

# PartManip: Learning Cross-Category Generalizable Part Manipulation Policy from Point Cloud Observations

Haoran Geng<sup>1,2\*</sup> Ziming Li<sup>1,2\*</sup> Yiran Geng<sup>1,2</sup> Jiayi Chen<sup>1,3</sup> Hao Dong<sup>1,2</sup> He Wang<sup>1,2†</sup>  
<sup>1</sup>CFCS, Peking University <sup>2</sup>School of EECS, Peking University <sup>3</sup>Beijing Academy of Artificial Intelligence

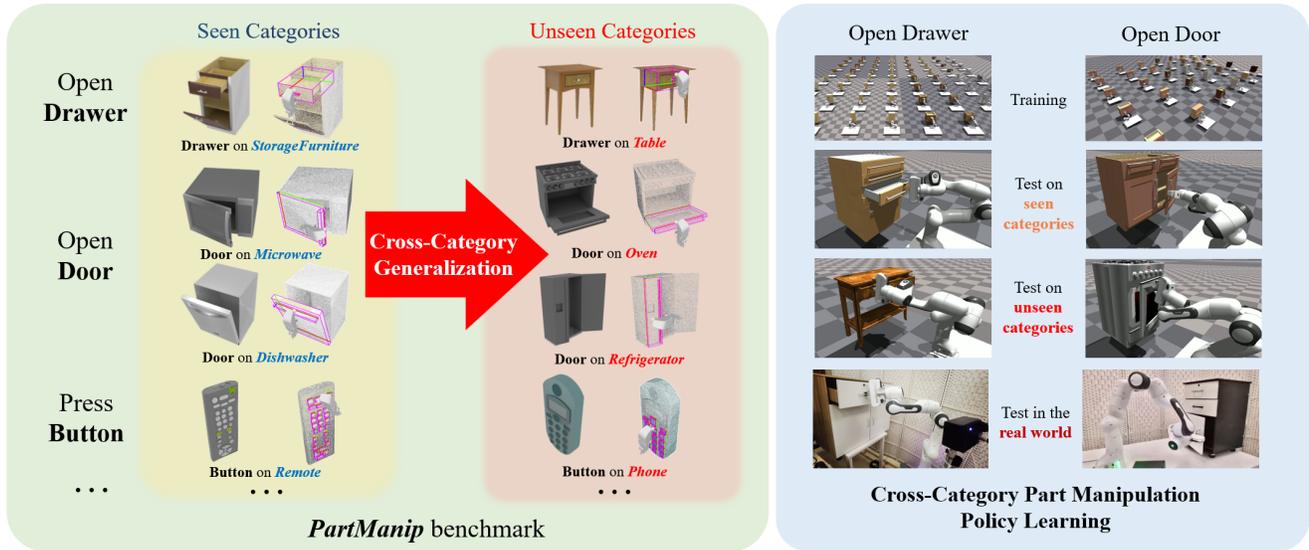


Figure 1. **Overview.** We introduce a large-scale cross-category part manipulation benchmark *PartManip* with diverse object datasets, realistic settings, and rich annotations. We propose a generalizable vision-based policy learning strategy and boost the performance of part-based object manipulation by a large margin, which can generalize to unseen object categories and novel objects in the real world.

## Abstract

Learning a generalizable object manipulation policy is vital for an embodied agent to work in complex real-world scenes. Parts, as the shared components in different object categories, have the potential to increase the generalization ability of the manipulation policy and achieve cross-category object manipulation. In this work, we build the first large-scale, part-based cross-category object manipulation benchmark, *PartManip*, which is composed of 11 object categories, 494 objects, and 1432 tasks in 6 task classes. Compared to previous work, our benchmark is also more diverse and realistic, i.e., having more objects and using sparse-view point cloud as input without oracle information like part segmentation. To tackle the difficulties of vision-based policy learning, we first train a state-

based expert with our proposed part-based canonicalization and part-aware rewards, and then distill the knowledge to a vision-based student. We also find an expressive backbone is essential to overcome the large diversity of different objects. For cross-category generalization, we introduce domain adversarial learning for domain-invariant feature extraction. Extensive experiments in simulation show that our learned policy can outperform other methods by a large margin, especially on unseen object categories. We also demonstrate our method can successfully manipulate novel objects in the real world. Our benchmark has been released in <https://pku-epic.github.io/PartManip>.

## 1. Introduction

We as humans are capable of manipulating objects in a wide range of scenarios with ease and adaptability. For building general-purpose intelligent robots that can work in

\*Equal contribution.

†Corresponding author: [hewang@pku.edu.cn](mailto:hewang@pku.edu.cn).

unconstrained real-world environments, it is thus important to equip them with generalizable object manipulation skills. Towards this goal, recent advances in deep learning and reinforcement learning have led to the development of some generalist agents such as GATO [32] and SayCan [1], however, their manipulation skills are limited to a set of known instances and fail to generalize to novel object instances. ManiSkill [25] proposes the first benchmark for learning category-level object manipulation, *e.g.*, learn open drawers on tens of drawer sets and test on held-out ones. However, this generalization is limited within different instances from one object category, thus falling short to reach human-level adaptability. The most recent progress is shown in GAPartNet [53], which defines several classes of generalizable and actionable parts (GAParts), *e.g.* handles, buttons, doors, that can be found across many different object categories but in similar ways. For these GAPart classes, the paper then finds a way to consistently define GAPart pose across object categories and devise heuristics to manipulate those parts, *e.g.*, pull handles to open drawers, based on part poses. As a pioneering work, GAPartNet points to a promising way to perform cross-category object manipulation but leave the manipulation policy learning unsolved.

In this work, we thus propose the first large-scale, part-based cross-category object manipulation benchmark, **Part-Manip**, built upon GAPartNet. Our cross-category benchmark requires agents to learn skills such as opening a door on storage furniture and generalizing to other object categories such as an oven or safe, which presents a great challenge for policy learning to overcome the huge geometry and appearance gaps among object categories.

Furthermore, our benchmark is more realistic and diverse. We use partial point clouds as input without any additional oracle information like part segmentation masks in the previous benchmark ManiSkill [17, 25], making our setting very close to real-world applications. Our benchmark also has much more objects than ManiSkill. We selected around 500 object assets with more than 1400 parts from GAPartNet [11] and designed six classes of cross-category manipulation tasks in simulation. Thanks to the rich annotation provided in GAPartNet, we can define part-based dense rewards to ease policy learning.

Due to the difficulty presented by our realistic and diverse cross-category setting, we find that directly using state-of-the-art reinforcement learning (RL) algorithms to learn a vision-based policy does not perform well. Ideally, we wish the vision backbone to extract informative geometric and task-aware representations, which can facilitate the actor to take correct actions. However, the policy gradient, in this case, would be very noisy and thus hinder the vision backbone from learning, given the huge sampling space. To mitigate this problem, we propose a two-stage training framework: first train a state-based expert that can access

oracle part pose information using reinforcement learning, and then distill the expert policy to a vision-based student that only takes realistic inputs.

For state-based expert policy learning, we propose a novel part-based pose canonicalization method that transforms all state information into the part coordinate frame, which can significantly reduce task variations and ease learning. In addition, we devise several part-aware reward functions that can access the pose of the part under interaction, providing a more accurate guide to achieve the manipulation objective. In combination, these techniques greatly improve policy training on diverse instances from different categories as well as a generalization to unseen object instances and categories.

For the vision-based student policy learning, we first introduce a 3D Sparse UNet-based backbone [16] to handle diverse objects, yielding much more expressivity than PointNet. To tackle the generalization issue, we thus propose to learn domain-invariant (category-independent) features via introducing an augmentation strategy and a domain adversarial training strategy [8, 9, 22]. These two strategies can alleviate the problem of overfitting and greatly boost the performance on unseen object instances and even categories. Finally, we propose a DAGger [33] + behavior clone strategy to carefully distill the expert policy to the student and thus maintain the high performance of the expert.

Through extensive experiments in simulation, we validate our design choices and demonstrate that our approach outperforms previous methods by a significant margin, especially for unseen object categories (more than 20% of the success rate in OpenDoor and OpenDrawer tasks). We also show real-world experiments.

