across all setups, including *Scale* and *Rot+Scale* (with only a minor performance drop in the simpler *Original* setup overall).

Our ablation experiments are presented in Fig. 7. The drop in performance of **Ours w/o Pretraining** and **Ours w/o Equivariance** ablations show that pre-training and equivariance are essential to prevent performance degradation. In particular, **Ours w/o Pretraining** ablation performs substantially worse than our method in the *Rotate* setup in the covering and closing tasks. In these tasks, different viewing angles can cause significant differences in point cloud observations, e.g. different self-occlusions of the object (though this effect is not significant in the folding task). This ablation result shows that the pre-training phase effectively makes our method robust to nonlinear changes in point cloud observations caused by varied viewing angles.

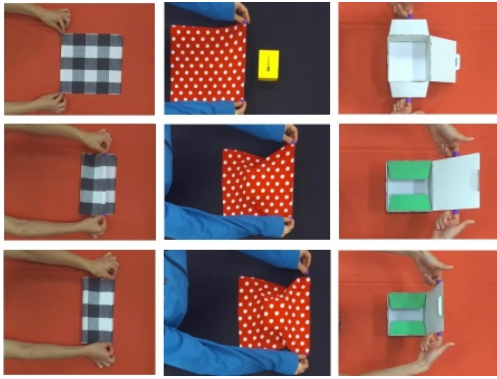


Fig. 8: **Human demonstrations collected for real robot experiments.** From left to right: cloth folding, object covering, and box closing tasks.

C. Real Robot Experiments

To illustrate the effectiveness of our method in the real world, we test it with both quantitative and qualitative experiments in a mobile manipulation setup.

Tasks and data collection. We run our method in the same three manipulation tasks as our simulation experiments. In each task, our method takes 20 human demonstrations collected on a tabletop setup as training data. We use a ZED 2 stereo camera positioned above the table to record the movement of the objects and human hands. After data collection, we segment out the part of the point cloud that corresponds to the objects the human is manipulating and also parse out the human finger positions in each frame. We treat segmentation as a separate research problem, which could be addressed by state-of-the-art methods, such as [38]. To avoid conflating the evaluation of the performance of various segmentation methods in our experiments, we segment out the objects using simple (robust) color filtering techniques.

Mobile robot setup. The robots operate in a large 4×3 meter workspace. We use holonomic mobile bases with a powered-caster drive system [39], and Kinova Gen3 7-DoF arms equipped with Robotiq 2F-85 grippers. We use ZED 2 stereo cameras positioned on the ceiling to obtain point clouds of the scene, then segment out the relevant parts of the scene. The policy takes the partial point cloud of the objects in the scene and the proprioception readings of the two mobile robots as input, then outputs a velocity and gripper open/close command for each robot.

Results. We compare our method with **PointNet+BC_{aug}**, the baseline that performed best in simulation. Table I shows quantitative results. We use similar evaluation metrics as those in the simulation experiments, with a few simplifica-

| | Cloth Folding | Object Covering | Box Closing |
|----------------------------|---------------|-----------------|--------------|
| PointNet+BC _{aug} | 0.083 | 0.000 | 0.000 |
| Ours (EquivAct) | 0.919 | 0.825 | 0.867 |

TABLE I: **Quantitative results for real robot experiments.** The table reports average performance over 10 trials. We randomize the position of the objects in a $\pm 20cm$ range and the orientation of the objects in a $\pm 15^\circ$ range. In all tasks, our method can complete the target manipulation task in the majority of trials, while the baseline fails to finish all tasks. Fig. 9 shows qualitative samples produced by our policy.

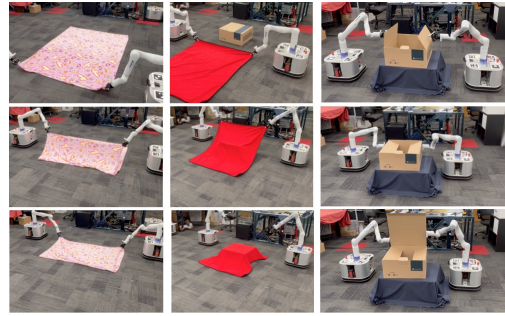


Fig. 9: **Qualitative samples of deploying the policy on hardware.** We deploy our learned policies to a mobile robot platform to manipulate target objects that are at least 6 times larger than the demonstrated objects. They have different visual appearances, aspect ratios, and physical properties (e.g. friction and elasticity of the cloth) than those in the source (tabletop) tasks. Our policy can generalize zero-shot to the target objects and successfully carry out the task. Please see our [website](#) for full videos of these samples.

