ADP3: Affordance-Guided Generalizable Visuomotor Policies through 3D Action Diffusion

by

Karthik Biju Nair

Supervisors:	Prof. Dr. Jens Kober[1]
	Dr. Milad Malekzadeh[2]
	Dr. Jeyhoon Maskani[2]
Thesis Committee:	Prof. Dr. Jens Kober
	Dr. Jeyhoon Maskani
	Dr. J. Micah Prendergast
	Dr. Chirag Raman
Project Duration:	Nov 2024 - Aug 2025
Programme:	MSc. Robotics
Department & Faculty:	Cognitive Robotics, Mechanical Engineering
Defense Date:	August 8, 2025, 10:00 AM

[1] Cognitive Robotics, TU Delft [2] Neura Robotics GmbH

TU Delft

ADP3: Affordance-Guided Generalizable Visuomotor Policies through 3D Action Diffusion

Karthik Biju Nair*†
*TU Delft
†Neura Robotics GmbH

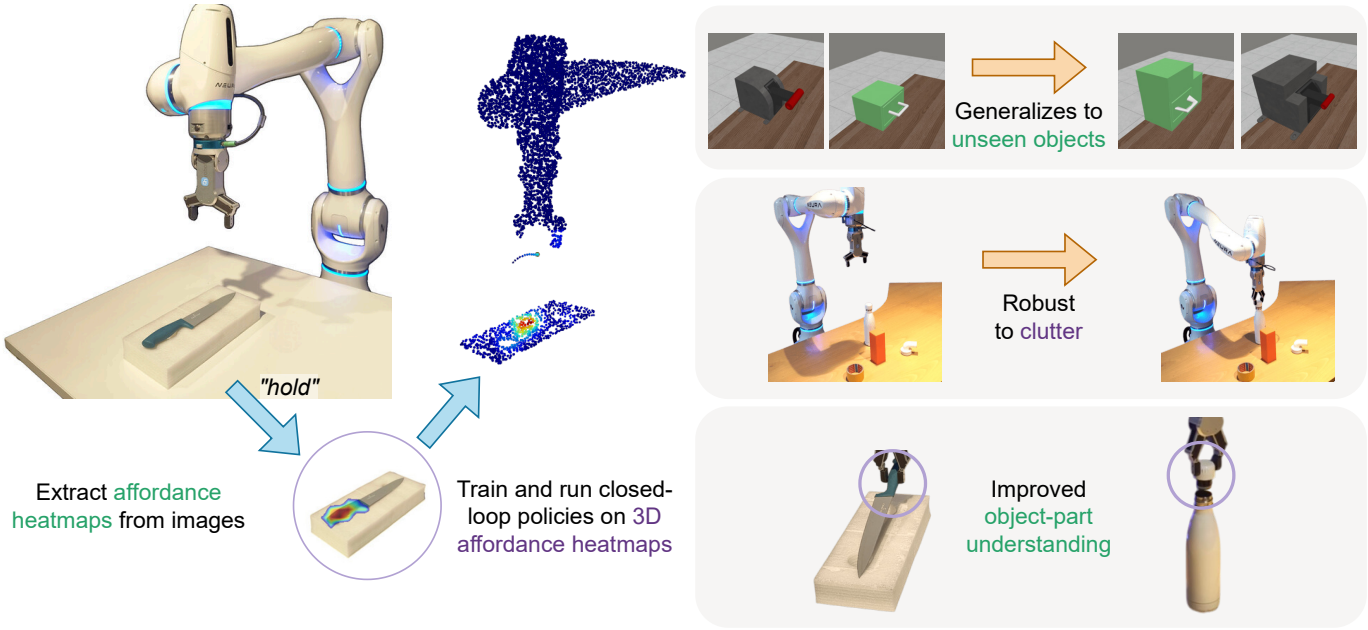


Fig. 1: We introduce Affordance-Guided 3D Diffusion Policy (ADP3), a generalizable and robust policy that uses 3D interaction heatmaps as the visual representation. By conditioning action generation on these heatmaps, ADP3 attains strong generalization abilities, robustness to clutter, and improved object-part understanding.

***Abstract*—Recent progress in visual imitation learning has shown that diffusion models are a powerful tool for training robots to perform complex manipulation tasks. While 3D Diffusion Policy uses a point cloud representation to improve spatial reasoning and sample efficiency, it still struggles to generalize across novel objects and environments due to spurious correlations learned from irrelevant visual features. In this work, a novel approach, Affordance-guided 3D Diffusion Policy (ADP3) is introduced, which integrates task-relevant affordance cues into the policy's point cloud input. By conditioning the policy on 3D affordance heatmaps instead of raw point clouds, the policy is biased to attend to task-relevant object regions. Using affordance heatmaps reduced the success rate drop to just 3% on unseen objects in 4 Meta-World tasks, compared to a 35% drop when using raw point clouds. ADP3 also demonstrates impressive performance in our real-world experiments, showing resilience to cluttered scenes and novel object-orientations.**

I. Introduction

In recent years, visual imitation learning, specifically behavioral cloning, has resurged as a powerful technique to teach robots complex manipulation tasks. A major contributor to this rise is the use of diffusion models for policy representation [1]. These policies offer stable training dynamics and can model multimodal action distributions, making them perfect for real-world tasks that have multiple ways to achieve the same goal.

It is desirable to teach robots manipulation skills that can be transferred to different objects and environments. However, diffusion-based policies that condition the action generation on RGB images are sensitive to changes in the object's appearance and the background. For instance, Diffusion Policy [1] trained to pour water from only one type of bottle is found to fail to transfer this knowledge to a different bottle [2]. It is cumbersome to train policies on a wide range of objects of the same category, requiring thousands of demonstrations for a single task. To overcome this limitation, policies such as DP3 [3] use a point cloud representation that allows for efficient learning, improved scope for spatial reasoning, and robustness to slight shape variations.

Despite the use of a point cloud representation, significant changes in the object's appearance still adversely affect the policy performance. The main reason for this is the joint training of the vision encoder and the policy head, leading to the policy learning spurious correspondences. For instance, a policy trained to open a drawer might focus on geometric features such as the drawer's frame or edge instead of focusing on the handle. This hampers the ability of the policy to generalize to novel objects, since variations in the irrelevant parts of the observation can cause the policy to fail.

Humans, on the other hand, have an excellent understanding of how to manipulate objects. We know that a knife needs to be held at the handle, which way to rotate the cap of a bottle to open it, and where to hold a bag to lift it. The term "affordance" is used to define such task-specific interaction possibilities with an object [4]. Most importantly, we can transfer object interaction knowledge across instances quite easily, even if the instances look dissimilar, by focusing on task-relevant information. Inspired by this idea, we introduce a simple yet effective strategy to address the issue of spurious correspondences by integrating affordances into the point cloud representation. Current foundation models have abundant open-world understanding, which we use to extract affordance heatmaps from images. These heatmaps implicitly guide the policy, indicating the task-relevant regions in an object, leading to the policy learning to focus on important visual information.

Specifically, we leverage the object interaction knowledge of OOAL [5] to obtain 2D affordance heatmaps that are lifted to 3D as an additional point cloud channel. We condition the Affordance-guided 3D Diffusion Policy (**ADP3**) on latent scene and affordance representations obtained from a point cloud using a two-stream encoder. This enables the policy to focus on task-specific regions in the observation while retaining visual context from the scene.

Even though we leverage large-scale pretraining to extract affordances through the use of a vision transformer, our method differs from Vision-Language-Action (VLA) models such as OpenVLA [6], Pi-Zero [7], and Gr00t N1 [8]. We focus on *injecting* semantic information into 3D observations rather than semantic *grounding* through large-scale vision-language-action pretraining.

Our contributions and results can be summarized as follows:

1) Affordance guidance as a mechanism to guide the diffusion policy's focus on task-specific object parts.
2) Simulated experiments that reveal the robustness of the policy to distractors and in generalizing to novel objects that look very different from the training object.
3) Real world experiments that showcase the effectiveness of ADP3 in realistic scenarios and in cluttered scenes.

II. Related Work

A. Visual Imitation Learning

Visual imitation learning (VIL) forgoes extensive hand engineering and reward definition, allowing for easily teaching manipulation skills through demonstrations. While 2D image-based policies are widely used [9, 10, 11, 1], 3D representations are becoming increasingly prevalent since they enable sample-efficient policies that can generalize spatially [12, 13, 14, 15, 3].

Another common technique to improve generalization is to use *pretrained visual representations*: features extracted from raw observations to make policy learning easier [16, 17, 18]. These representations are extracted by either training on *in-domain* data using latent dynamics models [19, 20, 21] or *out-of domain* internet-scale images [22, 23, 24, 25] and videos [26, 27, 28]. While such representations succeed in capturing semantic features, they also include irrelevant features and are task agnostic. In contrast, we extract 3D affordance heatmaps that act as a low-dimensional, task-specific semantic map, helping the policy to ignore irrelevant features.

B. Affordance Learning

Affordance learning refers to the task of segmenting regions in a scene pertaining to an object interaction, such as the handle of a kettle for holding or the cap of a bottle for opening. Early work in affordance learning relied on simulated interactions using predefined motion primitives [29, 30, 31] or used dense annotations on large datasets [32]. However, the former is slow due to its reliance on simulated environments, and the latter requires thousands of annotations even when restricted to a closed set of affordance labels. Recent approaches leverage semantic knowledge of foundation models to reduce manual affordance annotations. OOAL [5], trained by aligning affordance text embeddings with visual features, can predict affordance scores by learning from just one annotated image per category.

Transferring object-interaction knowledge to robotic manipulation to enable generalization has been an active research topic. Some approaches achieve this by just predicting affordance regions [33, 34], while others additionally predict post-contact trajectories [35, 36, 37, 38, 39]. In contrast, we inject affordance information into the observation space to provide additional guidance, instead of regressing trajectory waypoints directly.