## 2. Related Work

### 2.1. Learning Generalizable Manipulation Skills

Generalization is crucial yet challenging for robot application. Many works [7, 10, 15, 41, 52] combine supervise learning and motion planning to learn generalizable skills, *e.g.* grasping. However, these methods often require special architecture design for each task and may be unsuitable for complex manipulation tasks. Reinforcement learning (RL) has the potential to solve more complex task [2, 31], but the generalization ability of RL is an unsolved problem [13, 21]. To facilitate the research of generalizable manipulation skills, ManiSkill [25] proposes the first category-level object manipulation benchmark in simulation. [38] leverages imitation learning to learn complex generalizable manipulation skills from the demonstration. However, their demonstrations are collected by RL trained on every single instance, which requires a lot of effort to tune. In contrast, our experts can be directly trained on each object category.

	Generalization Level	# of Door Ins.	# of Door Cat.	# of Drawer Ins.	# of Drawer Cat.	Realistic Input
ManiSkill 1&2	Category-level	82	1	70	1	
<b>Ours</b>	<b>Cross-category-level</b>	<b>503</b>	<b>7</b>	<b>399</b>	<b>3</b>	✓

Table 1. **Comparison with ManiSkill 1&2 [17, 25]**. Realistic input indicates whether to need oracle part segmentation masks.

## 2.2. Vision-based Policy Learning

A lot of efforts are made to study how to learn policy from visual input [12, 20, 39, 40, 50, 54, 55]. Some works [30, 36, 37] use a pre-trained vision model and freeze the backbone to ease the optimization. Some [47, 48] leverage multi-stage training. The most related work to us is [4], which also trains a state-based expert and then distills to a vision-based policy, but the task is quite different.

## 2.3. 3D Articulated Object Manipulation

Manipulating articulated objects is an important and open research area due to the complexity of different objects’ geometry and physical properties. Previous work has proposed some benchmark [25, 42] but the diversity is limited. As for the methods, [3, 18, 27] explored motion planning, and [12, 24, 45, 47, 56] leverages visual affordance learning. Other works [6, 51] design special representations for articulated object manipulation and can generalize to a novel object category, but their methods are only suitable for the suction gripper.

## 3. PartManip Benchmark

By utilizing the definition of the generalizable and actionable part (GAPart) presented in GAPartNet [11], we build a benchmark for a comprehensive evaluation of the cross-category generalization policy learning approaches. GAParts are some kinds of parts that have similar geometry and similar interaction strategy across different object categories. For example, the handle on tables is often similar to those on safes, so we can regard the handle as a GAPart. The nature of GAPart ensures a general way for manipulation regardless of the object category, making it possible for cross-category generalization. We thus expect the manipulation policy trained on some object categories can generalize to other unseen object categories, and build the first benchmark for cross-category generalizable part manipulation policy learning.

Furthermore, our benchmark is more diverse and realistic than previous robotic manipulation benchmarks [25]. Diversity indicates that we have more object instances and categories, as shown in Table 1. Realism indicates that our observation space has less oracle information (*i.e.*, part segmentation masks) than ManiSkill [25] as discussed in Sec. 3.4, and thus is more acquirable in the real world.

We use Isaac Gym [23] as our simulator and most experiments are done in simulation. In the following, we in-

troduce our cross-category part-based object manipulation tasks in detail.

### 3.1. Task Formulation

We have six classes of tasks: **OpenDoor**, **OpenDrawer**, **CloseDoor**, **CloseDrawer**, **PressButton** and **GraspHandle**. Although **OpenDoor** and **OpenDrawer** require grasping the handle first, **GraspHandle** differs from them because it contains another GAPart *lid* with more object categories. Like traditional RL tasks, our task can be formulated as a Partially Observable Markov Decision Process (POMDP), because the true environment state  $s_t$  is not fully observable (especially for objects) in each timestep  $t$ . Given current partial observation  $o_t$ , a policy  $\pi$  needs to predict an action  $a_t$  to control the robot. After applying the action  $a_t$ , the next observation  $o_{t+1}$  and the reward  $r_{t+1}$  will be given by the environment, which can be used to train the policy. The final goal of the policy is to reach the success state (see supp. for more) in  $T$  steps (we set episode length  $T = 200$ ).

### 3.2. Object Assets

For each task, our object assets contain part-centered articulated objects, which are selected from the GAPartNet dataset [11]. GAPartNet dataset is a large-scale articulated object dataset with GAPart definitions and rich part annotations, including semantics and poses. The original GAPart classes contain *Round Fixed Handle*, *Slider Lid*, *Slider Button*, *Hinge Knob*, *Line Fixed Handle*, *Slider Drawer*, *Hinge Lid* and *Hinge Door*. But in this paper, we simplify them to only five classes: *handle*, *button*, *door*, *drawer* and *lid*. Each *door/drawer/lid* has a *handle* on it for manipulation. Figure 2 shows some visualization of our object assets, and Table 2 shows the statistics and split in our benchmark.

### 3.3. State Space and Action Space

The environment state  $s$  of each task is composed of the robot joint angles  $qp \in \mathbb{R}^{11}$ , joint velocity  $qv \in \mathbb{R}^{11}$ , gripper coordinate  $x \in \mathbb{R}^3$ , gripper rotation  $r \in SO(3)$ , and the target GAParts’ bounding boxes  $b_{\text{GAParts}} \in \mathbb{R}^{3 \times 8 \times n}$ , where  $n$  is the number of the target GAParts for the manipulation task. Specifically, there are two GAParts *handle* and *door/drawer/lid* for all the tasks in our benchmark except for **PressButton**, which only has one GAPart *button*.

For the action space, we use the Franka Panda arm and a parallel gripper as the end-effector. The action  $a_t$  is formulated as the gripper’s target 6 Dof pose at each timestep  $t$  for all tasks. It is then converted to the target joint angles by

	Open/CloseDoor						Open/CloseDrawer			PressButton				GraspHandle								
	Sto.	Mic.	Dis.	Ove.	Ref.	Tab.	Saf.	Sto.	Tra.	Tab.	Rem.	Was.	Mic.	Pho.	Tra.	Tab.	Sto.	Mic.	Saf.	Ove.	Dis.	Ref.
Training Set	338	4	21	-	-	-	-	238	8	-	178	30	7	-	8	36	40	3	1	-	-	-
Val-Intra Set	60	-	3	-	-	-	-	53	4	-	47	11	1	-	4	16	20	1	-	-	-	-
Val-Inter Set	-	-	-	20	43	13	1	-	-	96	-	-	-	40	-	-	-	-	-	20	24	43
Total	398	4	24	20	43	13	1	291	12	96	225	41	8	40	12	52	60	4	1	20	24	43

Table 2. **Task Statistics and Split.** Sto. = StorageFurniture, Mic. = Microwave, Dis. = Dishwasher, Ove. = Oven, Ref. = Refrigerator, Tab. = Table, Saf. = Safe, Tra. = Trashcan, Rem. = Remote, Was. = WashingMachine, Pho. = Phone. Note that all the numbers are the task number instead of the object number, *i.e.*, one object can have multiple parts and thus multiple tasks.



Figure 2. **Object Assets Visualization.** The object geometry and appearance are very different, especially in different object categories, which presents a great challenge for our *PartManip* benchmark.

inverse kinematics (IK), and used to control the robot arm by position control.

### 3.4. Observation Space

Because the true states of GAParts are often not observable in a realist setting, our acquirable input for policy learning is the partial observation  $o_t$ , which consists of the robot states  $(qp, qv, x, r)_t$ , the colored point clouds  $P_t \in \mathbb{R}^{6 \times 4096}$ , and a fixed point indicator  $c \in \mathbb{R}^3$  indicating the center of the target GAPart.

In our paper, we use a multi-camera setting in simulation to alleviate occlusion. The colored point clouds  $P_t$  are back-projected from 3-view RGBD images, assuming the camera poses are known. One camera is set above the object facing down, and the other two are set on each side of the object facing the objects. See We our supp. for the experiments with the single-camera setting.

The point indicator  $c$  is necessary for disambiguation because some objects have multiple actionable parts (*e.g.*, double-door refrigerator) and we need a way to specify which part we want the robot to manipulate. Specifically, for those tasks that have both *handle* and *door/drawer/lid*, we only give the center of the target *door/drawer/lid* at timestep 0 as the point indicator  $c$ . The policy has to find and manipulate the corresponding *handle* by only point clouds. Compared to ManiSkill 1&2 [25], which uses the oracle segmentation mask for disambiguation, our point indicator  $c$  is easier to acquire in the real world and poses a greater challenge for policy learning. Note that the point indicator  $c$  does not update over time, so it does not leak extra information about the part’s motion.