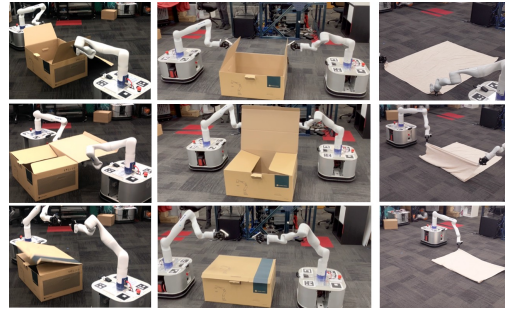


Fig. 10: **Task variations.** To illustrate the generalization capability of our method, we test it on various objects and initial poses. The first two columns show that the *Box Closing* policy can close different-sized boxes in different initial rotations. The third column shows that the *Cloth Folding* policy can handle cloths with distinct appearance and scale compared to the pink blanket in Fig. 9. See our [website](#) for further examples.

tions. In *Object Covering*, we measure the fraction of the box covered as seen from the top. In *Box Closing*, we count the number of box flaps fully closed (divided by the number of flaps) rather than computing exact angles. In all three tasks, our method extrapolates from tabletop-scale demonstrations and successfully manipulates objects that are up to six times larger in size. In contrast, the **PointNet+BC_{aug}** baseline has a lot of trouble when attempting all three tasks. The common failure mode of the baseline is that it keeps producing target robot velocities that point in one direction. This is likely because the method is not able to generalize to out-of-distribution visual inputs.

We further demonstrate the robustness of our method by varying the position, rotation, scale, and appearance of the objects (see Fig. 10). Our method can robustly generalize to all these variations.

V. CONCLUSION

We presented *EquivAct*, a visuomotor policy learning method that learns generalizable closed-loop policies for 3D manipulation tasks. We showed that our method successfully generalizes from a small set of demonstrations to a diverse set of target scenarios in both simulation and real robot experiments. We hope that our work motivates further use of equivariant architectures and facilitates learning generalizable robot manipulation policies.

