# Prepare Before You Act: Learning From Humans to Rearrange Initial States

Yinlong Dai, Andre Keyser, and Dylan P. Losey

***Abstract*— Imitation learning (IL) has proven effective across a wide range of manipulation tasks. However, IL policies often struggle when faced with out-of-distribution observations; for instance, when the target object is in a previously unseen position or occluded by other objects. In these cases, extensive demonstrations are needed for current IL methods to reach robust and generalizable behaviors. But when humans are faced with these sorts of atypical initial states, we often *rearrange* the environment for more favorable task execution. For example, a person might rotate a coffee cup so that it is easier to grasp the handle, or push a box out of the way so they can directly grasp their target object. In this work we seek to equip robot learners with the same capability: *enabling robots to prepare the environment before executing their given policy*. We propose ReSET, an algorithm that takes initial states — which are outside the policy's distribution — and autonomously modifies object poses so that the restructured scene is similar to training data. Theoretically, we show that this two step process (rearranging the environment before rolling out the given policy) reduces the generalization gap. Practically, our ReSET algorithm combines action-agnostic human videos with task-agnostic teleoperation data to i) decide when to modify the scene, ii) predict what simplifying actions a human would take, and iii) map those predictions into robot action primitives. Comparisons with diffusion policies, VLAs, and other baselines show that using ReSET to prepare the environment enables more robust task execution with equal amounts of total training data. See videos at our anonymous website: https://reset2025paper.github.io**

## I. Introduction

Robots should be able to learn tasks from human demonstrations. But learning seemingly simple manipulation tasks can become challenging under minor variations in the environment [1], [2]. Consider a setup in which an agent must grasp a cup (Figure 1). A visuomotor policy can effectively complete this grasping task when the cup is directly observable. However, policies are prone to failure when facing states outside the training distribution; for example, if visual access to the object is obstructed by a box. As shown in Figure 1, when we try to execute a direct policy rollout — and the box is in the way — the robot does not know what to do, leading to critical mistakes like missing the cup.

One standard approach is to try and overcome this challenge is by collecting larger and more diverse sets of training data. Given enough examples, the robot can figure out how to grasp around the box [3], [4]. However, we hypothesize that this is not the most efficient or time-effective way to solve the problem, particularly for long-horizon tasks. Instead, we propose an alternate approach based on how humans go about these inconvenient or unexpected initial states. Imagine a person faced with the scenario in Figure 1 — rather than attempting to reach around the box, humans often simplify the problem by first restructuring the environment. For instance, here a person could move the box out of the way, making the task of grasping the cup much easier to complete. Inspired by the way humans prepare environments, our insight is that:

*We should design policies that restructure the environment, bringing diverse start states into a familiar and manageable distribution before executing the actual task.*

Applying this insight we introduce **ReSET**, **Re**structuring **S**tates for **E**fficient policy **T**raining. Rather than learning one policy that tries to complete the entire task across a broad range of initial states, under ReSET we learn two policies. The first policy is a *reduction policy* that simplifies the initial state by bringing it into a tighter distribution (e.g., moving objects out of the way). The second policy — the default *task policy* — then completes the task from that known distribution (e.g., picking up the cup). By efficiently learning from human teachers how to restructure the environment, we effectively reduce task variability, enabling default policies to perform more complex manipulation tasks without requiring as much total training data.

Overall, we make the following contributions:

**Generalization and Data Efficiency.** We provide theoretical analysis that suggests learning a reduction policy on top of a task policy i) yields a lower generalization gap upper bound, and ii) requires less total training data.

**Flow-Based Approach.** We learn a flow-based reduction policy from action-agnostic human demonstrations and task-agnostic robot play data. This approach i) determines when to rearrange objects, ii) predicts how objects should be manipulated, and iii) maps visual point flows to robot actions.

**Experimental Validation.** We show that ReSET outperforms several state-of-the-art baselines in handling out-of-distribution states across a range of few-shot task settings.

## II. Related Work

**Imitation Learning.** Recent advances in imitation learning enable robots to learn policies capable of executing long-horizon tasks in complex real-world settings [5]–[7]. Diffusion policies [8] model the distribution over actions using denoising diffusion probabilistic models, enabling multimodal behaviors. Flow-matching approaches such as $\pi_0$ [9] learn a continuous velocity field that directly maps noise to actions.

This work is supported in part by NSF Grant #2337884. The authors are members of the Collaborative Robotics Lab (Collab), Dept. of Mechanical Engineering, Virginia Tech, Blacksburg, VA 24061.
Corresponding author's email: daiyinlong@vt.edu