### 3.5. Part Pose-aware Reward Design

We design dense rewards for policy training because we can access the ground truth (GT) environment state  $s$  in the simulator. Note that the state  $s$  is not available during testing, which makes it hard for direct motion planning. Inspired by ManiSkill [25], we designed a general reward for all tasks. The reward has 4 components:

**Rotation Reward.** GAParts such as *handle* and *button* can only be manipulated at certain angles. The reward  $R_{rot} = \cos(\vec{a}_p \cdot \vec{a}_r)$ , in which  $a_p \in \mathbb{R}^3$  is the y-axis of the part bounding box and  $a_r \in \mathbb{R}^3$  is the y-axis of robot gripper. When the gripper is vertical to the face of the target part, the reward reach the maximum value.

**Distance Reward.** The reward is the negative value of the distance from the center of two tips on gripper  $c_g \in \mathbb{R}^3$  to the handle center  $c_h \in \mathbb{R}^3$ . We also add a discount factor  $d_f \sim d_f(b_{GAParts})$  to reduce the distance reward after the target part has been removed. The intuition is that, for example, when the target door has been opened large enough, our humans won’t tightly grasp the handle in a rigid hand pose. So, we won’t encourage the gripper to grasp the handle as strictly as before either. Finally,  $R_{dist} = -d_f \|c_g - c_h\|_2$ . **Part Moving Reward.** The reward is computed by the movement of the target GAPart, encouraging the robot to move the part to the success state. For example, for **OpenDoor**, it’s the radian degree of the open angle; for **OpenDrawer**, it’s the open length. We formulate it as  $R_{pose}$ .

**Tips closure Reward.** The reward encourages tips to close for stable grasping. It’s computed by the distance of two fingertips on grippers. We formulate it as  $R_{tips}$ .

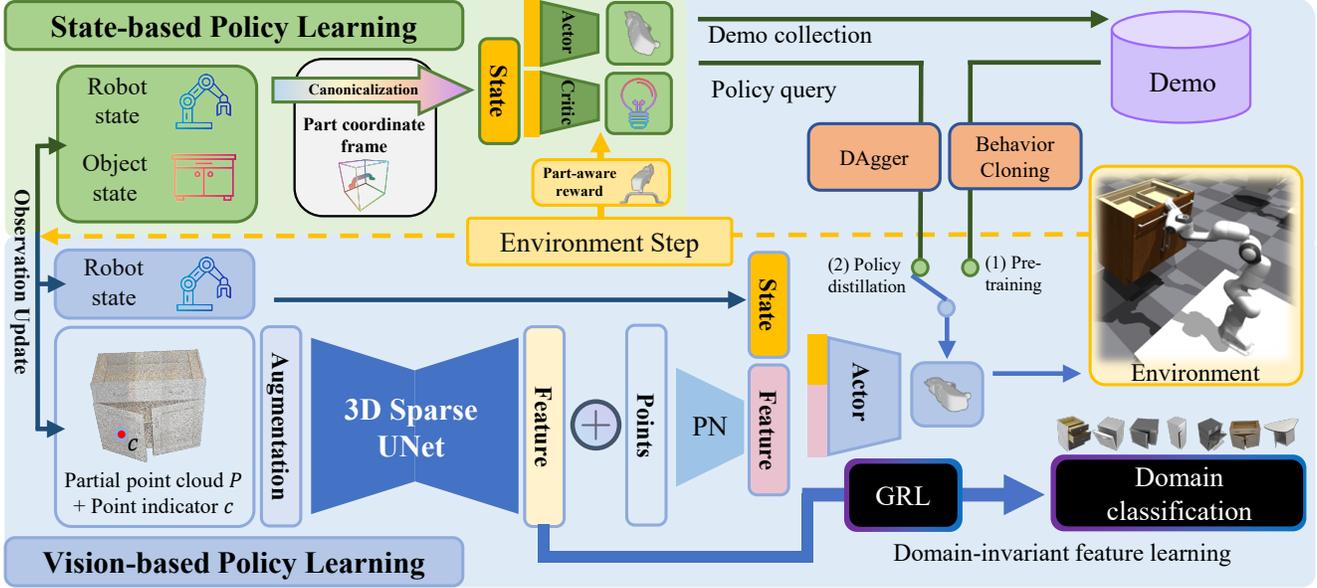


Figure 3. **Our Pipeline.** We first train state-based expert policy using our proposed canonicalization to the part coordinate frame and the part-aware reward. We then use the learned expert to collect demonstrations for pre-training the vision-based policy by behavior cloning. After pre-training, we train the vision-based policy to imitate the state-based expert policy using DAGger. We also introduce several point cloud augmentation techniques to boost the generalization ability. For the vision backbone, we introduce 3D Sparse-UNet which has a large expression capability. Furthermore, we introduced an extra domain adversarial learning module for better cross-category generalization.

The whole reward can be written as:

$$\lambda_r R_{\text{rot}} + \lambda_d R_{\text{dist}} + \lambda_p R_{\text{pose}} + \lambda_t R_{\text{tips}}$$

We use one set of hyper-parameters for each task and the specific value is shown in the appendix.

## 4. Method

Our *PartManip* benchmark is very difficult and thus simple methods can fail dramatically. Our benchmark requires the learned policy not only to manipulate multiple objects from partial visual observation but also to generalize the manipulation skill to unseen object categories. Directly applying state-of-the-art RL algorithms (*e.g.*, PPO [35]) on our benchmark to learn a vision-based policy cannot perform well, possibly due to the unstable RL training process. Therefore, we need specific designs to tackle the difficulty.

We start with expert policy learning using oracle environment state as input. Thanks to the rich part annotations in our benchmark, we propose a novel part-canonicalized strategy for policy learning in Sec. 4.1, which greatly improves the performance and generalization ability.

Given the state-based expert, we then introduce a knowledge distillation strategy to learn a vision-based student policy in Sec. 4.2, which takes input from the partial observation  $o$  instead of the oracle environment state  $s$ . Our State-to-Vision Distillation training strategy also enables visual

observation augmentation to boost the generalization ability. We thus introduce our point cloud augmentation technique here.

Additionally, due to the wide variety of objects and the inherent difficulty in processing visual input, it is crucial to use a backbone with large expression capability. To this end, we propose a Sparse Unet-based backbone architecture [16] for policy learning in Sec. 4.3, which offers superior feature extraction performance.

Last but not least, to overcome the challenges presented by cross-category task settings, we present a domain adversarial training strategy [8, 9, 11, 22] in Sec. 4.4 for our visual feature learning. This strategy improves the generalizability of the policy, particularly on novel object categories.

### 4.1. Part-Canonicalized State-Based RL

Compared to the partial visual observation  $o$ , using the oracle environment state  $s$  as input can greatly reduce the difficulty of policy learning because the network will not be distracted by visual feature extraction and can focus on learning manipulation skills. So we start with state-based expert policy learning using a state-of-the-art RL algorithm Proximal Policy Optimization (PPO).

PPO is a popular on-policy RL algorithm, which proposes a few first-order tricks to stabilize the training process. The stability is ensured by keeping the new policy close to the old one by gradient clipping or KL penalty. PPO trains an actor and a critic, and the objective to maximize

can be formulated as

$$\mathcal{L} = \mathbb{E} \left[ \frac{\pi_{\theta}(a|s)}{\pi_{\theta_{old}}(a|s)} \hat{A} - \epsilon KL[\pi_{\theta_{old}}(\cdot|s), \pi_{\theta}(\cdot|s)] \right] \quad (1)$$

in which  $\theta$  is the network parameters,  $\epsilon$  is a hyper-parameter and  $\hat{A}$  is the estimated advantage using GAE [34]. Thanks to the GPU-parallel data sampling in Isaac Gym [23], PPO can converge fast and stably, so we choose it as our baseline and develop on it.

In our scenario, we find that the relative pose between the robot gripper and the target GPart ( *e.g.*, *handle/button*) is more critical than the absolute pose. Take the task of **OpenDoor** as an example, if the relative poses between two *handles* and two grippers are similar, we can use a similar strategy to manipulate them. It doesn't matter whether these *handles* are on a microwave or a oven. It also doesn't matter what the absolute poses of the gripper and *handles* are.