REFERENCES

- [1] 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*, 2019, pp. 12 627–12 637.
- [2] B. Mehta, M. Diaz, F. Golemo, C. J. Pal, and L. Paull, "Active domain randomization," in *Conference on Robot Learning*. PMLR, 2020, pp. 1162–1176.
- [3] M. Laskin, K. Lee, A. Stooke, L. Pinto, P. Abbeel, and A. Srinivas, "Reinforcement learning with augmented data," *Advances in neural information processing systems*, vol. 33, pp. 19 884–19 895, 2020.
- [4] N. Hansen and X. Wang, "Generalization in reinforcement learning by soft data augmentation," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2021, pp. 13 611–13 617.
- [5] 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.
- [6] J. Lei, C. Deng, K. Schmeckpeper, L. Guibas, and K. Daniilidis, "Efem: Equivariant neural field expectation maximization for 3d object segmentation without scene supervision," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 4902–4912.
- [7] 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)*. IEEE, 2022, pp. 6394–6400.
- [8] 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*. PMLR, 2023, pp. 835–846.
- [9] 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)*. IEEE, 2023, pp. 1715–1722.
- [10] 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.
- [11] 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)*. IEEE, 2023, pp. 1814–1821.
- [12] N. Thomas, T. Smidt, S. Kearnes, L. Yang, L. Li, K. Kohlhoff, and P. Riley, "Tensor field networks: Rotation- and translation-equivariant neural networks for 3d point clouds," *arXiv preprint arXiv:1802.08219*, 2018.
- [13] F. Fuchs, D. Worrall, V. Fischer, and M. Welling, "Se (3)-transformers: 3d roto-translation equivariant attention networks," *Advances in neural information processing systems*, vol. 33, pp. 1970–1981, 2020.
- [14] H. Chen, S. Liu, W. Chen, H. Li, and R. Hill, "Equivariant point network for 3d point cloud analysis," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 14 514–14 523.
- [15] C. Deng, O. Litany, Y. Duan, A. Poulénard, 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*, 2021, pp. 12 200–12 209.
- [16] S. Assaad, C. Downey, R. Al-Rfou, N. Nayakanti, and B. Sapp, "Vn-transformer: Rotation-equivariant attention for vector neurons," *arXiv preprint arXiv:2206.04176*, 2022.
- [17] O. Katzir, D. Lischinski, and D. Cohen-Or, "Shape-pose disentanglement using se (3)-equivariant vector neurons," in *European Conference on Computer Vision*. Springer, 2022, pp. 468–484.
- [18] J. Li, S. Luo, C. Deng, C. Cheng, J. Guan, L. Guibas, J. Ma, and J. Peng, "Orientation-aware graph neural networks for protein structure representation learning," 2022.
- [19] A. Poulénard and L. J. Guibas, "A functional approach to rotation equivariant non-linearities for tensor field networks," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 13 174–13 183.
- [20] C. Deng, J. Lei, B. Shen, K. Daniilidis, and L. Guibas, "Banana: Banach fixed-point network for pointcloud segmentation with inter-part equivariance," *arXiv preprint arXiv:2305.16314*, 2023.
- [21] H.-X. Yu, J. Wu, and L. Yi, "Rotationally equivariant 3d object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 1456–1464.
- [22] X. Liu, J. Zhang, R. Hu, H. Huang, H. Wang, and L. Yi, "Self-supervised category-level articulated object pose estimation with part-level se (3) equivariance," *arXiv preprint arXiv:2302.14268*, 2023.
- [23] R. Kondor and S. Trivedi, "On the generalization of equivariance and convolution in neural networks to the action of compact groups," in *International Conference on Machine Learning*. PMLR, 2018, pp. 2747–2755.
- [24] T. S. Cohen, M. Geiger, and M. Weiler, "A general theory of equivariant cnns on homogeneous spaces," *Advances in neural information processing systems*, vol. 32, 2019.
- [25] M. Weiler, P. Forré, E. Verlinde, and M. Welling, "Coordinate independent convolutional networks—isoetry and gauge equivariant convolutions on riemannian manifolds," *arXiv preprint arXiv:2106.06020*, 2021.
- [26] J. Aronsson, "Homogeneous vector bundles and g-equivariant convolutional neural networks," *Sampling Theory, Signal Processing, and Data Analysis*, vol. 20, no. 2, p. 10, 2022.
- [27] Y. Xu, J. Lei, E. Dobriban, and K. Daniilidis, "Unified fourier-based kernel and nonlinearity design for equivariant networks on homogeneous spaces," in *International Conference on Machine Learning*. PMLR, 2022, pp. 24 596–24 614.
- [28] M. Jia, D. Wang, G. Su, D. Klee, X. Zhu, R. Walters, and R. Platt, "Seil: Simulation-augmented equivariant imitation learning," *ICRA*, 2023.
- [29] H. Ha and S. Song, "Flingbot: The unreasonable effectiveness of dynamic manipulation for cloth unfolding," in *Conference on Robot Learning*. PMLR, 2022, pp. 24–33.
- [30] C. Chi and S. Song, "Garmentnets: Category-level pose estimation for garments via canonical space shape completion," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 3324–3333.
- [31] T. Weng, S. M. Bajracharya, Y. Wang, K. Agrawal, and D. Held, "Fabricflownet: Bimanual cloth manipulation with a flow-based policy," in *Conference on Robot Learning*. PMLR, 2022, pp. 192–202.
- [32] B. Shen, Z. Jiang, C. Choy, L. J. Guibas, S. Savarese, A. Anandkumar, and Y. Zhu, "Acid: Action-conditional implicit visual dynamics for deformable object manipulation," *arXiv preprint arXiv:2203.06856*, 2022.
- [33] S. Xie, J. Gu, D. Guo, C. R. Qi, L. Guibas, and O. Litany, "Pointcontrast: Unsupervised pre-training for 3d point cloud understanding," in *Computer Vision—ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part III 16*. Springer, 2020, pp. 574–591.
- [34] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International conference on machine learning*. PMLR, 2020, pp. 1597–1607.
- [35] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660.
- [36] W. Sun, A. Tagliasacchi, B. Deng, S. Sabour, S. Yazdani, G. E. Hinton, and K. M. Yi, "Canonical capsules: Self-supervised capsules in canonical pose," *Advances in Neural Information Processing Systems*, vol. 34, pp. 24 993–25 005, 2021.
- [37] E. Coumans and Y. Bai, "PyBullet, a Python module for physics simulation for games, robotics and machine learning," <http://pybullet.org>, 2016–2019.
- [38] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, *et al.*, "Segment anything," *arXiv preprint arXiv:2304.02643*, 2023.
- [39] R. Holmberg and O. Khatib, "Development and control of a holonomic mobile robot for mobile manipulation tasks," *The International Journal of Robotics Research*, vol. 19, no. 11, pp. 1066–1074, 2000.