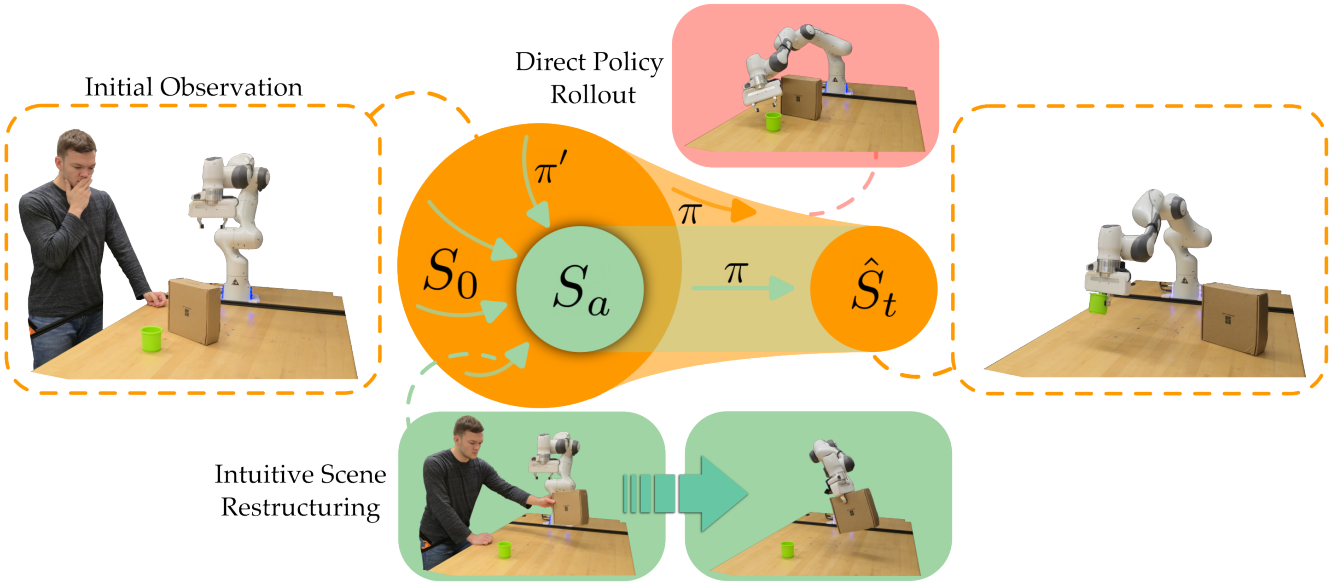


Fig. 1. Robot arm learning to grasp a cup. When encountering an out-of-distribution initial observation $S_0$ (e.g., the cup is obstructed by a box), conventional approaches train on large-scale datasets and attempt to directly rollout the robot policy $\pi$. As we show, this brute-force approach falls short when the robot encounters unexpected initial environment states. By contrast, ReSET first learns a reduction policy $\pi'$ based on how humans intuitively restructure the scene. This reduction policy rearranges objects (e.g., moving the box out of the way) so that the task is easier to perform and has lower state variance ($S_a$). Our approach then executes the default task policy $\pi$ from this simplified state distribution to reach the goal state $\hat{S}_t$.

On the other hand, Vision-Language-Action (VLA) models [7], [10] leverage the generalization capacity of pretrained large vision-language models, unifying vision, language, and actions within a shared feature space. But regardless of whether we use diffusion policies, flow-matching approaches, or VLAs, the model must fundamentally be able to work across a diverse set of initial states (e.g., different object locations, clutter, and visual features). However, few-shot deployment to scenarios outside of the training distribution remains an open challenge [11], [12].

**Scaffolding Approaches.** There are a variety of recent IL works designed to handle out-of-distribution states. We broadly refer to these strategies as "scaffolding." One simple scaffolding approach is to collect diverse data. Works such as [13]–[15] improve generalization by acquiring large-scale real-world datasets. But despite recent advances in data collection methods [16], [17], gathering millions of diverse trajectories still remains difficult and costly.

To work around the issue of manual data collection, methods like [18]–[20] leverage data augmentation or domain randomization and generate synthetic datasets. In practice, real world synthetic data can also be produced using generative models (e.g., diffusion models). However, careful prompting is required to avoid producing unrealistic or confusing samples [18], and ultimately there remains a gap between real data and synthetic images.

The methods most closely related to our proposed approach are object-centric recovery (*OCR*) [21] and dynamics-augmented diffusion policy (*Dynamics-DP*) [22]. *OCR* assumes direct access to object positions and learns a recovery policy by exploiting gradients on the object keypoint manifold within the training data. *Dynamics-DP* learns a dynamics model from robot play data in simulation and employs model-based control to generate augmented trajectories that capture recovery from out-of-distribution states. In practice, we find that both of these approaches assume access to ground-truth states and require significantly more training data that our proposed method. However, the central message conveyed by these works — reshaping the environment into a regime more tractable for policy execution — aligns with our research direction. Building on this perspective, we next provide a theoretical analysis demonstrating that restructuring can improve the performance of learned policies.

## III. The Effects of Anchor States

Returning to our motivating example, consider a robot arm trying to grasp a cup occluded by a box (see Figure 1). Existing approaches typically attempt to reason about out-of-distribution states and directly roll out the policy (i.e., maneuvering around the obstruction and grasping the object without visual access). By contrast, ReSET transforms the initial state into a set of simpler, more tractable intermediate states which we call the **anchor states**. In other words, our approach first reveals the cup by removing the box, and then proceeds to execute the given robot policy.

But is breaking the task into two parts and moving to these anchor states actually data efficient? In this section we show that — given the same amount of training data — leveraging a set of anchor states lowers the theoretical upper bound on the policy's generalization gap. This enables the policy to perform successfully across a wider distribution of initial states. We begin by defining the the generalization gap we aim to bound and the anchor states we try to enforce.

**Definitions.** Let $s \in \mathbb{R}^d$ be the state of the environment. A robot policy $\pi : s \to u$ maps states to robot actions $u \in \mathbb{R}^m$. When executed in an environment with a transition kernel

$P_\theta(s'|s,u)$, the policy yields a Markov transition operator over states [23]:

$$T_{\pi,\theta} = \sum_u P_\theta(s'|s,u)\pi(u|s) \tag{1}$$

where $\theta \in \Theta$ is the environment dynamics. Starting from the initial state distribution $S_0$ and executing $t$ actions, the policy $\pi$ produces a distribution over trajectories through its interaction with the environment. The induced distribution over final states can be expressed as $S_t \sim T^t_{\pi,\theta}(S_0)$. Here $T^t_{\pi,\theta} = \prod_{k=1}^t T_{\pi,\theta}$ denotes the $t$-fold composition of the Markov transition operator, conditioned on both the policy $\pi$ and the environment dynamics $\theta$.

**Generalization Gap.** Following [24], we define the generalization gap of the $t$-fold Markov transition operator induced by policy $\pi$ as the difference between its expected loss under the true distribution and its empirical loss on the training set:

$$\Delta(\mathcal{D}, S_0) = \mathbb{E}_{(S_0,\hat{S}_t)}\Big[l\Big(T^t_{\pi,\theta}(S_0), \hat{S}_t\Big)\Big] - \frac{1}{n}\sum_{i=1}^n l\big(T^t_{\pi,\theta}(s^i_0),\, \hat{s}^i_t\big) \tag{2}$$

where $\hat{S}_t$ denotes the desired goal state at time $t$, and the training dataset is given by $\mathcal{D} = \{(s^i_0, \hat{s}^i_t)\}_{i=1}^n$ containing $n$ samples drawn from the same distribution as the random variable pair $(S_0, \hat{S}_t)$. Here $l(\cdot,\cdot)$ denotes a loss function measuring the discrepancy between the predicted state and the target state. This generalization gap indicates how well a policy will perform on new states. We want to constrain the generalization gap so that the policy performs just as well on unseen situations as it does on the training data.