Therefore, we propose to transform all the coordinate-dependent states from world space to the target GPart's canonical space. Inspired by the normalized coordinate space (NOCS [44]) for category-level object pose estimation, we define the canonical space of each GPart category similarly using the part bounding box annotations provided by GPartNet [53]. For each GPart category, we choose the center of the part bounding box as the origin point, and three axes align to the three vertical edges of the bounding box. Here the scale is ignored because it can change the size of the robot gripper and bring potential harm. For each frame  $t$ , we can use the target GPart's pose (in the format of bounding box  $b_{GPart}^t$ ) to transform all the coordinate-dependent states into the GPart's canonical space.

This part-canonicalized strategy can transform all the absolute poses into relative poses to the target GPart, and thus reduce the variance of input. In our experiments in Sec. 5, we show that this strategy can greatly improve the generalization ability of the learned policy, and achieve a high success rate even on unseen object categories.

## 4.2. Augmented State-to-Vision Policy Distillation

**Dagger-based State-to-Vision Distillation.** The learned expert cannot be directly used in a realistic setting because in the real world the oracle environment state  $s$  is often unknown and we can only acquire the partial observation  $o$ . However, we can leverage the state-based expert knowledge to ease the learning process of the vision-based student. This technique is often called knowledge distillation [14].

One popular and simple method for knowledge distillation is Behavior Cloning (BC). BC requires the expert to collect an offline dataset of observation-action pairs with success trajectories. The objective of BC is to minimize the difference between the action label and the student policy output. Although it sounds reasonable and straightforward,

it can suffer from accumulating error over time because the observation distribution of the student policy is different from the one of the offline dataset.

To overcome the problem of accumulating error, we use DAgger [4, 33] for knowledge distillation. DAgger is an online imitation learning algorithm, which can directly train on the observed distribution of the student policy thanks to online interaction. The objective of DAgger is similar to BC except for the data distribution, which can be written as:

$$\mathcal{L}_{DA} = \frac{1}{|D_{\pi_{\theta}}|} \sum_{o, s \in D_{\pi_{\theta}}} \|\pi_{\text{expert}}(s) - \pi_{\text{student}}(o)\|_2 \quad (2)$$

in which  $D_{\pi_{\theta}}$  is the online interacted data sampled from student policy during training.

**Pretraining with BC.** One potential issue we observed in DAgger is that a randomly initialized policy may perform poorly in the first few iterations, stepping out of the expert's input distribution. Since the expert policy has not encountered those out-of-distribution states, it may not know how to act correctly to finish the task. So using the output of expert policy in such states for supervision can slow down the training process of students, leading to worse performance.

To address this issue, we use the learned expert to collect offline demonstrations and pre-train the student policy with BC. Although pure offline BC cannot be superior to pure online DAgger, BC is a good choice for initialization and can boost the further DAgger training process. Using this pre-training technique along with DAgger, we employ two different ways to make full use of the expert knowledge.

**Observation Augmentations.** Thanks to our state-to-vision distillation method, the student policy will not suffer from the noisy gradient in RL and thus can make full use of its network capability. To improve the generalization ability of the learned policy, a promising way is to enlarge the training distribution of the object geometry and appearance. Therefore, we propose to add augmentations on point clouds during the DAgger training process. These augmentations mainly include point cloud jittering and color replacement. Please see the appendix for more details.

## 4.3. 3D Sparse Unet-based Backbone

In order to effectively process complex and diverse visual input for cross-category generalization, it is essential to have an expressive backbone. To this end, we utilize the widely-used backbone, 3D Sparse-UNet [16], which has a stronger expression capacity compared to PointNet [28] and PointNet++ [29]. Sparse-UNet is often employed in state-of-the-art methods such as PointGroup [19] and SoftGroup [43]. However, a significant drawback of Sparse-UNet is its slow running speed. In order to address this issue, we introduce several algorithms (such as batch voxelization for point clouds and high parallelization of sparse convolution)

to optimize the implementation of Sparse-UNet. These modifications greatly speed up the forward and backward processes of the network, resulting in a runtime that is over 1.4 times faster than the best existing implementation. Additional details can be found in the appendix.

As illustrated in Fig. 3, the partial colored point cloud  $P \in \mathbb{R}^{N \times 6}$  is used as input to Sparse-UNet, which extracts per-point features  $F \in \mathbb{R}^{N \times C}$ . These per-point features are then processed by a small PointNet [28] and concatenated with the robot states to serve as input to the actor.

#### 4.4. Domain-invariant Feature Learning

Inspired by [8, 9, 11, 22], we introduce the Gradient Reverse Layer (GRL) and a domain classifier for domain-invariant feature learning, which improves the generalization ability across object categories, especially for unseen categories. The domain classifier takes the extracted visual features from the backbone as input to distinguish domains, *i.e.* object categories in our setting, while the GRL negates the gradient and encourages the backbone to fool the classifier and thus extract domain-invariant features. For a target part with mask  $M_i$ , we first query its feature  $F_{M_i}$  from the whole feature map  $F$ . Then a classifier  $\mathcal{D}$  with a small Sparse-UNet backbone and two MLPs takes it as input and classifies the domain label. The loss can be written as :

$$\mathcal{L}_{adv} = \mathcal{L}_{cls}(\mathcal{D}(F_{M_i}), y_i^{cate.}) \quad (3)$$

where  $y_i^{cate.}$  is the ground truth domain label (*i.e.*, category label).

### 5. Experiments

#### 5.1. Evaluation Settings and Main Results

Because our *PartManip* benchmark emphasizes the experiment settings to be realistic, all the algorithms are evaluated using partial observation  $o$  as input except those experts in Tab. 5. We use the task success rate as our main evaluation metric. To reduce the evaluation noise, we conduct each experiment 3 times using random seeds and report the mean performance as well as the variance. See appendix for more training details.

The main results of our method are shown in Tab. 3. We can see that **CloseDoor** and **CloseDrawer** are relatively easy and our method can achieve a high success rate, even on unseen object categories. The other four tasks are more challenging and the performance drops on the unseen categories, especially for **OpenDoor** and **OpenDrawer**.

#### 5.2. Comparison with Baselines

To demonstrate the effective of our method, we conduct several relative baselines on our *PartManip* benchmark. These baselines include a popular RL algorithm PPO [35], an affordance-based interaction method W2A [24], a novel

Success rate (%)	Training Set	Val-Intra Set	Val-Inter Set
CloseDoor	88.7±1.0	88.4±2.9	87.0±1.6
CloseDrawer	99.6±0.6	97.9±2.1	98.6±1.2
OpenDoor	68.4±1.1	57.2±0.4	49.1±1.5
OpenDrawer	82.3±2.1	78.7±2.0	54.7±4.2
PressButton	89.6±2.9	79.6±4.2	66.6±4.2
GraspHandle	79.8±2.4	70.0±2.4	56.4±2.9

Table 3. **Our Main Results on *PartManip* benchmark.**

Success rate (%)	OpenDoor			OpenDrawer		
	Training	Val-Intra	Val-Inter	Training	Val-Intra	Val-Inter
PPO [35]	4.5±3.8	4.9±3.5	0.2±0.2	8.9±2.8	11.3±2.8	3.3±1.6
ILAD [48]	13.3±4.9	6.3±2.5	5.0±4.1	18.7±3.6	18.3±2.9	3.3±2.9
Where2act [24]	25.4±0.1	23.4±0.0	15.2±0.1	39.6±0.2	37.2±0.2	20.5±0.1
SilverBullet3D [26]	54.6±2.5	49.9±1.0	26.9±2.2	77.7±3.3	60.0±2.0	31.2±5.1
Shen <i>et. al</i> [38]	1.5±0.6	0.3±0.6	2.3±4.0	9.7±0.5	18.0±2.2	2.7±1.9
Wu <i>et. al</i> [46]	45.9±2.3	34.1±3.8	17.8±1.4	70.5±2.2	53.3±3.3	28.5±2.4
Dubois <i>et. al</i> [5]	35.4±4.4	25.1±2.1	1.3±1.0	61.4±4.3	38.3±2.0	2.7±1.6
<b>Ours</b>	<b>68.4±1.1</b>	<b>57.2±0.4</b>	<b>49.1±1.5</b>	<b>82.3±2.1</b>	<b>78.7±2.0</b>	<b>54.7±4.2</b>

Table 4. **Comparison with Baselines.**

vision-based policy training method ILAD [48], and winners [5, 26, 38, 46] in ManiSkill Challenge [25]. Due to the page limit, we only show the results of two representative tasks, *i.e.*, **OpenDoor** and **OpenDrawer**, in our main paper. See the appendix for the performance of the other four tasks.

