Robotics: Science and Systems 2024
Delft, Netherlands, July 15-July 19, 2024

Diffusion Meets DAgger: Supercharging Eye-in-hand Imitation Learning

Xiaoyu Zhang Matthew Chang Pranav Kumar Saurabh Gupta
University of Illinois at Urbana-Champaign
{zhang401,mc48,pranav9,saurabhg}@illinois.edu
https://sites.google.com/view/diffusion-meets-dagger

***Abstract*—A common failure mode for policies trained with imitation is compounding execution errors at test time. When the learned policy encounters states that are not present in the expert demonstrations, the policy fails, leading to degenerate behavior. The Dataset Aggregation, or DAgger approach to this problem simply collects more data to cover these failure states. However, in practice, this is often prohibitively expensive. In this work, we propose Diffusion Meets DAgger (DMD), a method that reaps the benefits of DAgger but without the cost, for eye-in-hand imitation learning problems. Instead of *collecting* new samples to cover out-of-distribution states, DMD uses recent advances in diffusion models to *synthesize* these samples. This leads to robust performance from few demonstrations. We compare DMD against behavior cloning baseline across four tasks: pushing, stacking, pouring, and hanging a shirt. In pushing, DMD achieves 80% success rate with as few as 8 expert demonstrations, where naive behavior cloning reaches only 20%. In stacking, DMD succeeds on average 92% of the time across 5 cups, versus 40% for BC. When pouring coffee beans, DMD transfers to another cup successfully 80% of the time. Finally, DMD attains 90% success rate for hanging shirt on a clothing rack.**

I. Introduction

Imitation learning is an effective way to train robots for new tasks. However, even when testing policies in environments similar to those used for training, imitation learning suffers from the well-known *Compounding Execution Errors problem*: small errors made by the policy lead to out-of-distribution states, causing the robot to make even bigger errors (compounding errors) [56].

One solution to this problem is via manual collection of expert-labeled data on states visited by the learner, a strategy popularly known as Dataset Aggregation or DAgger [56]. However, DAgger [56] is challenging to put into practice: it requires an expert operator to supervise the robot during execution and guide it to recover from failures. Many alternatives have been proposed, *e.g.* [35, 36], but they all require collecting more expert data. In this paper, we pursue an alternate paradigm: *automatically generating observations and action labels for out-of-distribution states*. By replacing data collection with data creation, we improve the sample efficiency of imitation learning. In fact, data creation was Pomerleau's original solution to this problem [52]. We revisit, improve, and automate his solution from 30 years ago using modern data-driven image generation methods.

We target imitation learning problems in the context of eye-in-hand setups (*i.e.* setups where images come from a camera mounted on the robot hand), which are becoming increasingly more popular [64, 78, 85]. Given a behavior cloning trajectory τ collected from an expert, we design generative models to synthesize off-distribution states and compute corresponding action labels. For example, in the object pushing application shown in Figure 1, this corresponds to generating off-center views of the object from the good expert trajectory τ. Specifically, we learn a function $f(I_t, \Delta p)$ that synthesizes the observation $\tilde{I}_t$ at a small perturbation Δp to the trajectory at time step t. Δp lets us compute the ground truth action label $\tilde{a}_t$ for this out-of-distribution observation. We augment the expert demonstration τ with multiple off-distribution samples $(\tilde{I}_t, \tilde{a}_t)$ along the trajectory.