C. Generalization in Diffusion Policies

Diffusion policy [1] has shown impressive performance in a variety of day-to-day tasks through a simple behavioral cloning formulation. Nevertheless, it is subject to the distributional shift problem and is sensitive to factors such as the camera viewpoint, object geometry, and spatial layout. A 3D observation representation [15, 3, 41] improves generalization by capturing spatial structure more directly, but the policy still struggles when the object is significantly different in appearance. Some approaches rely on equivariance [42, 43], motivated by the idea that the transformation in the observation space results in the same transformation in the action space. However, these methods heavily rely on scene symmetry, making them sensitive to distractors. Others use keypoints [44] or object-centric representations [45] to simplify perception, but they remain limited in real-world applicability. GenDP [46] utilizes 3D descriptor fields for policy learning, that are representations that remain consistent across objects of the same category. However, the approach requires a reference image per category and does not incorporate task-specific guidance. Lan-o3dp [47] is the closest to our approach, using object segmentation masks to discard background information. Unlike Lan-o3dp, rather than completely discarding visual context, we highlight task-specific regions while retaining the full scene, allowing the policy to attend selectively to the information it requires.

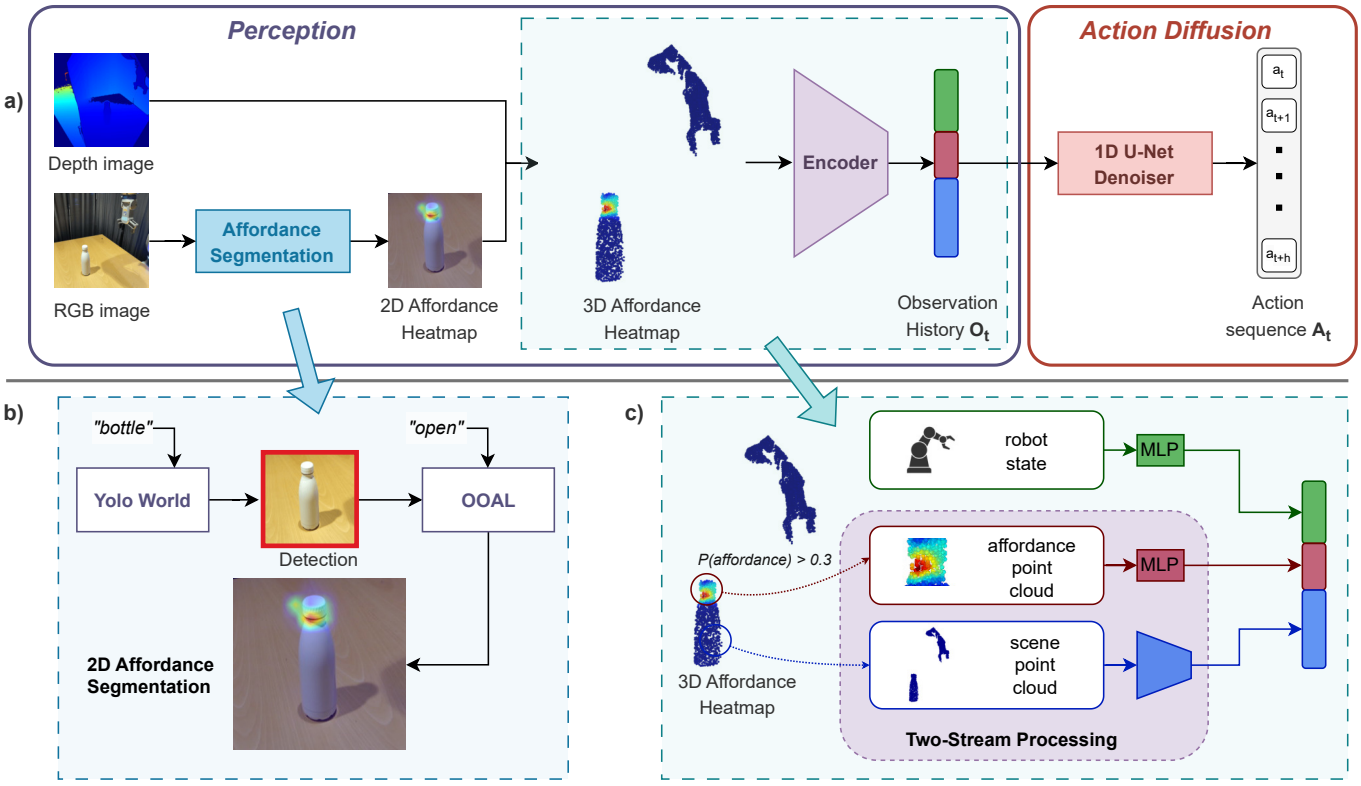


Fig. 2: **Method overview of Affordance-Guided Diffusion Policy (ADP3). (a) The ADP3 Pipeline.** We obtain affordance heatmaps from RGB images captured by a single-view depth camera and lift them to 3D using depth. The result is passed to a two-stream encoder to produce an observation encoding o_t. The diffusion head conditions on an observation history O_t, observation encodings stacked across timesteps, to generate an action sequence A_t (the future robot states); **(b) Affordance Segmentation Module.** We use YOLO-World [40] to extract a crop of the target object and assign pixel-level affordance scores using OOAL; **(c) Two-Stream Processing.** The final observation is a 1D vector concatenating robot state features and observation features. The latter is obtained by processing scene features (96 channels) and affordance features (32 channels) separately and stacking them, capturing both global context and task-specific information for the diffusion policy.

III. Method

A. Problem Formulation

The goal is to learn a single-task policy $\pi_\theta(a_t \mid s_t)$ from N expert demonstrations $\mathcal{D} = \{\tau_i\}_{i=1}^N$, where s_t is the state and a_t represents the action corresponding to that state. Each trajectory is denoted by $\tau_i = \{(s_0, a_0), ..., (s_T, a_T)\}$.

An overview of our method is illustrated in Figure 2. Our formulation is targeted towards short-horizon tasks, where we assume only one affordance type, which is preselected. Multi-step tasks require sequential and complex reasoning over multiple affordances which is outside the scope of our work.

The state $s_t \in \mathcal{S}$ is a combination of robot states and observations. The robot state consists of the robot pose and gripper state, while the observation includes a 3D affordance heatmap $(N, 4)$. The spatial coordinates of the points make up the first three dimensions of the heatmap, while the affordance score $\phi_a \in [0, 1]$ is the last dimension. We obtain 3D heatmaps from 2D affordance heatmaps using depth information.

B. Affordance-Guided Diffusion Policy

We aim to model the conditional distribution $P_\theta(A_t|O_t, Q_t)$ of a short-horizon action sequence $A_t = a_{t:t+n}$ using a denoising diffusion process, similar to Diffusion Policy [1]. Here, $O_t = o_{t-h:t}$ denotes the sequence of the h most recent observations, where each o_t is a 3D affordance heatmap; $Q_t = q_{t-h:t}$ denotes the corresponding robot states.

At each denoising step k, the reverse process is defined as:

$$A_{k-1}^t = \alpha_k \left(A_k^t - \gamma_k \epsilon_\theta(A_k^t, O_t, Q_t, k)\right) + \sigma_k \mathcal{N}(0, I),$$

where ϵ_θ is the denoising network conditioned on both the visual observations O_t and robot states Q_t, and α_k, γ_k, and σ_k are noise schedule parameters.

C. Training Objective

During training, we randomly sample a timestep $k \in \{1, \ldots, K\}$. For that timestep, we generate a noisy action sequence A_k^t from a clean action sequence A_0^t by perturbing the clean action sequence A_0^t with Gaussian noise. The clean actions are obtained from the training data. Specifically, we sample $\epsilon_k \sim \mathcal{N}(0, I)$ and obtain the noised sequence as:

$$A_k^t = \bar{\alpha}_k A_0^t + \bar{\beta}_k \epsilon_k,$$

where $\bar{\alpha}_k$ and $\bar{\beta}_k$ are scalar coefficients from the diffusion noise schedule.

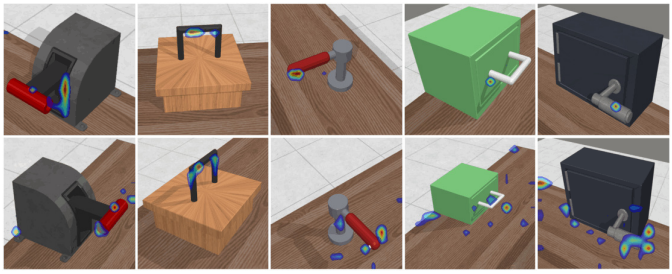


Fig. 3: Noisy heatmaps predicted by OOAL for the affordance *hold* in the Meta-World benchmark.

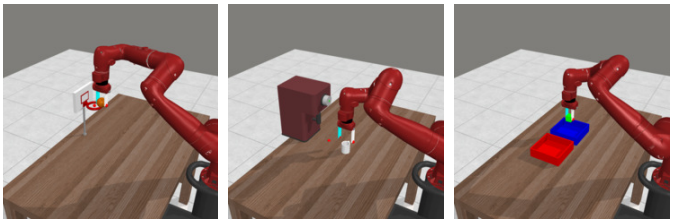


Fig. 4: To assess encoder performance, we select three tasks from the Meta-World benchmark.

The denoising network ϵ_θ is then trained to minimize the difference between the predicted and actual noise:

$$\mathcal{L}_{\text{MSE}} = \text{MSE}\left(\epsilon_k,\ \epsilon_\theta\left(A_k^t,\ k,\ O_t,\ Q_t\right)\right).$$

D. Visual Encoder

The visual encoder plays an integral role in extracting a compact visual representation from the observations for the downstream task: policy learning. It captures essential information from the point cloud as a one dimensional latent vector. We train the visual encoder along with the policy head on a behavioral cloning loss. The encoder is trained from scratch since pretraining only leads to a marginal improvement in performance.

We use a two-stream architecture (see Figure 2) in order to integrate 2D affordance heatmaps with the point cloud representation. We first transform pixel-level affordance heatmaps to the corresponding points in 3D using depth information. This is then processed by our adapted point cloud encoder, which we call the ADP3 encoder.

The ADP3 encoder consists of two parallel streams: 1) a scene encoder that processes the entire point cloud to extract global features, and 2) an affordance encoder that focuses on task-relevant regions extracted using affordance scores. The scene encoder is a pyramidal encoder, introduced in [41], that captures hierarchical geometric patterns across different scales. Interestingly, we observed that using a simple MLP as the affordance encoder led to the best success rates in comparison to traditional point cloud encoders. The affordance and scene features are concatenated and serves as the compact observation representation.

Nevertheless, simply adding affordance features to the observation is often not sufficient to guarantee that: 1) the policy avoids overfitting to irrelevant scene details and, 2) it attends to relevant local geometry. To address this, we randomly zero out scene features, forcing the policy to learn from the specific local geometry. This way, once trained, the policy can transfer the learned skill to different objects that have similar local geometry.