As shown in Tab. 4, we can see that although most baselines can work to some degree on the training set, the performance can drop dramatically on intra- and especially inter-category validation set. In contrast, our method can perform consistently better on all evaluation set without intense performance drop, which indicates the great generalization ability of our method.

#### 5.3. Ablation Study

The ablation studies are shown in Tab. 5. For state-based policy learning, our proposed part-based canonicalization can significantly improve performance on all three evaluation sets, especially for the inter-category validation set. Take **OpenDoor** as an example, the policy trained with part-based canonicalization can outperform the one without by 14.4%, 12.3%, and 27.3% respectively on three sets.

For vision-based policy learning, we analyze each component below (row number is counted from the visual policy):

1) (Row 1-4) Directly using the RL algorithm PPO to learn the policy performs poorly. We deduce that the RL gradient is noisy and can ruin the learning of a vision backbone, especially for a large network (*e.g.*, Sparse-UNet). DAgger can greatly alleviate this problem and perform well on the training set, but it also suffers from overfitting and thus is lack of strong cross-category generalization ability.

3) (Row 3-6) Sparse-UNet backbone has a better expressive capacity but may overfit to the training set. Using aug-

Success rate (%)	Canon	DAgger	Augm	S-Unet	Pretrain	DomAdv	Opening Door			Opening Drawer		
							Training	Val-Intra	Val-Inter	Training	Val-Intra	Val-Inter
State-based Expert	✓						67.8±3.4	50.2±1.9	23.4±3.9	71.5±2.1	62.5±2.3	37.5±5.2
							<b>82.2±0.2</b>	<b>62.5±2.6</b>	<b>50.7±4.1</b>	<b>92.7±0.9</b>	<b>88.1±1.0</b>	<b>63.4±2.4</b>
Vision-based Student				✓			4.5±3.8	4.9±3.5	0.2±0.2	8.9±2.8	11.3±2.8	3.3±1.6
							0.8±0.5	0.4±0.2	0.0±0.0	5.9±2.3	3.9±0.6	1.0±0.2
		✓					60.3±0.7	49.2±1.1	31.5±2.9	70.9±0.6	62.0±1.1	42.7±1.8
		✓		✓			66.8±2.7	50.2±1.7	28.8±2.1	77.4±2.7	61.9±3.0	36.4±3.3
		✓	✓				60.0±1.7	54.4±2.3	40.2±3.9	69.7±2.4	69.8±2.5	49.0±2.1
		✓	✓	✓			65.5±1.5	55.9±2.7	41.7±2.5	74.6±3.4	63.8±4.7	49.1±3.4
		✓	✓			✓	61.1±3.3	55.0±1.2	37.8±2.9	71.9±3.3	72.2±3.5	50.3±2.6
		✓	✓	✓	✓	✓	<b>71.2±1.8</b>	57.0±0.7	37.2±1.2	82.0±3.3	73.8±2.9	48.8±4.5
	✓	✓	✓	✓	✓	68.4±1.1	<b>57.2±0.4</b>	<b>49.1±1.5</b>	<b>82.3±2.1</b>	<b>78.7±2.0</b>	<b>54.7±4.2</b>	

Table 5. **Ablation Study.** Canon = Part pose canonicalization for the input states; Augm = Augmentation; S-Unet = Sparse-UNet; Pretrain = Pretraining with expert demonstration; DomAdv = Domain adversarial learning.

mentation can alleviate the overfit, achieving much better generalization ability.

5) (Row 5-8) Pretraining using the demo collected by the expert can provide a better initialization for the student, ease the problem of being out of expert distribution at the beginning of DAgger, and improve performance.

6) (Row 8,9) Our domain adversarial learning can further boost the performance, especially on previously unseen instances or categories. We reason that our adversarial training strategy helps domain-invariant feature extraction and thus helps robotics policy learning.

#### 5.4. Real-World Experiments

Finally, we validate our method in the real world, as shown in Fig. 4. We use a single RGB-D camera (Okulo) to capture visual observation and a Franka Emika robotic arm for manipulation. We try to directly apply the learned policy in simulation to the real world but find the success rate is much lower, due to the huge sim2real gap of the point cloud and the physics. Therefore, to minimize the sim2real gap, we use a digital twin system [12, 49] for the real-world experiment. The input of our method included: 1) point cloud from the simulator and 2) agent state from the real world. We then apply the output action of our policy to the robotic arm both in the simulator and the real world. Note that the testing object is unseen during policy learning. Experiments show that our trained model can successfully transfer to the real world. See appendix for more details.

#### 6. Limitation

Although our proposed method for cross-category object manipulation has already outperformed previous work by a large margin in simulation, there is still much room for improvement. It is worth studying how to further improve the performance on unseen object instances and categories. Another limitation can be the huge sim2real gap between the point cloud and the physics. We hope that our *PartManip*



Figure 4. **Real World Experiment.**

benchmark can provide a good starting point for future research on generalizable manipulation policy learning.

#### 7. Conclusion

In this work, we introduce a large-scale part-based cross-category object manipulation benchmark *PartManip*, with six tasks in realistic settings. To tackle the challenging problem of the generalizable vision-based policy learning, we first introduce a carefully designed state-based part-aware expert learning method, and then a well-motivated state-to-vision distillation process, as well as a domain generalization technique to improve the cross-category generalization ability. Through extensive experiments in simulation, we show the superiority of our method over previous works. We also demonstrate the performance in the real world.

#### 8. Acknowledgements

This work is supported in part by the National Key R&D Program of China (2022ZD0114900).