**Anchor States.** We define *anchor states* as $S_a \sim T^a_{\pi',\theta}(S_0)$, a set of states that the robot visits under a reduction policy $\pi'$ at timestep $a$ such that $0 < a < t$. Anchor states are a subset of states with a more concentrated distribution, making them more tractable for the base policy to execute trajectories from (see Figure 1). Formally, we say that $S_a$ constitutes an anchor if its distribution is more concentrated than that of the initial state distribution $S_0$. We formalize this condition by requiring that the trace of the covariance matrix of $S_a$ satisfy $\mathrm{tr}(\Sigma_a) < \mathrm{tr}(\Sigma_0)$. Here $\Sigma = \mathrm{Cov}(S)$ is the covariance matrix of $S_t$ and the trace of the matrix is $\mathrm{tr}(\Sigma) = \sum_{i=1}^n \sigma_i^2$, where $\sigma_i^2$ denotes the variance of the $d$-dimensional state-space along the $i^{th}$ dimension. In our motivating example, $S_a$ can represent the set of states where the box has been removed and the object is directly observable by the robot. Substituting $S_a$ in Equation (2), we can rewrite the expression for generalization gap as:

$$\Delta(\mathcal{D}, S_a) = \mathbb{E}_{(S_0,\hat{S}_t)}\Big[l\Big(T^t_{\pi,\theta}(T^a_{\pi',\theta}(S_0)), \hat{S}_t\Big)\Big] - \frac{1}{n}\sum_{i=1}^n l\big(T^t_{\pi,\theta}(T^a_{\pi',\theta}(s^i_0)),\, \hat{s}^i_t\big)\,. \tag{3}$$

### A. Using Anchor States to Reduce Generalization Error

Next we will show that constraining the state distribution by creating anchor states can lower the upper bound on the expected generalization gap, thus improving policy performance. We make the following assumptions in our analysis: i) the operator $T_{\pi',\theta}$ is linear and ii) the distribution $S$ is centered. Formally, we get the tightest upper bound on the generalization gap $\sup \mathbb{E}[\Delta(\mathcal{D}, S_a)] < \sup \mathbb{E}[\Delta(\mathcal{D}, S_0)]$.

We start by establishing a connection between the generalization error $\Delta$ and the spread of states $\mathcal{S}$. Following [25], we can write the upper bound on the expected generalization gap, with high probability over the random draw of the training sample $\mathcal{S}$, by leveraging the Rademacher complexity of the hypothesis class $\mathcal{H}$:

$$\mathbb{E}[\Delta(\mathcal{D}, S)] \leq 2\,\hat{\mathfrak{R}}_n(\mathcal{H}; S) \tag{4}$$

Here $\hat{\mathfrak{R}}_n(\mathcal{H}; S)$ denotes the Rademacher complexity with respect to the sample set $S = \{s_1, \ldots, s_n\}$, which measures how well a hypothesis class $\mathcal{H}$ can fit random noise [26]. For a linear hypothesis class $h_w(s) = \langle w, s\rangle$ with $\|w\| \leq B$, where $w$ is the parameter vector of the linear hypothesis, the Rademacher complexity can be expressed as [27]:

$$\hat{\mathfrak{R}}_n(\mathcal{H}; S) = \frac{B}{n}\sqrt{\sum_{i=1}^n \|s^i\|^2} \tag{5}$$

Building on our assumption that the distribution of $S$ is centered, we have $\mathbb{E}[S] = 0$. After combining Equations (4) and (5), we arrive at the following bound:

$$\mathbb{E}[\Delta(\mathcal{D}, S)] \leq O\left(\frac{B}{\sqrt{n}}\sqrt{\mathrm{tr}(\Sigma)}\right) \tag{6}$$

Equation (6) provides an approximate upper bound on the generalization error conditioned on a constant norm bound $B$ of the weights, sample size $n$, and the covariance of the state distribution $\Sigma$. This result suggests that the upper bound on the generalization gap *scales as a function of the variance in the state distribution*. By learning a set of anchor states that reduces the variance in the state distribution, we can reduce the upper bound on the generalization gap and thus improve policy performance on unexpected initial states.

We can also formalize this intuition by examining the state distribution and analyzing the generalization error bound from an information-theoretic perspective. Following [24], the generalization error can be upper bounded as:

$$\tilde{\mathcal{O}}\left(\sqrt{\frac{I(S_0; S_b \mid \hat{S}_t) + 1}{n}}\right) \quad \text{as } n \to \infty \tag{7}$$

where $I$ is the information gain and $S_b$ is set of intermediate states with $0 < b < t$. Consider a reduction policy $\pi'$ that leads to a linear transition operator $T_{\pi',\theta}$ which maps $S_b$ to anchor states, such that $S_a \sim T_{\pi',\theta}(S_b)$. By the data processing inequality [28], any operation on $S_t$ yields $I(S_0; S_a \mid \hat{S}_t) \leq I(S_0; S_b \mid \hat{S}_t)$. Furthermore, if the transformation reduces the effective rank, i.e., $\mathrm{rank}(\mathrm{Cov}(S_a)) < d$, then necessarily $\mathrm{rank}(T_{\pi',\theta}) < d$, making $T_{\pi',\theta}$ non-