E. Implementation Details

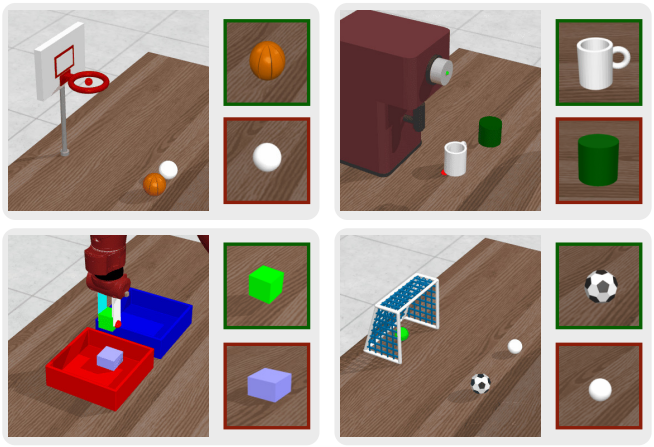
We adopt the conditional U-Net from Diffusion Policy [1] as the backbone denoising network. For diffusion sampling, we use a DDIM noise scheduler for faster sampling with fewer inference steps compared to DDPM. We use dense point clouds as observations, which are first cropped to the relevant workspace area and then downsampled to 4096 points, as in [3]. We use uniform sampling instead of Farthest Point Sampling (FPS) in all our experiments for efficient point cloud processing.

IV. Simulation Experiments

In this section, we describe the experiments conducted in order to assess the effectiveness and the generalization capabilities of affordance-guided policies. Through carefully designed experiments, we compare our method with the baseline, 3D Diffusion Policy (DP3) [3]. Our experiments aim to investigate the fine-grained geometric understanding of point cloud-based policies and their ability to generalize to different instances from the same category.

We start by first validating our method in simulation through experiments that validate the usefulness of affordance input. For this purpose, we use the Meta-World benchmark [48] and select tasks with varying difficulty. For all our experiments in Meta-World, we use 10 demonstrations generated using the hard-coded policies provided within the benchmark. We use ground truth affordances in simulated environments instead of extracting affordance heatmaps using OOAL. Since OOAL was trained on a small set of real images, it cannot consistently extract high quality affordance heatmaps from objects in simulation as illustrated in Figure 3. We obtain the 3D affordance heatmaps by assigning high affordance scores to all points within a distance of 0.03 m from the grasp location.

Standard point cloud encoders such as PointNet [49] and PointNet++ [50] produce latent representations that are not informative enough for policy learning. This was also observed in [3] when training the encoders from scratch. Hence, the pyramidal encoder from [41] was selected as the baseline, an improved version of the MLP-based encoder from [3]. In all simulated experiments, each policy is trained for 400 epochs, with 20 evaluation episodes in the environment run every 50 epochs. Our primary performance metric is the task success rate. We train policies for each task with seeds 0, 1, and 2 to reliably assess their performance.

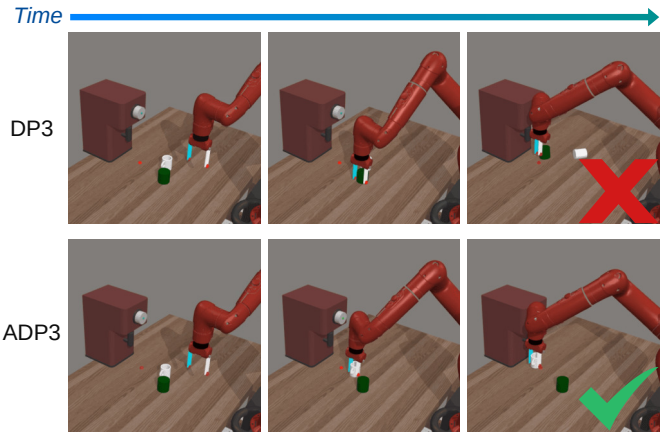


Task	Object (name, size (mm))	Distractor Object (type, size (mm))
Basketball	basketball, $r = 30$	sphere, $r = 25$
Coffee Push	mug, $r = 28$, $h = 33$	cylinder, $r = 30$, $h = 30$
Bin Picking	cube, $x = y = z = 20$	cuboid, $x = y = 25$, $z = 15$
Soccer	soccer ball, $r = 26$	sphere, $r = 20$

Fig. 5: **Robustness to distractor objects:** the figure shows the experimental setup with distractor objects for four tasks (starting from the top-left): *basketball*, *coffee push*, *bin picking* and *soccer*. The table lists the main and distractor objects used for each task.

Encoder	Basketball	Coffee Push	Bin Picking
DP3	99	20	17
ADP3	**100**	**73**	**78**

TABLE I: Success Rates (%) of different encoders across tasks.



Encoder	Coffee Push	Bin Picking	Basketball	Soccer
ADP3	**90 ± 2**	**95 ± 3**	**80 ± 0**	**47 ± 2**
DP3	53 ± 7	30 ± 6	36 ± 3	32 ± 10

Fig. 6: **Results of distractor robustness experiments:** Baseline DP3 fails to distinguish the distractor cylinder from the mug in the *coffee push* task, while affordance guidance enables the policy to do so. The table reports success rates (in percentage) across four tasks, averaged over three seeds.

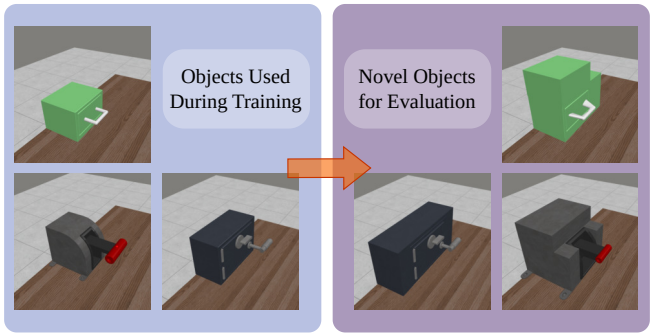
A. *Effects of Affordance Conditioning*

1) Improvement in Task Performance: We evaluate the effectiveness of an affordance input on three manipulation tasks: *basketball*, *bin picking*, and *coffee push*, depicted in Figure 4. It is observed that affordances significantly improve the task performance throughout all tasks. Affordance signals are found to enhance the latent representation, which becomes especially evident on tasks that the baseline encoder struggles with. The ADP3 encoder generates an encoding using the scene encoder, that is architecturally equivalent to the baseline. In ADP3, this encoding is later concatenated with the affordance encoding to form the final observation encoding. The observation encoding is a vector of size 128 in both cases for fair comparison. By including affordance information in the observation, the mean success rates increased from 20% to 73% for the *coffee push* task and 17% to 78% for the *bin picking* task. This highlights the need to extract a task-relevant representation for policy learning.

2) Robustness to Distractor Objects: For imitation learning to be effective in practical real-world applications, the policy needs to be able to operate in unstructured environments. It should be able to manipulate the target object even with several other objects present in the scene. DP3's ability to generalize to different object instances (cube, charger, cylinder, etc.) is attributed to the downsampling of the point cloud to a few representative points [3]. However, this also limits the policy's ability to distinguish similar objects. This experiment involves introducing distractor objects to the scene during training in order to test the policy attention to fine geometric details. Beyond increasing the task difficulty, distractor objects aid in evaluating the policy's ability to identify the target object.

We evaluate our method on 4 diverse Meta-World environments, modified to spawn both the target object and the distractor object (see Figure 5). The distractor objects are primitive shapes resembling the target's shape but of different dimensions. We increase the object spawn region to accommodate both objects without negatively affecting the expert policies. For each task, the distractor object is spawned during training as well as evaluation. This setup requires the encoder to capture local features for the policy to distinguish the target object from the distractor.

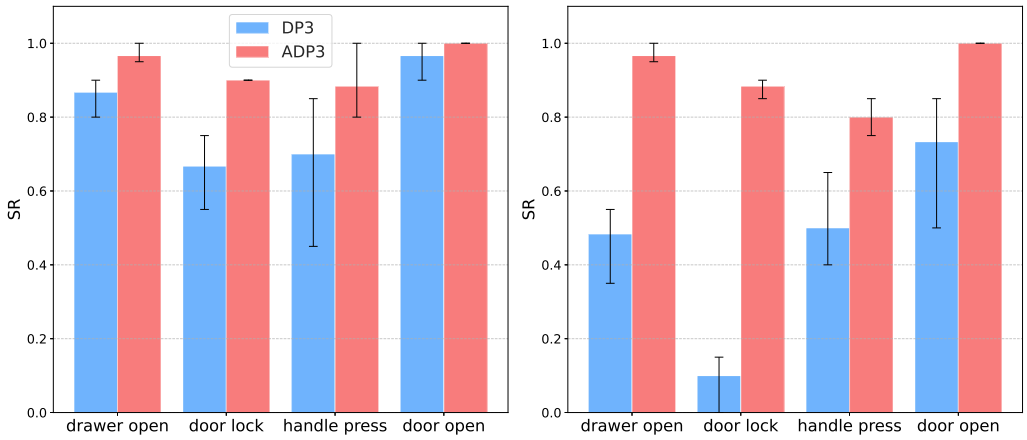
ADP3 consistently performs better than DP3 when distractor objects are present in the scene during training. In the *coffee push* task, the mug can be distinguished from the cylinder primarily by the presence of a handle, in addition to the size difference (See Figure 6). However, DP3 fails to distinguish the two, achieving a success rate of 53%, while ADP3 reaches 90%. A similar trend can be observed in other tasks, where the DP3 encoder fails to capture fine-grained



(a) Generalization to novel object instances

Task	DP3 Orig.	DP3 Novel	DP3 Δ	ADP3 (ours) Orig.	ADP3 (ours) Novel	ADP3 (ours) Δ
drawer open	87	48	-38	97	97	0
door lock	67	10	-57	90	88	-2
handle press	70	50	-20	88	80	-8
door open	97	73	-23	100	100	0
Average Drop			-35			-3

TABLE II: Success rates (%) for instance generalization



(b) Instance generalization performance

Fig. 8: **Instance generalization:** (a) For each of the four tasks: *drawer open*, *door lock*, *handle press*, and *door open*, training objects (left) and novel test objects (right) with significantly altered geometry. (b) Bar plots of success rates on training (original) and unseen (novel) objects. ADP3 (red) shows minimal performance degradation; DP3 (blue) struggles to generalize across shape and size variations.