## References

- [1] Michael Ahn, Anthony Brohan, Noah Brown, Yevgen Chebotar, Omar Cortes, Byron David, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, et al. Do as i can, not as i say: Grounding language in robotic affordances. *arXiv preprint arXiv:2204.01691*, 2022. [2](#)
- [2] Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. Solving rubik’s cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019. [2](#)
- [3] Felix Burget, Armin Hornung, and Maren Bennewitz. Whole-body motion planning for manipulation of articulated objects. In *2013 IEEE International Conference on Robotics and Automation*, pages 1656–1662. IEEE, 2013. [3](#)
- [4] Tao Chen, Jie Xu, and Pulkit Agrawal. A system for general in-hand object re-orientation. In *Conference on Robot Learning*, pages 297–307. PMLR, 2022. [3](#), [6](#)
- [5] Fabian Dubois, Eric Platon, and Tom Sonoda. Improving performance on the maniskill challenge via super-convergence and multi-task learning. In *ICLR 2022 Workshop on Generalizable Policy Learning in Physical World*, 2022. [7](#), [13](#)
- [6] Ben Eisner, Harry Zhang, and David Held. Flowbot3d: Learning 3d articulation flow to manipulate articulated objects. *arXiv preprint arXiv:2205.04382*, 2022. [3](#)
- [7] Hao-Shu Fang, Chenxi Wang, Minghao Gou, and Cewu Lu. Graspnet-1billion: A large-scale benchmark for general object grasping. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11444–11453, 2020. [2](#)
- [8] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International conference on machine learning*, pages 1180–1189. PMLR, 2015. [2](#), [5](#), [7](#)
- [9] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1):2096–2030, 2016. [2](#), [5](#), [7](#)
- [10] Wei Gao and Russ Tedrake. kpm-sc: Generalizable manipulation planning using keypoint affordance and shape completion. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6527–6533. IEEE, 2021. [2](#)
- [11] Haoran Geng, Helin Xu, Chengyang Zhao, Chao Xu, Li Yi, Siyuan Huang, and He Wang. Gapartnet: Cross-category domain-generalizable object perception and manipulation via generalizable and actionable parts. *arXiv preprint arXiv:2211.05272*, 2022. [2](#), [3](#), [5](#), [7](#)
- [12] Yiran Geng, Boshi An, Haoran Geng, Yuanpei Chen, Yaodong Yang, and Hao Dong. End-to-end affordance learning for robotic manipulation. *arXiv preprint arXiv:2209.12941*, 2022. [3](#), [8](#)
- [13] Dibya Ghosh, Jad Rahme, Aviral Kumar, Amy Zhang, Ryan P Adams, and Sergey Levine. Why generalization in rl is difficult: Epistemic pomdps and implicit partial observability. *Advances in Neural Information Processing Systems*, 34:25502–25515, 2021. [2](#)
- [14] Jianping Gou, Baosheng Yu, Stephen J Maybank, and Dacheng Tao. Knowledge distillation: A survey. *International Journal of Computer Vision*, 129:1789–1819, 2021. [6](#)
- [15] Minghao Gou, Hao-Shu Fang, Zhanda Zhu, Sheng Xu, Chenxi Wang, and Cewu Lu. Rgb matters: Learning 7-dof grasp poses on monocular rgbd images. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13459–13466. IEEE, 2021. [2](#)
- [16] Benjamin Graham and Laurens van der Maaten. Submanifold sparse convolutional networks. *arXiv preprint arXiv:1706.01307*, 2017. [2](#), [5](#), [6](#)
- [17] Jiayuan Gu, Fanbo Xiang, Xuanlin Li, Zhan Ling, Xiqiang Liu, Tongzhou Mu, Yihe Tang, Stone Tao, Xinyue Wei, Yunchao Yao, et al. Maniskill2: A unified benchmark for generalizable manipulation skills. *arXiv preprint arXiv:2302.04659*, 2023. [2](#), [3](#)
- [18] Advait Jain and Charles C Kemp. Pulling open novel doors and drawers with equilibrium point control. In *2009 9th IEEE-RAS international conference on humanoid robots*, pages 498–505. IEEE, 2009. [3](#)
- [19] Li Jiang, Hengshuang Zhao, Shaoshuai Shi, Shu Liu, Chi-Wing Fu, and Jiaya Jia. Pointgroup: Dual-set point grouping for 3d instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4867–4876, 2020. [6](#)
- [20] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, et al. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning*, pages 651–673. PMLR, 2018. [3](#)
- [21] Robert Kirk, Amy Zhang, Edward Grefenstette, and Tim Rocktäschel. A survey of generalisation in deep reinforcement learning. *arXiv preprint arXiv:2111.09794*, 2021. [2](#)
- [22] Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C Kot. Domain generalization with adversarial feature learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5400–5409, 2018. [2](#), [5](#), [7](#)
- [23] Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, et al. Isaac gym: High performance gpu-based physics simulation for robot learning. *arXiv preprint arXiv:2108.10470*, 2021. [3](#), [6](#)
- [24] Kaichun Mo, Leonidas J Guibas, Mustafa Mukadam, Abhinav Gupta, and Shubham Tulsiani. Where2act: From pixels to actions for articulated 3d objects. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6813–6823, 2021. [3](#), [7](#), [13](#)
- [25] Tongzhou Mu, Zhan Ling, Fanbo Xiang, Derek Yang, Xuanlin Li, Stone Tao, Zhiao Huang, Zhiwei Jia, and Hao Su. ManiSkill: Generalizable Manipulation Skill Benchmark with Large-Scale Demonstrations. In *Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2021. [2](#), [3](#), [4](#), [7](#), [13](#), [14](#)

- [26] Yingwei Pan, Yehao Li, Yiheng Zhang, Qi Cai, Fuchen Long, Zhaofan Qiu, Ting Yao, and Tao Mei. Silver-bullet-3d at maniskill 2021: Learning-from-demonstrations and heuristic rule-based methods for object manipulation, 2022. [7](#), [13](#)
- [27] L Peterson, David Austin, and Danica Kragic. High-level control of a mobile manipulator for door opening. In *Proceedings. 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2000)*(Cat. No. 00CH37113), volume 3, pages 2333–2338. IEEE, 2000. [3](#)
- [28] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas 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 (CVPR)*, pages 652–660, 2017. [6](#), [7](#)
- [29] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *Proceedings of Advances in Neural Information Processing Systems (NeurIPS)*, 2017. [6](#)
- [30] Ilija Radosavovic, Tete Xiao, Stephen James, Pieter Abbeel, Jitendra Malik, and Trevor Darrell. Real-world robot learning with masked visual pre-training. *arXiv preprint arXiv:2210.03109*, 2022. [3](#)
- [31] Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. *arXiv preprint arXiv:1709.10087*, 2017. [2](#)
- [32] Scott Reed, Konrad Zolna, Emilio Parisotto, Sergio Gomez Colmenarejo, Alexander Novikov, Gabriel Barth-Maron, Mai Gimenez, Yury Sulsky, Jackie Kay, Jost Tobias Springenberg, et al. A generalist agent. *arXiv preprint arXiv:2205.06175*, 2022. [2](#)
- [33] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011. [2](#), [6](#), [13](#)
- [34] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015. [6](#)
- [35] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017. [5](#), [7](#), [13](#)
- [36] Younggyo Seo, Danijar Hafner, Hao Liu, Fangchen Liu, Stephen James, Kimin Lee, and Pieter Abbeel. Masked world models for visual control. *arXiv preprint arXiv:2206.14244*, 2022. [3](#)
- [37] Younggyo Seo, Kimin Lee, Stephen L James, and Pieter Abbeel. Reinforcement learning with action-free pre-training from videos. In *International Conference on Machine Learning*, pages 19561–19579. PMLR, 2022. [3](#)
- [38] Hao Shen, Weikang Wan, and He Wang. Learning category-level generalizable object manipulation policy via generative adversarial self-imitation learning from demonstrations. *arXiv preprint arXiv:2203.02107*, 2022. [2](#), [7](#), [13](#)
- [39] Aravind Srinivas, Michael Laskin, and Pieter Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. *arXiv preprint arXiv:2004.04136*, 2020. [3](#)
- [40] Adam Stooke, Kimin Lee, Pieter Abbeel, and Michael Laskin. Decoupling representation learning from reinforcement learning. In *International Conference on Machine Learning*, pages 9870–9879. PMLR, 2021. [3](#)
- [41] Martin Sundermeyer, Arsalan Mousavian, Rudolph Triebel, and Dieter Fox. Contact-graspnet: Efficient 6-dof grasp generation in cluttered scenes. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13438–13444. IEEE, 2021. [2](#)
- [42] Yusuke Urakami, Alec Hodgkinson, Casey Carlin, Randall Leu, Luca Rigazio, and Pieter Abbeel. Doorgym: A scalable door opening environment and baseline agent. *arXiv preprint arXiv:1908.01887*, 2019. [3](#)
- [43] Thang Vu, Kookhoi Kim, Tung M Luu, Thanh Nguyen, and Chang D Yoo. Softgroup for 3d instance segmentation on point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2708–2717, 2022. [6](#)
- [44] He Wang, Srinath Sridhar, Jingwei Huang, Julien Valentin, Shuran Song, and Leonidas J Guibas. Normalized object coordinate space for category-level 6d object pose and size estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2642–2651, 2019. [6](#)
- [45] Yian Wang, Ruihai Wu, Kaichun Mo, Jiaqi Ke, Qingnan Fan, Leonidas J Guibas, and Hao Dong. Adaafford: Learning to adapt manipulation affordance for 3d articulated objects via few-shot interactions. In *European Conference on Computer Vision*, pages 90–107. Springer, 2022. [3](#)
- [46] Kun Wu, Yinuo Zhao, Zhiyuan Xu, Zhen Zhao, Pei Ren, Zhengping Che, Chi Harold Liu, Feifei Feng, and Jian Tang. A minimalist ensemble method for generalizable offline deep reinforcement learning. In *ICLR 2022 Workshop on Generalizable Policy Learning in Physical World*, 2022. [7](#), [13](#)
- [47] Ruihai Wu, Yan Zhao, Kaichun Mo, Zizheng Guo, Yian Wang, Tianhao Wu, Qingnan Fan, Xuelin Chen, Leonidas Guibas, and Hao Dong. VAT-mart: Learning visual action trajectory proposals for manipulating 3d articulated objects. In *International Conference on Learning Representations*, 2022. [3](#), [13](#)
- [48] Yueh-Hua Wu, Jiashun Wang, and Xiaolong Wang. Learning generalizable dexterous manipulation from human grasp affordance. *arXiv preprint arXiv:2204.02320*, 2022. [3](#), [7](#), [13](#)
- [49] Kaishu Xia, Christopher Sacco, Max Kirkpatrick, Clint Saidu, Lam Nguyen, Anil Kircaliali, and Ramy Harik. A digital twin to train deep reinforcement learning agent for smart manufacturing plants: Environment, interfaces and intelligence. *Journal of Manufacturing Systems*, 58:210–230, 2021. [8](#)
- [50] Yinzhen Xu, Weikang Wan, Jialiang Zhang, Haoran Liu, Zikang Shan, Hao Shen, Ruicheng Wang, Haoran Geng, Yijia Weng, Jiayi Chen, et al. Unidexgrasp: Universal robotic dexterous grasping via learning diverse proposal generation and goal-conditioned policy. *arXiv preprint arXiv:2303.00938*, 2023. [3](#)