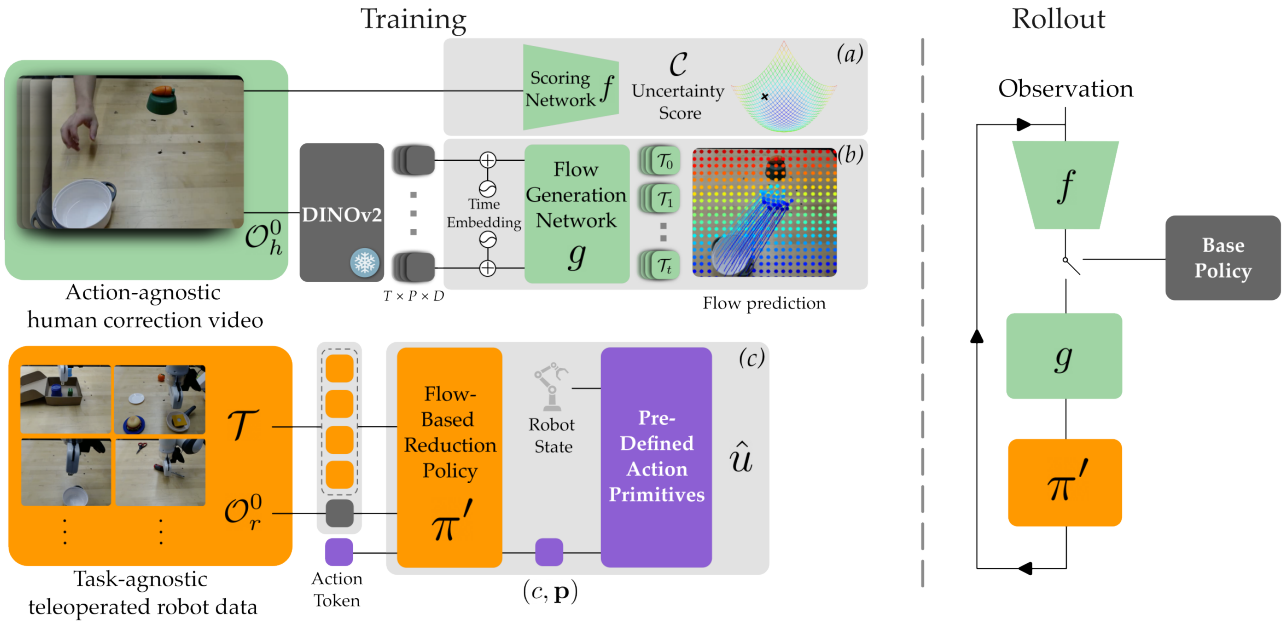


Fig. 2. *Left*: Network architecture of ReSET. The model consists of three key components: (a) a scoring network $f$, trained on human videos, which estimates the likelihood that a base policy will succeed under a given initial configuration; (b) a flow generation network $g$, which predicts flows encoding human intuition about how the scene should be restructured into anchor states; and (c) a reduction policy $\pi'$ that translates the predicted flows $\mathcal{T}$ into executable robot action primitives $\mathcal{A}$. *Right*: At rollout, the scoring network evaluates the current observation to determine whether it is ready to execute the base policy. If not, the flow generation network produces a flow plan, and the reduction policy will execute that plan. The scoring network then re-evaluates the updated scene before deciding whether to proceed with the base policy.

invertible. In this case, provided that $S_b$ is not already confined to a lower-dimensional subspace, we reach:

$$I(S_0; S_a \mid \hat{S}_t) < I(S_0; S_b \mid \hat{S}_t) \tag{8}$$

This provides an alternative analysis that demonstrates concentrating $S_b$ reduces the information it carries about the input, thereby tightening the generalization error bound.

### B. Reduction and Information-Preservation Trade-off

In Section III-A we prove that reducing the variance in the state distribution improves the generalization bounds. However, this may also simultaneously reduce the mutual information between $S_b$ and the goal states $\hat{S}_t$ such that $I(S_a; S_g) < I(S_b; S_g)$, making it harder for the policy to learn mappings from intermediate states to goal states. Returning to our motivating example, restructuring the scene by placing the target at a same position behind the box reduces the spread of states, but the target remains invisible to the robot and cannot be reliably grasped. In contrast, reconstructing the scene by removing the obstacle and exposing the target both reduces the spread of states and preserves the information necessary for the robot to execute the subsequent open-loop grasp.

The challenge therefore lies in constructing anchor states $S_a$ that (i) reduce the spread of $S_t$ to tighten the generalization error bound, while (ii) preserving the mutual information $I(S_a; S_g)$ so that $S_a$ remains sufficiently informative about the final state $S_g$. In the next section we will present our method, ReSET, where we use action agnostic human videos as a source to create anchor states that intuitively satisfy both of these requirements.

## IV. ReSET

Building on our theoretical analysis, our algorithmic approach learns a reduction policy that rearranges the environment so that it is easier for the default policy to succeed. More formally, the reduction policy takes the initial states to anchor states. By first compressing the trajectory into a narrower region of the state space, the reduction policy ensures that subsequent learning occurs on a distribution with smaller variance, which improves sample efficiency and increases the likelihood that the default policy succeeds. In this section we explain how the reduction policy is learned.

In order to reduce the state distribution into anchor states while preserving mutual information with the goal states, our method leverages *action-agnostic human videos* as a natural source for constructing anchor states. We recognize that humans instinctively restructure environments in ways that simplify task execution, while also maintaining information relevant to the final goal (e.g., humans move the box so they can see the target cup). We aim to encode how humans simplify the initial states — as captured by videos of human motion — into our reduction policy.

To learn meaningful action plans from how humans restructure the environment, we draw inspiration from recent approaches that leverage action-agnostic videos for policy learning [29], [30], and represent human videos as point flows to extract abstract actions. Building on these representations, we train a flow-based reduction policy that imitates human strategies and simplifies the environment based on initial observations. Practically, we propose a learning framework consisting of three key components:

1) A scoring network that evaluates the environment and

determines if we should keep simplifying the environment or proceed to rollout the base policy (Figure 2 (a)).
2) A flow generation network that predicts point flows which captures the intuitive strategies humans will employ to restructure the environment. (Figure 2 (b)).
3) A task-agnostic reduction policy, $\pi'$. Given the flow proposed by the flow generation network, $\pi'$ will achieve the desired outcome indicated by the flow (Figure 2 (c)).

In what follows we will discuss each component in detail.

### A. Scoring Network

The scoring network determines when to switch from the reduction policy to the base policy. In other words, it distinguishes anchor states $S_a$ from other initial states. We introduce the *scene score* $\mathcal{C}$ to quantify how likely it is for a trained policy to succeed under the initial configuration. Given a human video $O_h = \{o_h^t\}_{t=1}^T$, the score assigned to each frame $o_h^t$ is labeled with a fixed temporal prior; our assumption here is that over the course of the video the human is reconfiguring the environment into simpler state. We model this prior using a monotonically decreasing function with respect to time $t$. Frames closer to the end of the video receive lower values, indicating that the base policy is more likely to succeed by rolling out from those later frames. Specifically, we employ a second-degree polynomial in the temporal index $t$ with a negative leading coefficient:

$$\tilde{\mathcal{C}}_t = \alpha - \left(\frac{t}{\beta}\right)^2 \tag{9}$$

Here $\alpha$ and $\beta$ are tunable parameters controlling the scale and curvature of the decay. We found a parabolic decay helpful as it provides stronger discrimination at larger values of $t$, where the curvature increases with $t$. Designers can modify Equation (9) to other monotonically decreasing functions; our method is not tied to any specific scoring index.