As we have access to the entire trajectory, one option is to

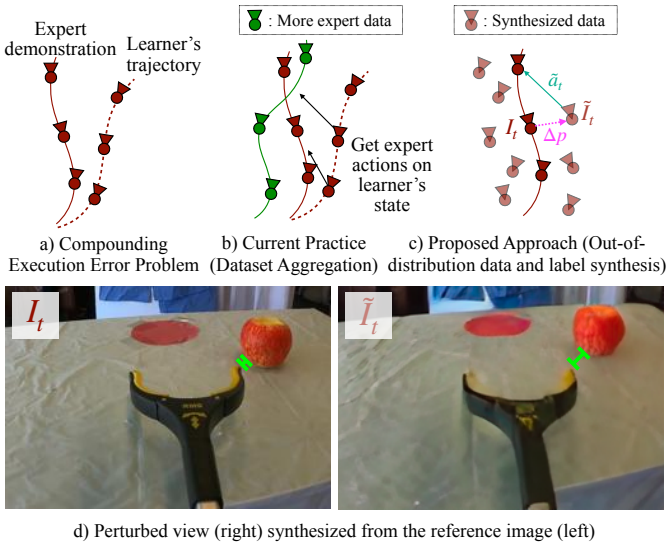

Fig. 1: Eye-in-hand Imitation learning with DMD: A common failure mode in an imitation learning setting is the problem of poor generalization due to compounding execution errors at test time as shown in (a). This can be solved by collecting more expert data to cover these off-trajectory states as shown in (b) however, this is an expensive process. Our proposed approach is to *synthesize* data instead of *collecting* it (c). Magenta arrow represents small perturbation (Δp) to the trajectory. Cyan arrow represents label ($\tilde{a}_t$) for this out-of-distribution observation. We use a state-of-the-art diffusion model to take images I_t from expert demonstrations (d, left) and generate realistic off-trajectory images $\tilde{I}_t$ (d, right). Note the distance between the grabber and the apple denoted by the green line. This synthetic data augments expert demonstrations for policy learning, leading to more robust policies.

realize $f(I_t, \Delta p)$ using NeRFs [48, 86]. While NeRFs work very well for view synthesis in computer vision, we find they are not suitable for the synthesis task at hand because of the inevitable deformations in the scene as the gripper manipulates the objects. We thus switch to using diffusion models, and this change leads to high-quality image synthesis even when the scene deforms during manipulation. Computing action labels for these samples present yet another challenge (Figure 5). The labels should align with progress towards the goal. We thus investigate different schemes for sampling Δp's and computing action labels $\tilde{a}_t$.

We present experiments that evaluate the aforementioned design choices in developing a data creation framework to supercharge eye-in-hand imitation learning. The framework is tested in four settings: non-prehensile pushing, stacking, pouring, and hanging a shirt. Across all tasks, we see a sizeable improvement over vanilla behavior cloning, demonstrating the effectiveness of our framework Diffusion Meets DAgger (DMD). For the task of non-prehensile pushing, DMD achieves 80% success rate given only 8 expert demonstrations, while behavior cloning (BC) reaches only 20%. It also outperforms a NeRF-based augmentation method [86] by 50%. For the task of stacking cups on a box, DMD achieves a success rate of 93% on average across three training cups, and 85% success for two unseen cups. In comparison, BC only reaches 36% and 45% success rates, respectively. For the task of pouring coffee beans into a cup, DMD succeeds 80% of the time, while BC falls short at 30%. Finally, DMD achieves a success rate of 90% for the hanging shirt task, improving upon the 20% success rate for BC.

II. Related Work

A. Imitation Learning

Behavior cloning (BC), or training models to mimic expert behavior, has been a popular strategy to train robots over the last many decades [51, 52, 58], and has witnessed renewed interest in recent times [29, 88]. [56] theoretically and empirically demonstrate the compounding execution error problem (small errors in the learner's predictions steer the agent into out-of-distribution states causing the learner to make even bigger errors) and propose a Dataset Aggregation (DAgger) strategy to mitigate it.

Over the years, researchers have sought to improve the basic BC and DAgger recipe in several ways [4, 27]. [19, 62, 64, 78, 85] devise ways to ease and scale-up expert data collection, while [84] design ways to collect demonstrations in virtual reality. [35, 36] devise ways to simplify DAgger data collection. [14, 61] model the multiple modes in expert demonstrations, while [49] employ a non-parametric approach. Researchers have also shown the effectiveness of pre-trained representations for imitation learning [49, 75]. [13, 46, 82, 86] employ image and view synthesis models to synthetically augment data. Complementary to our work, [11, 46, 82] focus on generalization across objects via *semantic* data augmentation by adding, deleting and editing objects in the scene.