- [51] Zhenjia Xu, Zhanpeng He, and Shuran Song. Universal manipulation policy network for articulated objects. *IEEE Robotics and Automation Letters*, 7(2):2447–2454, 2022. 3
- [52] Zhenjia Xu, Beichun Qi, Shubham Agrawal, and Shuran Song. Adagrasp: Learning an adaptive gripper-aware grasping policy. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4620–4626. IEEE, 2021. 2
- [53] Zetong Yang, Yanan Sun, Shu Liu, Xiaoyong Shen, and Ji-aya Jia. Ipod: Intensive point-based object detector for point cloud, 2018. 2, 6, 11
- [54] Denis Yarats, Amy Zhang, Ilya Kostrikov, Brandon Amos, Joelle Pineau, and Rob Fergus. Improving sample efficiency in model-free reinforcement learning from images. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 10674–10681, 2021. 3
- [55] Fangyi Zhang, Jürgen Leitner, Michael Milford, Ben Ufford, and Peter Corke. Towards vision-based deep reinforcement learning for robotic motion control. *arXiv preprint arXiv:1511.03791*, 2015.
- [56] Yan Zhao, Ruihai Wu, Zhehuan Chen, Yourong Zhang, Qingnan Fan, Kaichun Mo, and Hao Dong. Dualafford: Learning collaborative visual affordance for dual-gripper object manipulation. *arXiv preprint arXiv:2207.01971*, 2022. 3

## A. More Details about Our Benchmark

### A.1. Objects and Robot Assets

We select 494 object instances from GPartNet [53] dataset. GPartNet provides large-scale articulated objects with rich annotations. For doors and drawers, our benchmark requires a handle on it. For the button, there is no constraint. Because there are too many buttons on some remote or phone, we randomly select a maximum of 5 buttons on each object. The total benchmark contains 494 objects and 1432 different target parts.

### A.2. Environment Settings

We carefully set the environment parameters in Isaacgym. To simulate real-world physics, we set the contact offset = 1e-3, which means that our gripper is harder to manipulate the part by scratching or rubbing the edge of the part. Also, we set a slight recovery force (= 0.1) to avoid the agent moving the part to a successful state by slightly touching it. The stiffness of the cabinet dof (slider joint for drawer and button, rotation joint for door) is set to 20. The damping of the cabinet dof is set to 200. The friction coefficient of cabinet dof is set to 5.

For robot controlling, we use pos control mode. This means that we need to input each joint angle of the Franka arm to the network at each step. We find that in our tasks, using pos control is easier for imitation learning.

### A.3. Reward Weight

For each task, we tune the reward weight to train a human-like policy. We list rotation weight  $\lambda_r$ , handle distance weight  $\lambda_d$ , part moving weight  $\lambda_p$ , and tips closure weight  $\lambda_t$  in Table 7.

**rotation weight.** For the closing task and pressing button task, we don't need the gripper to grasp the handle, so we set  $\lambda_r = 0$ . For the other tasks, we set it to 0.2.

**handle distance weight.** The value of  $\frac{\lambda_d}{\lambda_p}$  is important for opening task. If the value is too small, the policy wouldn't learn to open the part using the handle. If the value is too large, the policy wouldn't try to open the door, but only learn to grasp the handle. The  $\lambda_{df}$  is set to non-zero for the opening task because after the part is opened, the gripper can move away from the handle.

**part moving weight.** For the grasping task, we don't require the agent to manipulate part, so we set  $\lambda_p = 0$ .

**tip closure weight.** For the opening and closing tasks, we focus on using the handle to finish the task but don't require the final grasp pose of the gripper. So we set  $\lambda_t = 0$ .

### A.4. Initialization and Success Criteria

For each task, we initialize the gripper at a certain distance (*i.e.*, 50cm) away from the center of the target part,

facing the object. The initialization of objects and the success criteria for each task are shown below:

**OpenDoor:** The door should be opened to more than 30 degrees from the initial closed state.

**OpenDrawer:** The drawer should be opened to more than 20% of the maximum opening length from the initial closed state.

**CloseDoor:** The door should be closed to less than 1 degree from the initial opening angles of 45 degrees.

**CloseDrawer:** The drawer should be closed from the initial opening length of 30 cm to less than 1 cm.

**PressButton:** The button should be pressed for more than 50% of the maximum pressing depth.

**GraspHandle:** The robot should close two tips to less than a threshold from different sides of the handle, while the center of the two tips is inside the handle bounding box.

## B. More Details about Our Method

Our pseudo code is shown in algorithm 1 and algorithm 2.

For state-based policy training, we update  $E$  epochs in one step. In one epoch, the state and actions buffer  $D_{\pi_\theta}$  is divided in  $B$  minibatch to compute one gradient step. So after sampling once, the network update  $E * B$  time.

For state-to-vision distillation, we use point cloud augmentation. We use point cloud jittering with a distance of 0.1 and a strong color augmentation, which changes the GAParts color to a random color during a specific episode. Although we randomly choose a color, we fix it during one episode. This technique works well and improves performance in the unseen category.

There is a potential problem for expert distillation. The output of the actor can be an arbitrary value in  $\mathbb{R}^n$ . Here  $n$  is the output dimension of the actor. Because we use the pose control, and the joint angle is in the range  $(-\pi, \pi)$ . If the value  $a_i$  in  $i_{th}$  dimension is out of this range, it would shift to  $a'_i$  satisfied  $a'_i = a_i + 2k\pi, k \in \mathbb{Z}$ . Because of this, multiple outputs would correspond to one action in the simulator so the L2 loss of expert action and student action is not positively associated with the similarity between expert action and student action. To tackle this problem, we add an additional Tanh layer and scale the action to  $(-\pi, \pi)$ . We use the scaled action to compute dagger loss and update the network.

## C. More Details about the Experiment Setting

### C.1. Training Details

We train our state-based policy on Nvidia GeForce RTX 2080Ti for 6 hours. For each task, we use all of the data in our dataset claimed in the paper. The PPO hyperparameter is shown in Table 8.

---

### Algorithm 1 State-based Expert Training

---

**Input:** robot states  $S$ , handle bounding box  $b_{handle}$ , part bounding box  $b_{part}$ , state  $s = (S, b_{part}, b_{handle})$ , policy network  $\theta_p$  (i.e., actor  $\theta_p^a$  and critic  $\theta_p^c$ ),

**for**  $t = 0, 1, 2 \dots$  **do**

Transfer observation to canonical space

Sample trajectories  $D_{\pi_\theta} = \{(s_i, a_i)\}_{i=1}^n$

**for**  $e = 1, 2, \dots, E$  **do** ▷ PPO update

**for**  $b = 1, 2, \dots, B$  **do**

update policy network  $\theta_p$ , according to:  $\mathcal{L}_{RL}$

Select highest success rate  $\theta_p \rightarrow \theta_{\text{expert}}$

**while**  $|D_{\text{demo}}| \leq \text{buffer size}$  **do**

sample trajectory  $t_i$  by  $\theta_{\text{expert}}$

append  $t_i$  to  $D_{\text{demo}}$

---



---

### Algorithm 2 Vision-based Student Training

---

**Input:** partial point cloud  $P \in \mathbb{R}^{N \times 3}$ , robot states  $s$ , vision backbone  $\theta_b$ , policy network  $\theta_p$  (i.e., actor  $\theta_p^a$  and critic  $\theta_p^c$ ), expert policy  $\theta_{\text{expert}}$ , demonstration buffer  $D_{\text{demo}}$

**Pre-training:** update the vision backbone  $\theta_b$  and actor MLPs  $\theta_p^a$ , according to  $\mathcal{L}_{BC}$

**for**  $t = 0, 1, 2 \dots$  **do**

Sample trajectories  $D_{\pi_\theta} = \{(s_i, o_i, a_i)\}_{i=1}^n$

**for**  $e = 1, 2, \dots, E$  **do**

**for**  $b = 1, 2, \dots, B$  **do**

augment point cloud observation  $o_i$  as  $\mathcal{A}(o_i)$

update backbone  $\theta_b$  and the actor of policy network

$\theta_p^a$ , according to:  $\lambda_{\text{DA}} \mathcal{L}_{\text{DA}} + \lambda_{\text{adv}} \mathcal{L}_{\text{adv}}$