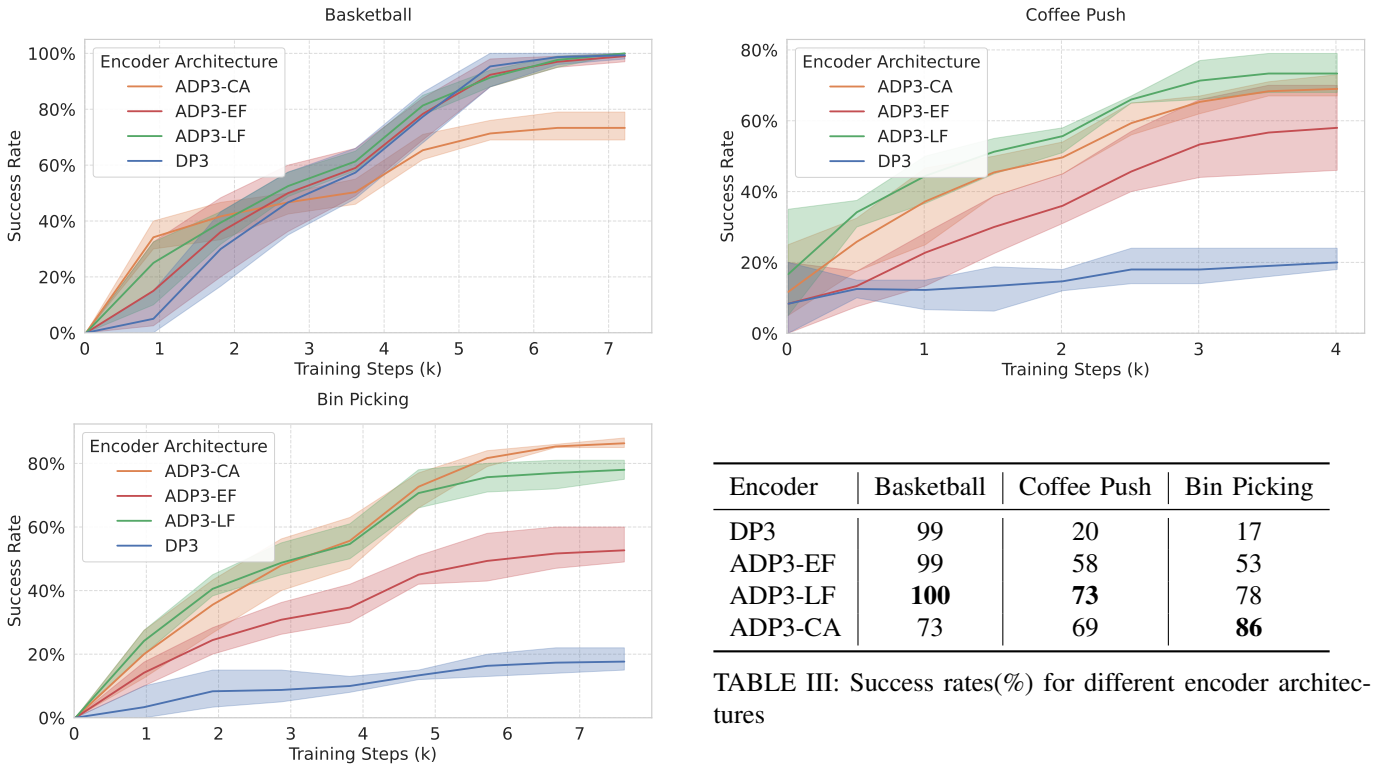
spatial information without affordance guidance.

3) Instance Generalization: The ability of the policy to generalize to objects of different shapes and sizes without the need for retraining remains an active research problem. This requires the policy to attend to task-relevant salient regions within the observation. As shown in previous experiments, the DP3 encoder often fails to extract sufficient spatial information for the diffusion head. To investigate whether this limitation persists in scenarios where the object itself varies, we conduct instance generalization experiments.

In our instance generalization experiments, the policies are trained on 4 Meta-World tasks using a specific object instance per task and evaluated on novel object instances (see Figure 7a). The unseen objects are generated by: 1) scaling the original object, thus altering object size significantly, such as in *door lock*, or 2) modifying the contours in task-relevant areas through warping or scaling, such as the handles in *drawer open* and *handle press*. The purpose of modifying interaction region geometry is to test whether the policy overfits to a specific local geometry. The functional aspects, such as the drawer's withdrawal distance or the door handle's rotation, remain unchanged.

a) Results and Analysis: The trained policy is evaluated on both objects: seen and unseen. Table II summarizes the success rates. The performance of both policies is comparable when the policy is rolled out on the original object, although ADP3 is consistently superior. However, the performance of DP3 drops significantly when the novel object is introduced. For the *door lock* task, which has the smallest interaction region among all tasks, we observe the largest success rate drop of 57%. The lock is represented by a sparse set of points, which can potentially have a lower influence on the latent representation. Since the encoders are trained from scratch alongside the policy head on a behavioral cloning loss, the policy can learn to complete the task by attending to various parts of the object. This can be the points on the top and side surfaces or the handle—and not necessarily the task-relevant region, i.e., the lock itself. The sparser the task-relevant region, the lower the probability that the local geometry is captured in the latent representation. At the same time, the performance dip is less pronounced when the interaction region is significantly larger.

In contrast, ADP3 showcases impressive performance when evaluated on novel object instances, with a mean performance



Encoder	Basketball	Coffee Push	Bin Picking
DP3	99	20	17
ADP3-EF	99	58	53
ADP3-LF	**100**	**73**	78
ADP3-CA	73	69	**86**

TABLE III: Success rates(%) for different encoder architectures

Fig. 9: Results from our experiments comparing different methods across tasks. Learning curves showing success rate vs. training steps for different affordance integration methods on three tasks. Late fusion achieves the best balance between training efficiency and task success rate.

drop of just 3% in comparison to DP3's 35%. It succeeds in ignoring changes to task-irrelevant features in the scene. Additionally, ADP3's performance remains stable even when the interaction region geometry is modified. No performance drop is observed for *drawer open*, while there is only a minor drop of 8% in the success rates on *handle press*, indicating that the policy does not overfit to a specific local geometry.

B. Ablation Studies

1) Encoder Architecture: We aim to integrate affordances into the observation representation to enable the policy to transfer the learned skill to unseen objects and environments. However, combining geometric information from the point cloud with semantics from affordance scores to form a good representation for policy learning is a challenging task. As illustrated in Figure 10, we evaluate three approaches for integrating affordance information into the observation: *early fusion*, *late fusion*, and *cross-attention*.

Besides the baseline pyramidal encoder, we evaluated the standard point cloud encoder PointNet++ and an ROI-seeded variant that ensures high-affordance points are not lost during downsampling. However, these did not outperform the baseline encoder and are discussed in further detail in Appendix A. We evaluate each affordance integration method on three manipulation tasks: *basketball*, *bin picking*, and *coffee push* (See Figure 4).

Table III presents the performance metrics for each encoder architecture across the three tasks. We report the success rates by averaging over five best policy checkpoints and three seeds as in [3]. The main findings from this experiment are:

a) Late Fusion extracts the best latent representation for policy learning: Although affordance signals aid in forming effective representations, the integration mechanisms plays an equally significant role. Despite being the most straightforward method for integrating affordances, early fusion performs relatively poorly on all tasks, albeit better than the baseline. The encoder struggles to extract a good latent representation and in separating affordance information from geometric features. This indicates that simply adding affordance as an additional channel is not enough to provide sufficient guidance. On the other hand, the late fusion approach consistently achieves excellent success rates on all tasks. Explicitly separating the processing paths for geometry and affordance allows the policy to quickly learn effective representations.

The cross-attention approach passes the separately processed scene and affordance encodings through a cross-attention layer instead of simply concatenating them. Although intuitively, a cross-attention layer at the end could better encode the relationship between geometry and semantics, the approach still falls behind. Except for the *bin picking* task for which it does exceptionally well, its overall performance is inferior to that of the late fusion approach. We hypothesize that this is due to the lack of information in the scene encoding extracted by the pyramidal encoder. The PointNet-style pyramidal encoder extracts a global feature vector from

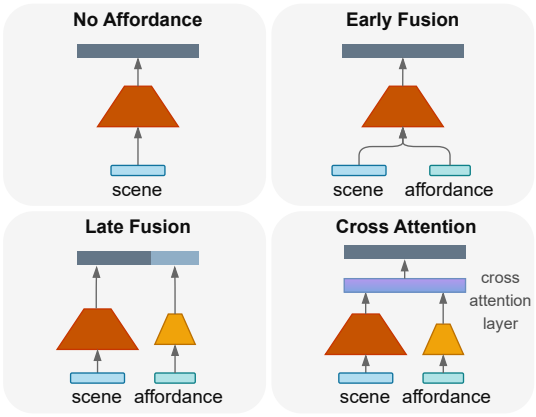


Fig. 10: **Experiments in Simulation.** We experiment with various mechanisms to integrate affordances with 3D observations. The main approaches are: (1) *No Affordance (Baseline)*: process only geometric information without affordance; (2) *Early Fusion*: concatenate affordance as an additional input channel before it is passed to the encoder; (3) *Late Fusion*: process geometric information and affordance separately, and combine at the last layer; (4) *Cross-Attention*: late fusion with a cross-attention layer to improve the association between geometry and semantics.

the entire point cloud using max-pooling. This leads to a loss of local geometry information, which the cross-attention layers have no way of recovering.

V. Real World Experiments

We include experiments on a real robot for *opening a bottle* and *picking a knife* to gauge the real-world effectiveness of our approach.

Hardware Setup. We use a 7-dof MAiRA7M robot and single-view observations obtained from an Orbbec Femto Bolt depth camera, which we choose for its good point cloud quality. Demonstrations are collected via teleoperation using a gamepad: 40 demonstrations for simple tasks.

Action Processing. We observed that absolute positions are more suitable as the action representation compared to delta positions. The action has 10 dimensions: 3 for position, 6 for orientation [51], and 1 for the gripper width.