The scoring network $f : \mathcal{O}_h \longrightarrow \mathcal{C}$ is trained solely on human motion videos: it predicts the surrogate scene score of a given observation during rollout with the loss function:

$$\mathcal{L}_{\text{score}} = \frac{1}{T}\sum_{t=1}^{T}\left(f(o^t) - \tilde{\mathcal{C}}_t\right)^2 \tag{10}$$

We then introduce a threshold on the scene score $\hat{\mathcal{C}}$ and compare it with the predicted score $f(o^t)$ to determine when to switch from the reduction policy to the base policy (see Figure 2 right). Returning to our motivating example, observations where the cup is directly visible receive a lower score $\mathcal{C}$, indicating that these states are more suitable for executing the base policy.

### B. Flow Generation Network

We now have a way to determine when we need to rearrange the environment and when we are ready to execute the default policy. But how should our reduction policy actually change the environment? Here we draw from human videos. Our approach attempts to transfer the actions performed by humans to the robot learner; i.e., we capture how humans

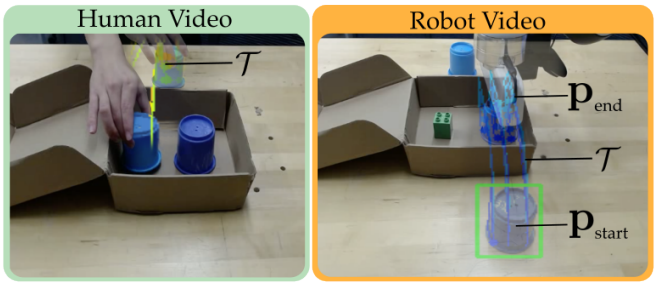


Fig. 3. Our data preprocessing pipeline. *Left*: we track the point flow of object movement $\mathcal{T}$ as a guidance to restructure the environment. *Right*: For the robot videos, we locate the object by identifying points with large displacements. Alongside the point flow, we record the parameters of the associated action primitive, i.e., the start and end positions, as well as the rotations of the robot's end effector.

prepare the environment, and enable robots to recognize and perform similar motions. The heart of this challenge is mapping *action-agnostic* human behaviors (i.e., humans manipulating objects with their own hands) into embodied actions the robot arm can perform.

We facilitate transfer across the embodiment gap by leveraging object *point flows*. We record the point flows of manipulated objects $\mathcal{T}$ in human videos using an off-the-shelf point tracking foundation model *CoTracker3* [31] (see Figure 3). We then train a flow generation network $g : \mathcal{O} \longrightarrow \mathcal{T}$ based on the videos and the extracted point flows. Given an initial scene observation $o_0$, this network predicts a point flow that represents the most likely object movements consistent with how humans might have rearranged those objects to reduce the scene score $\mathcal{C}$. To make point flows $\mathcal{T}$ easier to predict, we reduce the temporal resolution of the flow trajectories by performing linear interpolation on each point sequence along the temporal dimension. In particular, we down-sample the trajectories to a fixed horizon of 18 time steps, thereby preserving the continuous temporal structure of the original data while maintaining expressiveness.

In order to encode a richer representation of the restructuring actions, we seek to model the spatio-temporal dependencies between points. We adopt a spatiotemporal transformer architecture inspired by [32] which interleaves spatial and temporal attention layers (see Figure 2). The input image is first encoded into patch tokens using a pretrained *DINOv2* encoder [33], yielding a representation of shape $1 \times 400 \times 768$. These tokens are reconstructed by applying another patching along the spatial dimension with a $5 \times 5$ grid, resulting in $P = 16$ patches, each with token dimension $D = 5 \times 5 \times 768$. The patches are then repeated across $T$ steps, corresponding to the horizon of the point flow to be predicted. This produces a feature sequence of shape $T \times P \times D$, which is then passed into the flow generator. Notably, we observe that incorporating a learnable temporal encoding along the time dimension enhances the model's capacity to capture richer point flow representations.

Note that the flow generation network functions as a planner for restructuring the scene and is not restricted to the point-flow representation in our method. In our motivating example, its purpose is to indicate how the obstructor should be repositioned to reveal the target object behind it. Any

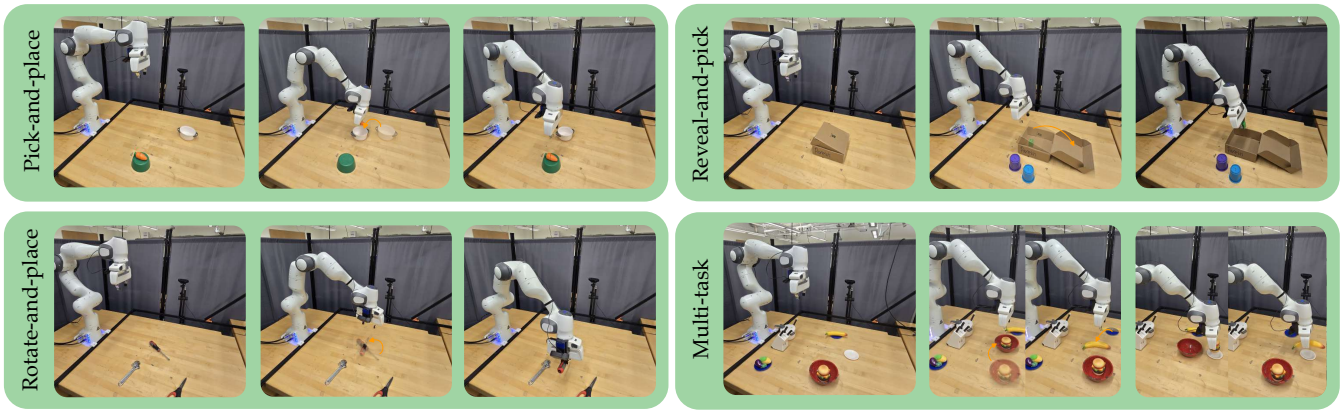


Fig. 4. We evaluated ReSET on four real-world tasks. For each task, the first image illustrates a randomly initialized scene configuration, the second shows one of the anchor states reached after our reduction policy, and the third shows the subsequent execution of the base policy.

approach capable of emulating human-like reasoning could serve the same role, such as a vision–language model.