---

For actor and critic networks, we use 3 hidden layers of MLP. The hidden layer dimension of the MLP is 512, 512, 64. For vision-based policy, we use the Sparse-Unet backbone. If the input point cloud has more than  $N = 20,000$  points, we first downsample it to 20000 points using FPS (Farthest Point Sampling). Then we voxelize the input point cloud into a  $100 \times 100 \times 100$  voxel grid. The backbone U-Net has an encoder and decoder, both with a depth of 6 (with channels of [16, 32, 48, 64, 80, 96, 112]) and outputs a  $N \times K$  per-point feature  $\mathbf{F}$  where  $K = 16$ . We speed up the 3D Sparse UNet inference speed by introducing batch voxelization for point clouds and high parallelization of sparse convolution, thanks to the latest high-performance third-party code base like open3d and sparse

Because PPO is an on-policy RL algorithm, for each  $N$  step, we update the policy. To leverage the fast convergence of PPO, we want to update as frequently as we can. On the other side, due to the noisy gradient of RL, the batch that is used to compute the gradient should not be too small, which is equal to  $T_{\text{env}} * N/M$ . Here  $T$  is the training environment number, and  $M$  is the minibatch size. Empirically we find that the batch size near to 2000 is fine. For six tasks, due to the number of training data, we choose proper minibatch

	robot state	part bounding box	handle bounding box	Point Cloud	part mask	handle mask
Ours (state-based)	✓	✓	✓			
Where2Act [24]	✓			✓	✓	
VAT-Mart [47]	✓			✓	✓	
Maniskill [25]	✓			✓	✓	✓
Ours (vision-based)	✓			✓		

Table 6. Comparison with Other Methods.

name	opening door	opening drawer	closing door	closing drawer	pressing button	grasping handle
Train environment number	363	246	363	246	215	88
minibatches	2	3	2	3	2	2
nsteps	20	20	20	20	20	40
$\lambda_r$	0.2	0.2	0	0	0	0.2
$\lambda_d$	2	1.3	1	1	1	1
$\lambda_p$	1	1	1	1	100	0
$\lambda_t$	0	0	0	0	10	1
$\lambda_{df}$	1	2	0	0	0	0

Table 7. Task Specific Hyperparameters of State-based Policy Training

task	method	Training Set	ValIntra Set	ValInter Set
Closing Door(%)	Where2act [24]	77.3±0.1	54.6±0.0	51.5±0.2
	PPO [35]	35.5±1.1	37.6±0.9	15.4±0.5
	DAGger [33]	84.5±2.5	79.4±1.1	69.9±2.3
	Ours	<b>88.7±1.0</b>	<b>88.4±2.9</b>	<b>87.0±1.6</b>
Closing Drawer(%)	Where2act [24]	89.9±0.2	90.5±0.1	89.9±0.3
	PPO [35]	69.9±5.9	75.2±2.6	59.3±2.1
	DAGger [33]	95.9±1.2	97.3±1.1	91.5±0.2
	Ours	<b>99.6±0.6</b>	<b>97.9±2.1</b>	<b>98.6±1.2</b>
Pushing Button(%)	Where2act [24]	15.5±0.2	16.2±0.1	19.3±0.3
	PPO [35]	25.5±0.2	21.6±1.1	7.9±5.5
	DAGger [33]	32.8±2.2	<b>41.2±6.6</b>	29.8±1.2
	Ours	<b>89.6±2.9</b>	<b>79.6±4.2</b>	<b>66.6±4.2</b>
Grasping Handle(%)	Where2act [24]	27.7±0.1	25.4±0.2	13.9±0.3
	PPO [35]	15.7±2.2	13.2±0.6	9.9±3.5
	DAGger [33]	45.6±2.2	35.5±2.1	29.8±2.9
	Ours	<b>79.8±2.4</b>	<b>70.0±2.4</b>	<b>56.4±2.9</b>

Table 9. More Results of Method Comparison and Baselines

and nsteps. The task-specific hyperparameters are shown in Table 7.

PPO params	value / type
learning rate	3e-4
optimizer	Adam
gamma	0.99
lambda	0.95
desired kl	0.01
clip range	0.1
entropy coef	0.01
init noise std	1

Table 8. PPO Hyperparameters of Policy Training

## C.2. More Details and Results of the Baselines

For opening the door and drawer, thank the previous exploration, We compare our policy with many baselines.

For [5, 26, 38, 46], they focus on tasks like opening drawers and doors and we can modify their method to our OpenDoor and OpenDrawer tasks. And for the other four tasks, we also compare with some possible methods if they can be easily modified to fit our framework. More results are shown in Table 9.

**PPO [35].** We directly use the PPO algorithm to learn a vision-based policy to handle each task. The detailed PPO parameter and training strategy is the same as the state-based expert training in our method.

**Where2Act [24].** We input the part mask as an extra dimension in our task as a baseline, and others remain the same. We modified the where2act interaction pipeline to finish our tasks. We use a similar pulling motion for the first three tasks and a pushing motion for the fourth task. Giving only a point to indicate the part to be interacted with makes it challenging for where2act to perform proper actions, especially for opening drawers and doors. We thus provide additional information (*i.e.*, the handle center of the target door and drawer), and this method needs to select one point from the given points. Then, after motion direction selection, the action is performed to finish the task. We constrain  $N_{w2a} = 10$  actions to finish these tasks.

**ILAD [48].** Due to we have designed a dense reward in our task, we use our dense reward instead of their extra Q functions to compute the advantage in the third term of *gILAD*. The demonstrations are also collected by expert policy as the GAIL [38] baseline implementation. We don't input part 6D pose into the network as a fair comparison to our method.

**ManiSkill [25] Winners, *i.e.*, Shen et. al [38], SilverBullet3D [26], Wu et. al [46], Dubois et. al [5].** We follow

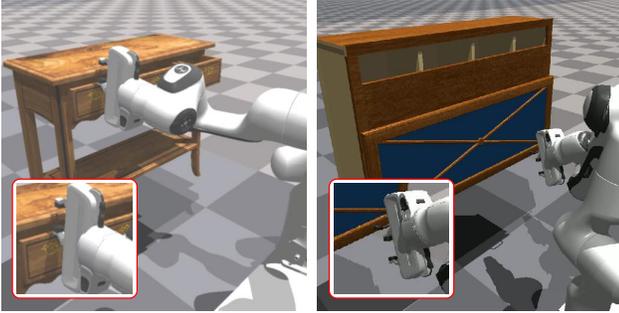


Figure 5. **Failure Cases**

the ManiSkill [25] settings and follow the corresponding policy learning strategy to learn. If the method needs collected demonstrations as input, we provide the demonstrations collected by the state-based expert.

For these baselines, we analyze that their performances are limited due to the distribution shift in behavior cloning, lacking vital information with realistic sensory observation input and noisy reinforcement learning gradient.

### C.3. More Results for a Single Camera Setting

Here we provide more experiment results for a single camera setting. For OpenDoor, the performances are  $36.7 \pm 3.3$ ,  $33.9 \pm 2.4$ ,  $22.6 \pm 3.0$  in the training set, Val-Intra set and Val-Inter set respectively. For OpenDrawer, the performances are  $64.5 \pm 5.5$ ,  $60.2 \pm 4.4$ ,  $17.1 \pm 2.2$  in the training set, Val-Intra set and Val-Inter set respectively.

### C.4. Some Qualitative Results for the Failure Cases

Here we provide some qualitative results for failure cases. In Fig. 5, we show two failure cases. For the left one, the gripper fails to identify the handle and grasps the wrong position due to the thin and flat handle shape (yellow, zoom in to see), while for the right one, the door opening fails later for unstable grasping.

## D. Real Experiment

We use the robot arm (FRANKA) to manipulate previously unseen real objects with only partial point cloud observations. A partial point cloud of the target object instance is acquired from the RGB-D camera (Okulo P1 ToF sensor in our experiments). To set up the interaction environment, we use aruco markers to calibrate the camera sensor and place the object and the robot arm in the proper positions, the same as the trained policy in the simulator. We also provide a point to indicate the part to interact with, just like we did in the simulator. During manipulation, we use the control API provided by the robot arm system to follow the trajectory (a sequence of joint angle and gripper position) and finish the tasks.