Observation Processing. Point clouds are cropped to discard points from the background and the table. downsampled using uniform sampling to 4096 points. 2D affordance heatmaps, a key component of our approach, are obtained from RGB images using OOAL [5]. However, a single image could contain multiple objects with parts corresponding to the same affordance: for example, one can *hold* the handle of a cup as well as the strap of a bag. To avoid such ambiguity, we first obtain a bounding box using the Yolo World model [40] for open-vocabulary object detection before segmenting affordances. We transform the observations to the robot frame using the extrinsic calibration parameters in order to make it easy to define cropping bounds.

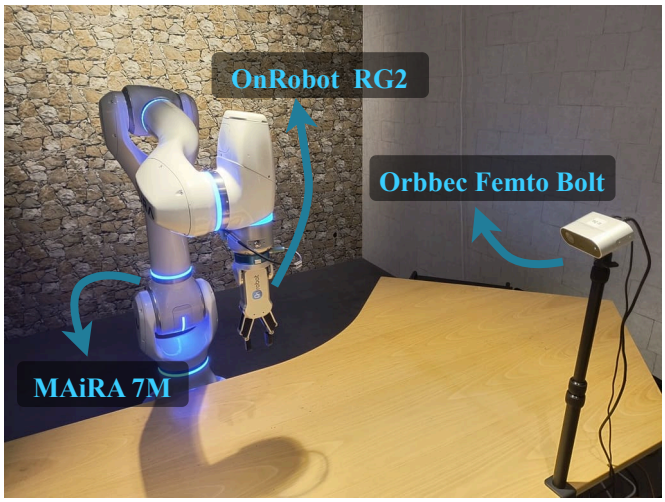


Fig. 11: Hardware setup

Inference. The entire pipeline operates at approximately 10 Hz, after parallelized observation and action processing, while running inference on an Nvidia RTX 3090 GPU. We additionally chunk action sequences across multiple inferences [11] to prevent sudden jerks when switching from one sequence to the next.

A. Evaluation in Clutter

A generalizable policy must be able to handle irrelevant changes in the environment that do not affect the task at hand. In this experiment, we train ADP3 and DP3 on the *bottle open* task with no other objects present and by randomizing the object position in every demonstration. We then evaluate the policy in two test scenarios: 1) one where only the target object in the scene serves as a benchmark, and 2) another where the scene is cluttered with objects of different shapes and sizes. The spatial arrangement of all objects is randomized during every run.

a) Results and Analysis: We run 15 policy rollouts for each trained policy, and the results are reported in Table IV. An evaluation run is counted as a success if the robot successfully grasps the cap and lifts it to a distance of 5 cm above the bottle.

	Only bottle	In clutter
DP3	100	0
ADP3	100	100

TABLE IV: Success rates (in percentage) with and without clutter for 15 evaluation episodes.

It is observed that both policies achieve a perfect task completion rate when only the bottle is present. However, once the clutter is introduced, the performance of DP3 dips drastically. Although DP3 was able to smoothly inch closer to the bottle without clutter, it struggled to generate meaningful action sequences, slowly drifting away from the object (see Figure 12). This is an expected phenomenon since training

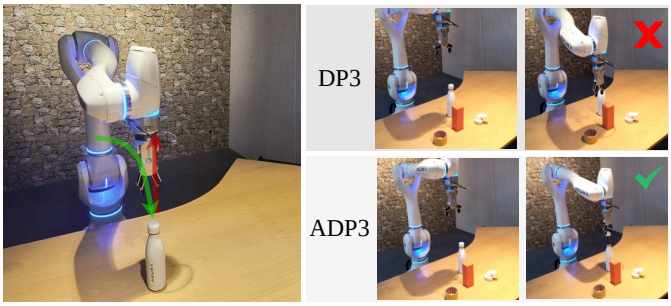


Fig. 12: **Evaluation in clutter. Left:** The demonstrations for the *open bottle* task consist of two motion segments that involve the robot: (1) starting from a fixed position, moving towards the bottle and aligning itself with the lid (green), and (2) closing the gripper and moving vertically upwards (red). **Right:** Both policies are rolled out in a cluttered scene. From the very start of the DP3 rollout, the robot starts drifting away from the target object and misses the grasp by a large margin. On the other hand, ADP3 is not affected by the clutter and manages to pick up the cap successfully.

without clutter and testing with clutter creates a major distribution shift. The performance degradation of DP3 can be linked to the information bottleneck due to the max pooling layers in the pyramidal encoder. The max pooling operation only selects the highest activation per feature channel present in the scene encoding (128 channels for DP3 and 96 for ADP3). In cluttered scenes, other objects can create strong feature activations that can easily overpower the bottle's feature activations.

Contrastingly, ADP3 retains the same performance even in clutter and manages to successfully complete the task on all 15 trials. ADP3 uses the same scene encoder architecture as DP3 to generate scene encodings. In cluttered scenes with completely randomized spatial arrangements, the scene encoding generated should experience a similar disruption as observed for DP3. However, ADP3 can handle significant distribution shifts by using the affordance encoding to obtain sufficient information.

B. Generalization to Novel Object Configurations

Besides robustness to clutter, ADP3 additionally demonstrates the ability to generalize to object configurations unseen during training. The policy is trained on demonstrations of the robot picking a specific knife (see Figure 13), in which the knife's location is altered without changing its orientation (with the blade pointing toward the robot base). Once trained, the policy is rolled out for two object configurations, similar to [46]: 1) **normal configuration** (the blade pointing towards the robot base), and 2) **flipped configuration** (the blade pointing away from the robot base). The policy is evaluated through 15 trials for each policy and configuration.

Table V reports the results for both configurations. Besides the SR, we additionally include the number of "unsafe" grasps attempted by the policy. A grasp attempt is counted as unsafe if the robot touches the blade. In the normal configuration, affordance guidance does not offer a noticeable performance gain, with both policies achieving an SR of 80%. However, the

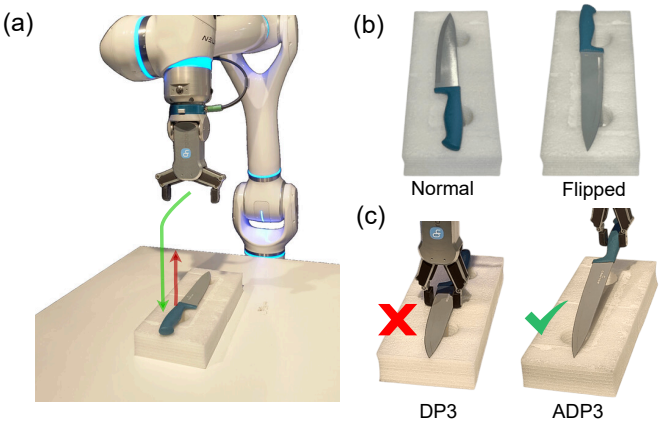


Fig. 13: **Picking a knife: (a) Task Description.** The robot moves towards the knife, aligning the end-effector with the knife handle, closes the gripper and moves upwards by about 5 cm. **(b) Two Evaluation Configurations.** The figure illustrates the normal configuration that the policy was trained on and the flipped configuration to test object-part understanding. **(c) Safe Behavior.** DP3 does not differentiate the handle from the blade and grasps the blade. ADP3 exhibits safe and predictable behavior, never attempting to grasp the blade.

Config.	DP3		ADP3	
	SR	Unsafe Grasps	SR	Unsafe Grasps
Normal	80.0	0/15	80.0	0/15
Flipped	13.3	7/15	**53.0**	**0/15**

TABLE V: Success Rate (SR) and Unsafe Grasps ($n/15$) for different knife orientations.

performance of DP3 drops to just 13% in the flipped configuration. In the failure cases, the robot either drifts away from the object or attempts to grasp the blade. This indicates that DP3 fails to effectively capture the intent behind the demonstration which is to grasp the handle, and almost consistently attempts to grasp the blade. Moreover, in the two trials (out of 15) in which DP3 managed to grasp the knife handle, the knife was placed at the extremities of the demonstration distribution. The trained diffusion head outputs absolute positions that approximately lie within these extremities. The successful grasps were only recorded at object positions in which only the handle points are inside this distribution. Hence, even in the few successful grasps, DP3 does not demonstrate object part-level understanding. In contrast, ADP3 never attempts to grasp the blade even though the SR drops to 53%.

VI. Conclusion

In this work, we introduce Affordance-Guided 3D Diffusion Policy (ADP3), an approach that tackles the current challenges in training generalizable visual imitation learning policies. By fusing object interaction knowledge from foundation models into point clouds, we generate 3D affordance heatmaps that improve generalization. By evaluating the policy in 8 simulated tasks and real-world experiments, we demonstrate the ef-

fectiveness of affordance guidance in policy learning. Through extensive encoder ablations, we report the effectiveness of a two-stream approach in fusing affordance information to form meaningful latent representations. Moreover, we conduct various experiments that indicate the robustness of the approach to the presence of distractors in the scene. We also showcase the ability of our method to generalize to novel objects in simulated environments with no retraining, even when the interaction region geometry drastically varies. Finally, we conduct real-world experiments that evaluate ADP3 in cluttered scenes and novel object orientations, further highlighting the efficacy of affordance guidance.

Limitations and Future Work. Although our work improves the generalization abilities of 3D Diffusion Policy, our method has several limitations. Firstly, obtaining accurate 2D affordance heatmaps for all objects encountered in the real-world is still a challenge. Occlusions and the presence of several objects in close proximity can make affordance regions harder to extract. Moreover, the use of 3D semantic maps leads to a loss of visual context in comparison to images, and the policy could struggle with tasks that require complex understanding of color and texture. Future work can address this issue by exploring ways to inject task-specific visual context, beyond affordances, through pretrained visual representations.

VII. Acknowledgments

I would like to extend my gratitude to Dr. Jens Kober for his valuable guidance and support. I am equally grateful to Jeyhoon Maskani for the frequent and thought-provoking discussions that often resulted in good ideas. A special thanks to Milad Malekzadeh for giving me the opportunity to carry out my research, and for providing me with necessary hardware and computing resources. Finally, I thank Anna Vorontsova for the insightful discussions and reviews that significantly improved my work.

References

[1] 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," *The International Journal of Robotics Research*, p. 02783649241273668, Oct. 2024. Publisher: SAGE Publications Ltd STM.