### C. Task-Agnostic Flow-Based Reduction Policy

The flow generation network from Section IV-B provides guidance on how the scene should be manipulated to reach anchor states and make the task easier to complete. Our final step is for the robot to actually perform these actions and restructure the environment. This is our *reduction policy*: a mapping from images and predicted point flows into robot behaviors. Our reduction policy is trained using task-agnostic teleoperated robot data (e.g., videos of the robot playing with objects in the environment). In theory, the reduction policy could map to low-level robot actions. But because it remains challenging to directly predict fine-grained actions based on a point flow, our reduction policy instead outputs the parameters for a set of predefined action primitives. Across all setups, we observe that reconstruction actions can be grouped into three categories: (i) pick-and-place, (ii) push-and-pull, and (iii) rotation. Hence, we need primitives that can handle these sorts of behaviors.

We represent the action primitive space as $\mathcal{A} = \{(c, \mathbf{p})\}$, where $c \in \{c_1, c_2, c_3\}$ denotes the primitive class (pick-and-place, push-and-pull, rotate), and $\mathbf{p}$ are the continuous parameters associated with each primitive. For example, in the pick-and-place primitives $\mathbf{p}$ are the coordinates for the pick and the place. Practically, we first get the bounding box of the manipulated object around the points with the largest displacement in the scene. The action primitives are then parsed by a temporal window between the beginning and end of the detected motion (see Figure 3).

The reduction policy takes as input a point flow $\mathcal{T}$ and a robot initial observation $O_r^0$ and predicts an action primitive $\mathcal{A}$, $\pi' : \mathcal{T} \times \mathcal{O}_r^0 \longrightarrow \mathcal{A}$. We use a transformer architecture as backbone. A learnable action token is first initialized and concatenated with the input tokens, consisting of the point flow and robot observation features. This joint token sequence is then processed by the transformer to produce the predicted action primitive $\mathcal{A}$. To train $\pi'$ we use a composite loss that combines classification loss for the primitive type and regression loss for its parameters. Given model predictions $(\hat{c}, \hat{\mathbf{p}})$, the total loss is defined as:

$$\mathcal{L}_{\pi'} = \lambda_{\text{cls}}\, \mathcal{L}_{\text{cls}}(c, \hat{c}) \;+\; \lambda_{\text{reg}}\, \mathcal{L}_{\text{reg}}(\mathbf{p}, \hat{\mathbf{p}}), \tag{11}$$

where $(c, \mathbf{p})$ denote the ground-truth primitive type and parameters. The first term, $\mathcal{L}_{\text{cls}}(c, \hat{c})$ is the cross-entropy loss over primitive classes, and $\mathcal{L}_{\text{reg}}(\mathbf{p}, \hat{\mathbf{p}})$ is the mean-squared error (MSE) loss for parameters. The coefficients $\lambda_{\text{cls}}, \lambda_{\text{reg}} > 0$ balance the two objectives.

## V. Experiments

We experimentally compare ReSET to state-of-the-art alternatives. Within our experiments we evaluate tabletop object manipulation tasks that require composite, long-horizon reasoning by a robot arm. We focus on the interplay between training data and task performance, aiming to answer the following question: does introducing a reduction policy that restructures the environment reduce the effort of data collection — both for training the reduction policy itself and for the base model? For demonstrations of our experiments, please see the accompanying videos.

**Baselines.** We compare ReSET with multiple visual imitation learning baselines on a variety of task settings.

*1) ReSET Naive:* An ablation of ReSET. Rather than learning a flow-based reduction policy, ReSET Naive directly maps from the initial observation to corrective actions. The policy is trained on an augmented dataset that explicitly aligns initial observations from human demonstrations and actions from robot play data based on the most similar flows, measured via the $\|\circ\|_2$ norm of the flow difference. Comparing to this baseline will show that our flow-based reduction policy is a necessary component for learning a more diverse range of reconstructive actions.

*2) Diffusion Policy [8]:* Uses same rollout policy as our method, but does not call ReSET to prepare the environment.

*3) Dynamics-DP [22]:* Generates augmented data on top of expert demonstrations to train a more robust diffusion policy. We implement the method introduced in the original paper and adapted an action decoder proposed by [34], extending it into a visual dynamics model for Model Predictive Path Integral (MPPI) [35] Planning.