[18, 32, 50, 86] pursue synthetic data augmentation to address the compounding execution error problem. [18, 32, 50] conduct these augmentation in low dimensional state spaces. [50] develops a principled way to sample these augmentations. Closest to our work, Zhou et al. [86]'s SPARTN model train NeRF's on demonstration data to synthesize out-of-distribution views. NeRF assumes a static scene. Thus, SPARTN can't reliably synthesize images when the scene undergoes deformation upon interaction of the robot with the environment. Our use of diffusion models to synthesize images gets around this issue and experimental comparison to SPARTN demonstrate the effectiveness of our design choice.

B. Image and View Synthesis Models

Recent years have witnessed large progress in image generation [20, 26, 54, 63, 65]. This has led to a number of applications: text-conditioned image generation [5, 22, 53, 54, 57, 83], image and video inpainting [10, 44, 47], image to image translation [28] and generation of novel views for objects and scenes from one or a few images [34, 42, 70, 73, 81]. While diffusion models have proven effective at image synthesis tasks, Neural Radiance Fields (NeRFs) [48] excel at view interpolation tasks. Given multiple (20 – 100) images of a static scene, vanilla NeRFs can effectively interpolate among the views to generate photo-realistic renders. Researchers have pursued NeRF extensions that enable view synthesis from few images [7, 21, 76, 80] and even model deforming scenes [9, 67]. In general, NeRFs are more effective at interpolation problems, while diffusion models excel at problems involving extrapolation (*e.g.* synthesizing content not seen at all) or where exactly modeling changes in 3D is hard (*e.g.* deforming scenes). As our work involves speculating how the scene would like if the gripper were at a different location, we adopt a diffusion based methods. We show comparison against a NeRF based method as well.

C. Diffusion Models and Neural Fields for Robot Learning

Effective generative models and neural field based representations have been used in robotics in other ways than for data augmentation. Researchers have used diffusion models for representing policies [1, 14, 37, 38, 72], generating goals or subgoals [6, 11, 31], offline reinforcement learning [3, 43, 72], planning [30, 40, 41, 74], behaviorally diverse policy generation [25], predicting affordances [77], and skill acquisition from task or play data [12]. Neural fields have been used to represent scenes for manipulation [39, 71] and navigation [2].

III. Approach

A. Overview

Given task data $\mathbf{D}$, imitation learning learns a task policy π. The task data comprises a set of trajectories τ_i, each consisting of a sequence of image action pairs, (I_t, a_t). Policy π is trained using supervised learning to regress action a_t from images I_t. In this paper, we consider manipulation of objects using an eye-in-hand camera, where input images I_t correspond to