[2] F. Lin, Y. Hu, P. Sheng, C. Wen, J. You, and Y. Gao, "Data Scaling Laws in Imitation Learning for Robotic Manipulation," Oct. 2024. arXiv:2410.18647 [cs].

[3] Y. Ze, G. Zhang, K. Zhang, C. Hu, M. Wang, and H. Xu, "3D Diffusion Policy: Generalizable Visuomotor Policy Learning via Simple 3D Representations," in *Robotics: Science and Systems XX*, Robotics: Science and Systems Foundation, July 2024.

[4] J. J. Gibson, *The ecological approach to visual perception: classic edition*. Psychology press, 2014.

[5] G. Li, D. Sun, L. Sevilla-Lara, and V. Jampani, "One-shot open affordance learning with foundation models," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 3086–3096, 2024.

[6] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, *et al.*, "Openvla: An open-source vision-language-action model," *arXiv preprint arXiv:2406.09246*, 2024.

[7] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, L. X. Shi, J. Tanner, Q. Vuong, A. Walling, H. Wang, and U. Zhilinsky, "π_0: A vision-language-action flow model for general robot control," 2024.

[8] J. Bjorck, F. Castañeda, N. Cherniadev, X. Da, R. Ding, L. Fan, Y. Fang, D. Fox, F. Hu, S. Huang, *et al.*, "Gr00t n1: An open foundation model for generalist humanoid robots," *arXiv preprint arXiv:2503.14734*, 2025.

[9] N. M. Shafiullah, Z. Cui, A. A. Altanzaya, and L. Pinto, "Behavior transformers: Cloning k modes with one stone," *Advances in neural information processing systems*, vol. 35, pp. 22955–22968, 2022.

[10] P. Florence, C. Lynch, A. Zeng, O. Ramirez, A. Wahid, L. Downs, A. Wong, J. Lee, I. Mordatch, and J. Tompson, "Implicit Behavioral Cloning," Sept. 2021. arXiv:2109.00137 [cs].

[11] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, "Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware," Apr. 2023. arXiv:2304.13705 [cs].

[12] A. Zeng, P. Florence, J. Tompson, S. Welker, J. Chien, M. Attarian, T. Armstrong, I. Krasin, D. Duong, V. Sindhwani, *et al.*, "Transporter networks: Rearranging the visual world for robotic manipulation," in *Conference on Robot Learning*, pp. 726–747, PMLR, 2021.

[13] S. James, K. Wada, T. Laidlow, and A. J. Davison, "Coarse-to-fine q-attention: Efficient learning for visual robotic manipulation via discretisation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13739–13748, 2022.

[14] T. Gervet, Z. Xian, N. Gkanatsios, and K. Fragkiadaki, "Act3d: 3d feature field transformers for multi-task robotic manipulation," in *7th Annual Conference on Robot Learning*, 2023.

[15] T.-W. Ke, N. Gkanatsios, and K. Fragkiadaki, "3D Diffuser Actor: Policy Diffusion with 3D Scene Representations," July 2024. arXiv:2402.10885 [cs].

[16] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, and A. Joulin, "Unsupervised learning of visual features by contrasting cluster assignments," *Advances in neural information processing systems*, vol. 33, pp. 9912–9924, 2020.

[17] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," in *International conference on machine learning*, pp. 1597–1607, PmLR, 2020.

[18] S. Parisi, A. Rajeswaran, S. Purushwalkam, and A. Gupta, "The Unsurprising Effectiveness of Pre-

Trained Vision Models for Control," Aug. 2022. arXiv:2203.03580 [cs].

[19] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi, "Dream to Control: Learning Behaviors by Latent Imagination," Mar. 2020. arXiv:1912.01603 [cs].

[20] C. Gelada, S. Kumar, J. Buckman, O. Nachum, and M. G. Bellemare, "DeepMDP: Learning Continuous Latent Space Models for Representation Learning," in *Proceedings of the 36th International Conference on Machine Learning*, pp. 2170–2179, PMLR, May 2019. ISSN: 2640-3498.

[21] A. Zhang, R. McAllister, R. Calandra, Y. Gal, and S. Levine, "Learning Invariant Representations for Reinforcement Learning without Reconstruction," Apr. 2021. arXiv:2006.10742 [cs].

[22] L. Yen-Chen, A. Zeng, S. Song, P. Isola, and T.-Y. Lin, "Learning to See before Learning to Act: Visual Pre-training for Manipulation," July 2021. arXiv:2107.00646 [cs].

[23] R. Shah and V. Kumar, "RRL: Resnet as representation for Reinforcement Learning," Nov. 2021. arXiv:2107.03380 [cs].

[24] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick, "Masked autoencoders are scalable vision learners," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16000–16009, 2022.

[25] A. Stone, T. Xiao, Y. Lu, K. Gopalakrishnan, K.-H. Lee, Q. Vuong, P. Wohlhart, S. Kirmani, B. Zitkovich, F. Xia, C. Finn, and K. Hausman, "Open-World Object Manipulation using Pre-trained Vision-Language Models," Oct. 2023. arXiv:2303.00905 [cs].

[26] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta, "R3M: A Universal Visual Representation for Robot Manipulation," Nov. 2022. arXiv:2203.12601 [cs].

[27] A. Majumdar, K. Yadav, S. Arnaud, J. Ma, C. Chen, S. Silwal, A. Jain, V.-P. Berges, T. Wu, J. Vakil, *et al.*, "Where are we in the search for an artificial visual cortex for embodied intelligence?," *Advances in Neural Information Processing Systems*, vol. 36, pp. 655–677, 2023.

[28] Y. J. Ma, S. Sodhani, D. Jayaraman, O. Bastani, V. Kumar, and A. Zhang, "VIP: Towards Universal Visual Reward and Representation via Value-Implicit Pre-Training," Mar. 2023. arXiv:2210.00030 [cs].

[29] K. Mo, L. Guibas, M. Mukadam, A. Gupta, and S. Tulsiani, "Where2Act: From Pixels to Actions for Articulated 3D Objects," Aug. 2021. arXiv:2101.02692 [cs].

[30] Y. Wang, R. Wu, K. Mo, J. Ke, Q. Fan, L. J. Guibas, and H. Dong, "AdaAfford: Learning to Adapt Manipulation Affordance for 3D Articulated Objects via Few-Shot Interactions," in *Computer Vision – ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXIX*, (Berlin, Heidelberg), pp. 90–107, Springer-Verlag, Oct. 2022.

[31] Y. Geng, B. An, H. Geng, Y. Chen, Y. Yang, and H. Dong, "RLAfford: End-to-End Affordance Learning for Robotic Manipulation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5880–5886, May 2023.

[32] T.-T. Do, A. Nguyen, and I. Reid, "AffordanceNet: An End-to-End Deep Learning Approach for Object Affordance Detection," Mar. 2018. arXiv:1709.07326 [cs].

[33] M. Goyal, S. Modi, R. Goyal, and S. Gupta, "Human Hands as Probes for Interactive Object Understanding," Apr. 2022. arXiv:2112.09120 [cs].

[34] T. Nagarajan, C. Feichtenhofer, and K. Grauman, "Grounded Human-Object Interaction Hotspots from Video," Apr. 2019. arXiv:1812.04558 [cs].

[35] S. Liu, S. Tripathi, S. Majumdar, and X. Wang, "Joint Hand Motion and Interaction Hotspots Prediction from Egocentric Videos," in *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (New Orleans, LA, USA), pp. 3272–3282, IEEE, June 2022.

[36] S. Bahl, R. Mendonca, L. Chen, U. Jain, and D. Pathak, "Affordances from Human Videos as a Versatile Representation for Robotics," in *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, (Vancouver, BC, Canada), pp. 01–13, IEEE, June 2023.

[37] Y. Ju, K. Hu, G. Zhang, G. Zhang, M. Jiang, and H. Xu, "Robo-ABC: Affordance Generalization Beyond Categories via Semantic Correspondence for Robot Manipulation," Jan. 2024. arXiv:2401.07487 [cs].

[38] Y. Kuang, J. Ye, H. Geng, J. Mao, C. Deng, L. Guibas, H. Wang, and Y. Wang, "RAM: Retrieval-Based Affordance Transfer for Generalizable Zero-Shot Robotic Manipulation," July 2024. arXiv:2407.04689 [cs].

[39] H. Chen, B. Sun, A. Zhang, M. Pollefeys, and S. Leutenegger, "VidBot: Learning Generalizable 3D Actions from In-the-Wild 2D Human Videos for Zero-Shot Robotic Manipulation," Mar. 2025. arXiv:2503.07135 [cs].

[40] T. Cheng, L. Song, Y. Ge, W. Liu, X. Wang, and Y. Shan, "Yolo-world: Real-time open-vocabulary object detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16901–16911, 2024.

[41] Y. Ze, Z. Chen, W. Wang, T. Chen, X. He, Y. Yuan, X. B. Peng, and J. Wu, "Generalizable humanoid manipulation with improved 3d diffusion policies," *arXiv preprint arXiv:2410.10803*, 2024.

[42] J. Yang, Z.-a. Cao, C. Deng, R. Antonova, S. Song, and J. Bohg, "EquiBot: SIM(3)-Equivariant Diffusion Policy for Generalizable and Data Efficient Learning," Oct. 2024. arXiv:2407.01479 [cs].

[43] D. Wang, S. Hart, D. Surovik, T. Kelestemur, H. Huang, H. Zhao, M. Yeatman, J. Wang, R. Walters, and R. Platt, "Equivariant Diffusion Policy," Oct. 2024. arXiv:2407.01812 [cs].

[44] S. Wang, J. You, Y. Hu, J. Li, and Y. Gao, "SKIL: Semantic Keypoint Imitation Learning for Generalizable Data-efficient Manipulation," Jan. 2025. arXiv:2501.14400 [cs].

[45] K. Rana, J. Abou-Chakra, S. Garg, R. Lee, I. Reid, and N. Suenderhauf, "Affordance-Centric Policy Learning: Sample Efficient and Generalisable Robot Policy Learning using Affordance-Centric Task Frames," Oct. 2024. arXiv:2410.12124 [cs].

[46] Y. Wang, G. Yin, B. Huang, T. Kelestemur, J. Wang, and Y. Li, "GenDP: 3D Semantic Fields for Category-Level Generalizable Diffusion Policy," Oct. 2024. arXiv:2410.17488 [cs].