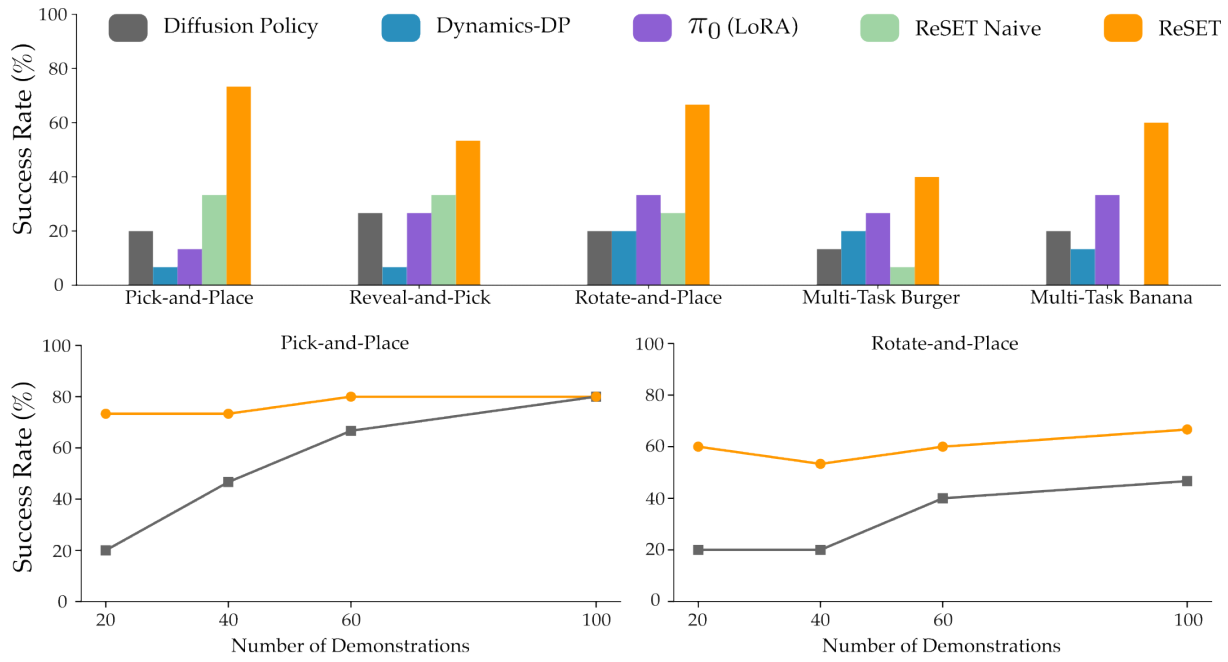


Fig. 5. Results from our real-world experiments. Each task was evaluated on 15 test scenarios, including out-of-distribution states that were not covered in the expert robot demonstrations. *Top*: Success rate over four tasks. *Bottom*: Performance of ReSET and Diffusion Policy with 20, 40, 60, and 100 demonstrations. We observe that across all tasks ReSET achieves a higher success rate and outperforms the baselines. We also observe that — across different amounts of data available for training — ReSET achieves a higher success rate as compared to Diffusion Policy.

*4) $\pi_0$ [9]:* A VLA baseline. The policy leverage large-scale multimodal training data to align visual perception with language understanding. We take the flow matching version of the $\pi_0$ policy, and perform low-memory finetuning (LoRA) [36] with task specific data for $30,000$ steps.

**Setup.** Our experiments are conducted using a Franka Emika robot arm on a tabletop manipulation setup. We employ a GELLO controller [37] to teleoperate the robot, collecting both the play dataset used for training the reduction policy and the task-oriented expert dataset used for training the base policy. The experimental setup incorporates two cameras. The static side-view camera is used as a primary camera for all datasets, and the gripper-mounted camera is used as a secondary camera in the demonstration dataset.

**Tasks.** We evaluated ReSET against the baselines on four real-world tasks (see Figure 4).

1) *Pick-and-place*: Grasp the carrot from its initial location and deposit it into the bowl.
2) *Reveal-and-pick*: Actively remove obstructing objects to uncover the hidden block, then grasp that block.
3) *Rotate-and-place*: Pick up a screwdriver that is originally at an angle, and place it parallel to the other tools.
4) *Multi-task*: Serve either a burger or a banana based on user input. The two foods are on separate plates.

**Results.** Figure 5 compares the performance of each method across the four tasks. All methods are trained on 20 expert robot demonstrations, with the exception of Dynamics-DP, which also uses an augmented dataset. We collected the same amount of augmented data as human data for fair comparison. ReSET and its naive point-track matching variant incorporate 20 action-agnostic human videos and 20 minutes of robot play data for each task. We examine the data efficiency of our approach in the following sections. Also note that ReSET combines robot play data for all the tasks and learns a single reduction policy. Each task was evaluated on 15 test scenarios, approximately $80\%$ of which featured distributional shifts relative to the training set.

### A. How well does ReSET recover from unexpected states?

We found that ReSET learns the restructuring strategies featured in the human videos and executes the reduction policy from out-of-distribution states. For example, in the pick-and-place task, the robot pulls the white bowl towards the carrot so that it is easier for the base policy to accurately drop the carrot into the bowl. In the rotate-and-place task, the robot first rotates the screwdriver into an easier-to-grasp orientation before picking it up and placing it among other tools (see Figure 4 and supplemental videos).

In long horizon tasks like reveal-and-pick, ReSET is able to execute a sequence of reconstructions of the scene until an anchor state is reached. If the robot observes the block is still covered under a cup after opening the box, it will proceed to pick up the cup and reveal the block. In a multi-task setup, ReSET generates different restructuring actions according to different instructions ("Serve burger" or "Serve orange").

### B. How does a flow-based reduction policy help?

Comparing ReSET to ReSET Naive underscores the importance of incorporating flow into the reduction model. ReSET adapts more effectively to variations in the initial observation, generating richer restructuring plans than the naive ablation. Indeed, point track matching consistently fails to handle slight positional shifts of objects. This behavior is expected: during training for ReSET Naive the same robot

action is often aligned with multiple distinct initial observations from the human dataset, thereby reducing the effective diversity of action variants available for learning. Our flow based reduction policy captures more diverse actions from robot play data, enabling more flexible and adaptive behavior.

### C. Is ReSET more data efficient that the alternatives?

To demonstrate that our method remains data efficient despite its reliance on both human and robot play data, we trained diffusion policies with an increased number of expert robot demonstrations and compared their performance against ReSET (Figure 5 bottom). We found that diffusion policies requires at least 70 expert demonstrations for each task to reach comparable performance. Based on our experience, collecting this amount of data takes at least one hour for an expert demonstrator. In contrast, ReSET only requires about 10 minutes of task-specific human corrections plus 20 minutes of task-invariant robot play data. This highlights that with ReSET the robot can recover from out-of-distribution scenarios with substantially less time spent collecting demonstrations.

## VI. Conclusion

In this paper we introduced ReSET, an algorithmic framework that restructures environments for easier task execution. ReSET learns from two sources of training data: action-agnostic videos of humans rearranging the scene, and task-agnostic videos of teleoperated robot play. From the human videos we extract i) a score that determines whether the robot should roll-out its base policy or simplify the environment, and ii) a flow prediction that captures how humans would manipulate objects. Using the robot play videos, we then convert these into iii) a policy to map the point flow into robot actions that interact with the environment and lower the difficulty score. Overall, our learned approach rearranges the environment to bring out-of-distribution initial states back into a smaller, known distribution (i.e., anchor states). Our theoretical analysis suggests that moving to anchor states before executing the baseline policy improves generalization in a data-efficient manner. This analysis is supported by our experiments, where we show that ReSET leads to higher performance across multiple tasks and data amounts.

## References


[1] R. R. Sanchez, H. Nemlekar, S. Sagheb, C. M. Nunez, and D. P. Losey, "RECON: Reducing causal confusion with human-placed markers," in *IEEE/RSJ IROS*, 2024.

[2] Y. Dai, R. R. Sanchez, R. Jeronimus, S. Sagheb, C. M. Nunez, H. Nemlekar, and D. P. Losey, "CIVIL: Causal and intuitive visual imitation learning," *arXiv preprint arXiv:2504.17959*, 2025.