[47] H. Li, Q. Feng, Z. Zheng, J. Feng, and A. Knoll, "Language-Guided Object-Centric Diffusion Policy for Collision-Aware Robotic Manipulation," July 2024. arXiv:2407.00451 [cs].

[48] T. Yu, D. Quillen, Z. He, R. Julian, A. Narayan, H. Shively, A. Bellathur, K. Hausman, C. Finn, and S. Levine, "Meta-World: A Benchmark and Evaluation for Multi-Task and Meta Reinforcement Learning," June 2021. arXiv:1910.10897 [cs].

[49] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," 2017.

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

[51] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li, "On the continuity of rotation representations in neural networks," 2020.

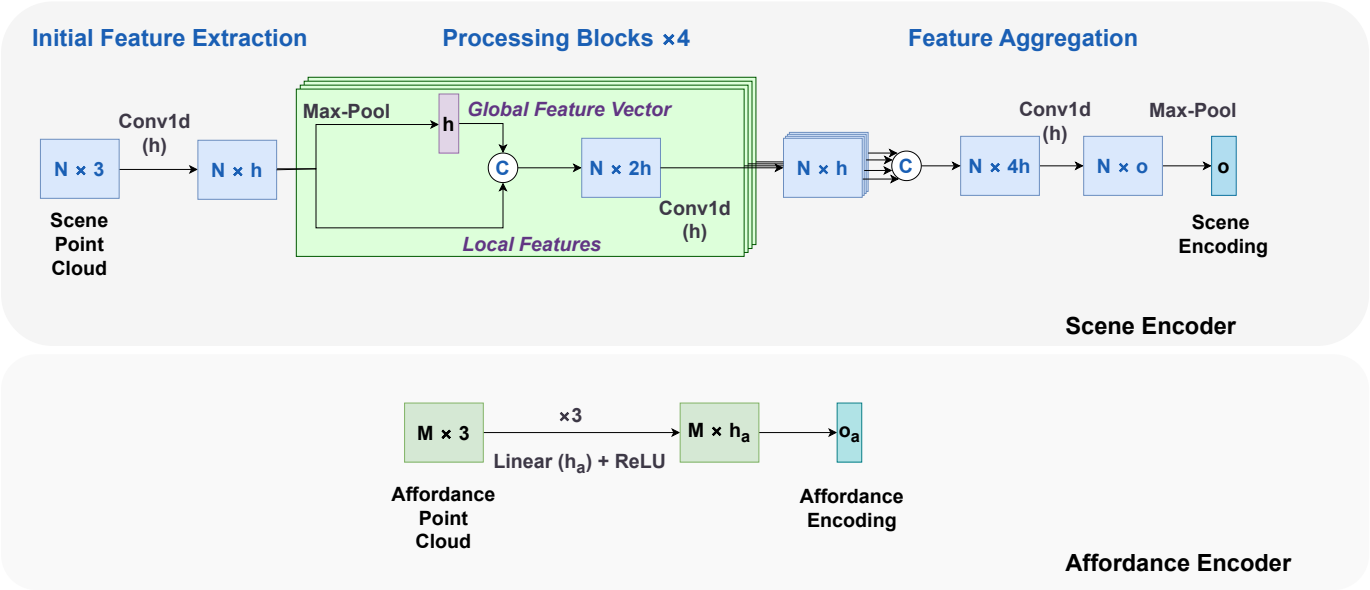


Fig. 14: **Detailed Network Architecture** The architecture consists of two parallel processing streams. **Top:** The scene encoder ([41]) processes the entire point cloud using an initial feature extraction layer (Conv1d-h & max-pooling), followed by four processing blocks that combine local features with global context (features). Note that after the first processing block, the subsequent processing blocks receive the feature vector processed by the previous processing block. The four feature vectors from the processing blocks are concatenated and aggregated to extract a scene encoding of dimension o. **Bottom:** The affordance encoder processes points above a threshold using three fully-connected layers with ReLU activations, extracting in an affordance encoding of dimension o_a. The final representation concatenates both encodings. Here, N denotes the number of input points, M the number of affordance points ($M \leq N$), h the hidden dimension, o the scene encoding dimension, h_a (128) the affordance hidden dimension and o_a the affordance encoding dimension. C represents the concatenation operation.

APPENDIX

A. Detailed Encoder Ablations

In our work, we use a two-stream approach to process the scene and affordance point clouds separately. We found that fusing the encodings using two streams to form the observation representation was the most effective strategy to integrate affordances. The scene encoder is a PointNet-style pyramidal encoder (see Figure 14), while the affordance encoder is a simple MLP. The scene encoder differs from the PointNet encoder in a few significant ways, similar to [41]:

- **No Batch Normalization (BN).** BN negatively affects Exponential Moving Average (EMA) that stabilizes the training dynamics of the diffusion model.
- **No T-Net layers.** T-Net is a small network used inside the PointNet architecture that predicts transformations to align input point clouds to attain transformation invariance. Such invariance is beneficial in 3D object classification and segmentation since the embedding does not change with the object's spatial orientation. However, in our case, the policy needs access to the exact spatial position and orientation of the target object in order to predict accurate action sequences. Some methods have experimented with equivariant latent representations as a mechanism to overcome this issue but are still not suitable for generalizable policy learning [42, 43]. This is because equivariance relies on scene symmetry and works best when there is a single object in the scene, which is rarely the case in the real world.
- **Captures global features more effectively.** The PointNet architecture is incapable of capturing local geometry by design [50]. The pyramidal encoder overcomes this limitation partially by repeatedly sharing global features with local features as illustrated in Figure 14.

Processing scene and affordance point clouds separately and then fusing the features has already been shown to be more effective than early fusion. However, a more obvious design decision to encode point clouds would be the use of standard point-cloud encoders such as PointNet and PointNet++. Even though these cloud encoders performed poorly in the encoder ablation studies in [3], they are yet to be evaluated on 3D affordance heatmaps. We list the encoders we select for our extended encoder ablation studies on Meta-World tasks *basketball*, *coffee push*, and *bin picking*:

1) Pyramidal Encoder (DP3): We include the pyramidal encoder which does not leverage affordance information as the baseline.

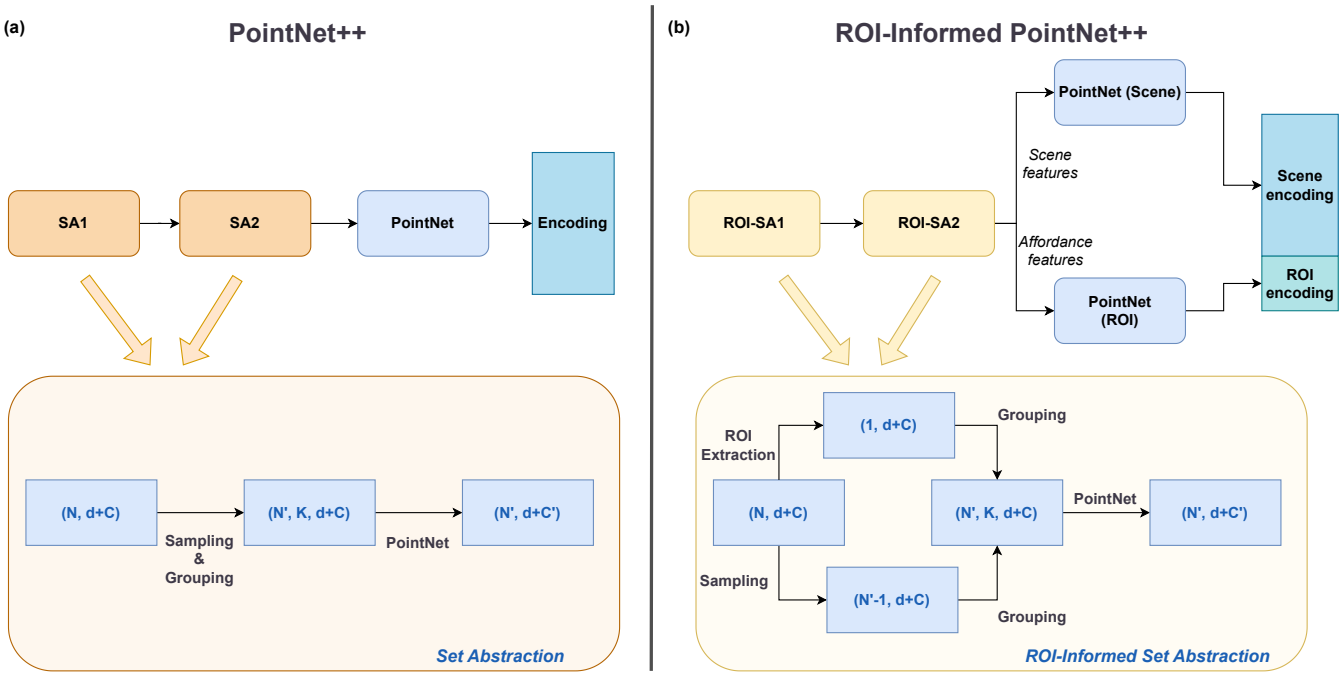


Fig. 15: **(a) PointNet++ architecture.** PointNet++ captures local geometry effectively through a hierarchical approach, using two SA layers. Centroids are sampled using FPS and the local geometry at these centroids are extracted. The sparse affordance regions could be discarded in this sampling process, which can detrimentally affect policy learning. **(b) ROI-Informed PointNet++ architecture.** Using ROI-Informed SA layers, ensure that the affordance points are not discarded.

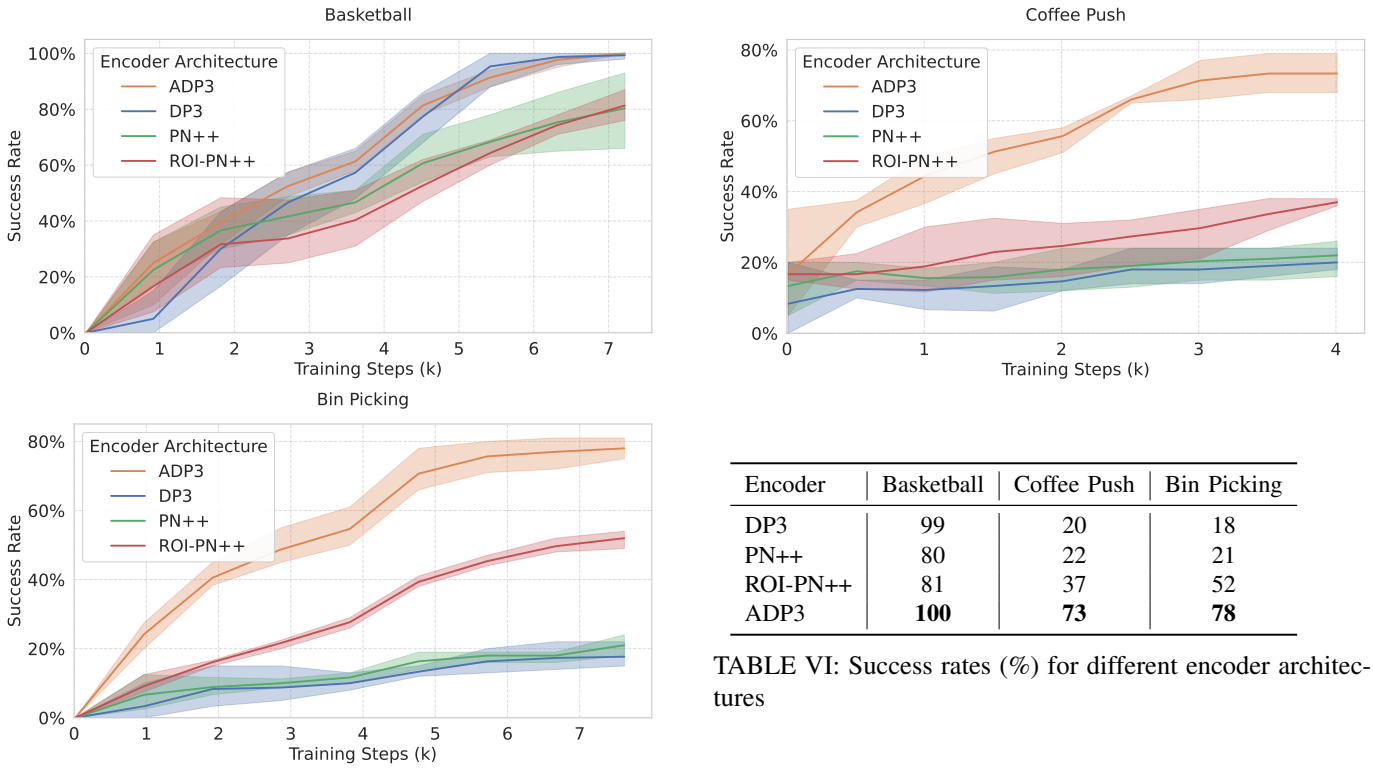
2) PointNet++ (PN++): Even though the pyramidal encoder allows for better local geometry awareness than the PointNet encoder, it misses some key features that PointNet++ possesses. PointNet++ uses Set Abstraction (SA) layers to encode a point cloud. Our point clouds have $d = 3$, consisting of the cartesian coordinates, and $C = 1$ for the affordance channel. In SA layers, a subset of the points are selected as centroids (*sampling*), and points in the local neighborhood are gathered (*grouping*), as depicted in Figure 15. Subsequently, a small PointNet encoder extracts local features from each group. This hierarchical approach increases receptive field as the layers get deeper, and effectively captures local as well as global geometry.

3) ROI-Informed PointNet++ (ROI-PN++): Due to the sampling process within the SA layers of the PN++ encoder, it cannot be guaranteed that affordance regions are represented in the embedding. To address this, we modify the PN++ architecture by changing the sampling stage as illustrated in Figure 15. In each SA layer inside the vanilla PN++ encoder, we sample N' centroids from N' points. Instead of sampling N' points, we first extract an Region Of Interest (ROI) point $(1, d + C)$. ROI extraction is simply selecting the point (and corresponding features) that has the highest affordance score. We then sample the rest of the points $(N' - 1)$ using FPS, and concatenate it with the affordance point, resulting in N' centroids. This explicit processing step ensures that the output of the SA layer contains the affordance features. This structure is preserved throughout the first two SA layers. We then process the affordance and scene features through separate PointNets, obtaining two 1D features vectors. We concatenate these in order to obtain the final observation embedding. This approach allows for improved control over the affordance signal without increasing the network size significantly, simply by modifying the sampling procedure and processing the regions of interest separately at the final stage.

4) ADP3 Encoder (ADP3): We also include our best-performing encoder that separately processes scene and affordance points, followed by *late-fusion*, for better comparison.

Results and Analysis. We illustrate the learning curves of the policies trained with various encoder architectures in Figure 16. It is observed that despite providing an affordance signal, PN++ barely outperforms the baseline, which has no access to affordance information. PN++ is unable to extract task-relevant latent representations which the policy can use to improve task performance. The affordance points are sparse in comparison to the scene points: on average, 24 out of 4096 points belong to the affordance region for the bin-picking task.

In contrast, by modifying the sampling process, PN++ improves the ensuring that the task performance by 31% in the *bin picking* task and by 15% in the *coffee push* task. Since the affordance points are ensured to be present and well-represented in the observation encoding, ROI-PN++ significantly improves the task performance. However, it is still not consistently better



Encoder	Basketball	Coffee Push	Bin Picking
DP3	99	20	18
PN++	80	22	21
ROI-PN++	81	37	52
ADP3	**100**	**73**	**78**

TABLE VI: Success rates (%) for different encoder architectures

Fig. 16: Learning curves and final success rates for different encoder architectures for three manipulation tasks. PointNet++-based encoders underperform in comparison to the ADP3 encoder (late fusion).

than the DP3 encoder as observed in the *basketball* task, where ROI-PN++ only reaches an SR 0f 80% in comparison to DP3's 99%. Moreover, it is inferior to ADP3, a much simpler approach of processing the scene and affordance point clouds separately. We hypothesize that this performance gap is due to the lack of a loss function that trains the encoder to generate rich observation embeddings. We train the encoder from scratch along with the policy on a behavioral cloning loss which is found to be insufficient on its own to generate meaningful representations. This could get exacerbated as the network complexity increases, which is the case for ROI-PN++. Future work could involve exploring pretraining strategies with reconstruction or object-part segmentation objectives, and fine-tune the encoder on the specific task.

B. Sample Efficiency

Sample efficiency is an important aspect of scaling imitation learning policies since collecting several demonstrations can be a time-consuming and labor-intensive task. We test the sample efficiency of our policies by training on 2^n demonstrations, where $n \in \{1, 2, 3, 4, 5\}$. We train on two Meta-World tasks: *handle press* and *coffee push* and average the results over seeds 0, 1, and 2.

# Episodes	SR (%)		Normalized SR (%)	
	DP3	ADP3	DP3	ADP3
2	33	38	71	40
4	42	84	90	88
8	43.5	87	94	91
16	46.5	92	100	96
32	46.5	95.5	100	100

TABLE VII: Comparison of absolute and normalized success rates (%) averaged across two tasks. Normalized SR values are obtained by scaling the SR with respect to each policy's peak SR. Both methods have comparable sample efficiencies.

It is observed that although ADP3 significantly outperforms DP3 in terms of peak SR, there is no clear evidence to make a case for either policy in terms of sample efficiency. As indicated by the normalized SR, both policies cross 90% of their maximum success rates with just 4 demonstrations.

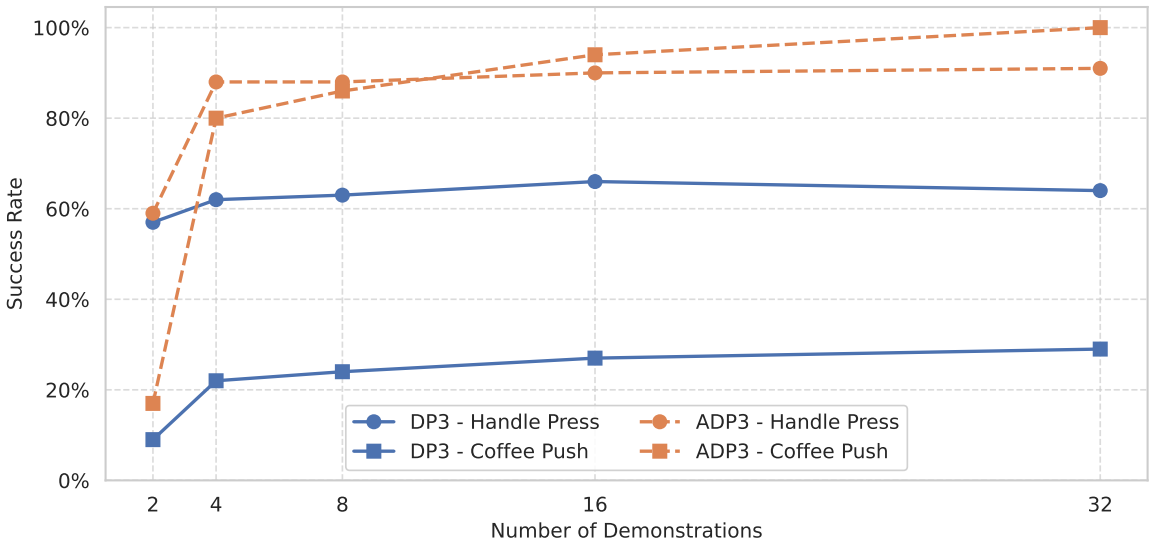


Fig. 17: Success rate as a function of the number of demonstrations for *handle press* and *coffee push*.

C. Implementation Details

1) Hyperparameters: We report the hyperparameters of our diffusion model in Table VIII; we use nearly identical hyperparameters for simulated and real-world tasks. Both states and actions are normalized to the range $[-1, 1]$ during training. The action horizon denotes the number of future actions predicted by the policy, and the action steps are the number of steps actually executed by the policy. The observation steps are the number of observation frames which the policy conditions its actions on. The hyperparameters for the diffusion model not reported here remain the same as in [1].

Hyperparameter	**Value**
General Configuration	
Action Horizon	16
Observation Steps	2
Action Steps	8
Latency Steps	0 (sim), 4 (real)
Diffusion Parameters	
Diffusion Steps (Training)	50
Inference Steps	16
Noise Schedule Function	Squared Cosine
Training Parameters	
Batch Size	64
Learning Rate	3.0e-4
Optimizer	AdamW
Number of Epochs	401
LR Warmup Steps	500

TABLE VIII: **Hyperparameters of the Model**