[3] M. Dalal, M. Liu, W. Talbott, C. Chen, D. Pathak, J. Zhang, and R. Salakhutdinov, "Local policies enable zero-shot long-horizon manipulation," in *IEEE ICRA*, 2025, pp. 13 875–13 882.

[4] H. Bharadhwaj, R. Mottaghi, A. Gupta, and S. Tulsiani, "Track2Act: Predicting point tracks from internet videos enables generalizable robot manipulation," in *ECCV*, 2024.

[5] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne, "Imitation learning: A survey of learning methods," *ACM Computing Surveys*, 2017.

[6] S. Haldar, Z. Peng, and L. Pinto, "Baku: An efficient transformer for multi-task policy learning," *NeurIPS*, pp. 141 208–141 239, 2024.

[7] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid, *et al.*, "Rt-2: Vision-language-action models transfer web knowledge to robotic control," in *Conference on Robot Learning*, 2023, pp. 2165–2183.

[8] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," *IJRR*, 2024.

[9] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, *et al.*, "*pi*_0: A vision-language-action flow model for general robot control," *arXiv preprint arXiv:2410.24164*, 2024.

[10] M. Kim, K. Pertsch, *et al.*, "OpenVLA: An open-source vision-language-action model," *arXiv preprint arXiv:2406.09246*, 2024.

[11] P. Gupta, H. Admoni, and A. Bajcsy, "Adapting by analogy: OOD Generalization of visuomotor policies via functional correspondence," *arXiv preprint arXiv:2506.12678*, 2025.

[12] J. Mao, J. Guan, *et al.*, "OmniD: Generalizable robot manipulation policy via image-based bev representation," *arXiv preprint arXiv:2508.11898*, 2025.

[13] F. Lin, Y. Hu, P. Sheng, C. Wen, J. You, and Y. Gao, "Data scaling laws in imitation learning for robotic manipulation," *arXiv preprint arXiv:2410.18647*, 2024.

[14] A. Mandlekar, J. Booher, M. Spero, A. Tung, A. Gupta, Y. Zhu, A. Garg, S. Savarese, and L. Fei-Fei, "Scaling robot supervision to hundreds of hours with roboturk: Robotic manipulation dataset through human reasoning and dexterity," in *IEEE/RSJ IROS*, 2019.

[15] F. Ebert, Y. Yang, K. Schmeckpeper, B. Bucher, G. Georgakis, K. Daniilidis, C. Finn, and S. Levine, "Bridge data: Boosting generalization of robotic skills with cross-domain datasets," *arXiv preprint arXiv:2109.13396*, 2021.

[16] A. Iyer, Z. Peng, Y. Dai, I. Guzey, S. Haldar, S. Chintala, and L. Pinto, "Open teach: A versatile teleoperation system for robotic manipulation," *arXiv preprint arXiv:2403.07870*, 2024.

[17] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, "Learning fine-grained bimanual manipulation with low-cost hardware," *arXiv preprint arXiv:2304.13705*, 2023.

[18] Z. Chen, Z. Mandi, H. Bharadhwaj, M. Sharma, S. Song, A. Gupta, and V. Kumar, "Semantically controllable augmentations for generalizable robot learning," *IJRR*, 2024.

[19] Z. Mandi, H. Bharadhwaj, V. Moens, S. Song, A. Rajeswaran, and V. Kumar, "CACTI: A framework for scalable multi-task multi-scene visual imitation learning," *arXiv preprint arXiv:2212.05711*, 2022.

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

[21] G. J. Gao, T. Li, and N. Figueroa, "Out-of-distribution recovery with object-centric keypoint inverse policy for visuomotor imitation learning," *arXiv preprint arXiv:2411.03294*, 2024.

[22] R. Wu, H. Chen, M. Zhang, H. Lu, Y. Li, and Y. Li, "Neural dynamics augmented diffusion policy," in *IEEE ICRA*, 2025.

[23] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artificial Intelligence*, vol. 101, no. 1-2, pp. 99–134, 1998.

[24] K. Kawaguchi, Z. Deng, X. Ji, and J. Huang, "How does information bottleneck help deep learning?" in *ICML*, 2023, pp. 16 049–16 096.

[25] P. L. Bartlett and S. Mendelson, "Rademacher and gaussian complexities: Risk bounds and structural results," *Journal of Machine Learning Research*, 2002.

[26] V. Koltchinskii and D. Panchenko, "Empirical margin distributions and bounding the generalization error of combined classifiers," *The Annals of Statistics*, 2002.

[27] C. Lampert, "Useful properties of the (empirical) Rademacher complexity."

[28] N. J. Beaudry and R. Renner, "An intuitive proof of the data processing inequality," *arXiv preprint arXiv:1107.0740*, 2011.

[29] M. Xu, Z. Xu, Y. Xu, C. Chi, G. Wetzstein, M. Veloso, and S. Song, "Flow as the cross-domain manipulation interface," *arXiv preprint arXiv:2407.15208*, 2024.

[30] C. Wang, L. Fan, J. Sun, R. Zhang, L. Fei-Fei, D. Xu, Y. Zhu, and A. Anandkumar, "MimicPlay: Long-horizon imitation learning by watching human play," *arXiv preprint arXiv:2302.12422*, 2023.

[31] N. Karaev, I. Makarov, J. Wang, N. Neverova, A. Vedaldi, and C. Rupprecht, "CoTracker3: Simpler and better point tracking by pseudo-labelling real videos," *arXiv preprint arXiv:2410.11831*, 2024.

[32] J. Bruce, M. D. Dennis, *et al.*, "Genie: Generative interactive environments," in *ICML*, 2024.

[33] M. Oquab *et al.*, "DINOv2: Learning robust visual features without supervision," 2023.

[34] S. Gao, S. Zhou, Y. Du, J. Zhang, and C. Gan, "AdaWorld: Learning adaptable world models with latent actions," in *ICML*, 2025.

[35] G. Williams, A. Aldrich, and E. A. Theodorou, "Model predictive path integral control: From theory to parallel computation," *Journal of Guidance, Control, and Dynamics*, 2017.

[36] E. J. Hu *et al.*, "LoRA: Low-rank adaptation of large language models," *ICLR*, 2022.

[37] P. Wu, Y. Shentu, Z. Yi, X. Lin, and P. Abbeel, "Gello: A general, low-cost, and intuitive teleoperation framework for robot manipulators," in *IEEE/RSJ IROS*, 2024, pp. 12 156–12 163.