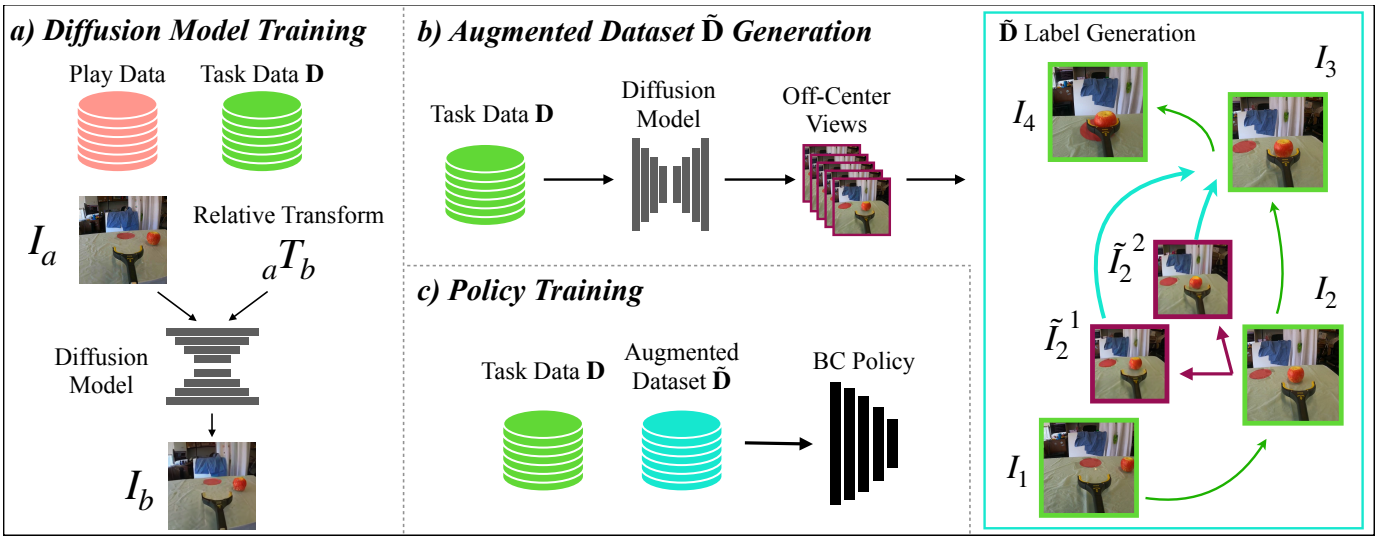

Fig. 2: DMD System Overview: Our system operates in three stages. **a)** A diffusion model is trained, using task and play data, to synthesize novel views relative to a given image. **b)** This diffusion model is used to generate an augmenting dataset that contains off-trajectory views ($\tilde{I}_2^1$, $\tilde{I}_2^2$) from expert demonstrations. Labels for these views (cyan arrows) are constructed such that off-trajectory views will still converge towards task success (right). Images with a green border are from trajectories in the original task dataset. Purple-outlined images are diffusion-generated augmenting samples. **c)** The original task data and augmenting dataset are combined for policy learning.

views from a wrist camera, and the actions a_t are the relative end-effector poses between consecutive time steps.

When trained with few demonstrations, π exhibits poor online performance upon encountering states not covered by the expert demonstrations. To address this issue, as shown in Figure 2, our approach generates an augmented dataset $\tilde{\mathbf{D}}$ and trains the policy jointly on $\tilde{\mathbf{D}} \cup \mathbf{D}$. Augmented samples are specifically generated to be out-of-distribution from $\mathbf{D}$, thus helping the policy to generalize. $\tilde{\mathbf{D}}$ is generated through a conditional diffusion model f that synthesizes a Δp-perturbed view $\tilde{I}_t$ of an image I_t from a given trajectory τ: $\tilde{I}_t = f(I_t, \Delta p)$. We use action labels in the trajectory τ to compute the action label $\tilde{a}_t$ for this perturbed view. The design of the conditional diffusion model is described in Section III-B, and the procedure for sampling augmenting images and computing action labels is detailed in Section III-C.

B. Diffusion Model for Synthesizing Δp Perturbed Views

The function $f(I_t, \Delta p)$ is realized using a conditional diffusion model that is pretrained on large Internet-scale data. Specifically, we adopt the recent work from Yu et al. [81] and finetune it on data from $\mathbf{D}$. As shown in Figure 2(a), the model produces image I_b by conditioning on a reference image I_a taken by camera a and a transformation matrix ${}_aT_b$ that maps points in camera b's frame to camera a's frame. By representing the desired Δp as the transformation between two cameras (${}_aT_b = \Delta p$), and using images from $\mathbf{D}$ as the reference images ($I_a \sim \mathbf{D}$), the learned model can generate the desired perturbed views for augmenting policy learning.

1) Model Architecture: A diffusion model is an iterative denoiser: given a noisy image x_t, diffusion timestep t, it is trained to predict the noise ϵ added to the image:

$$\mathcal{L} = \|\epsilon - \epsilon_\theta(x_t, t)\|_2^2.$$

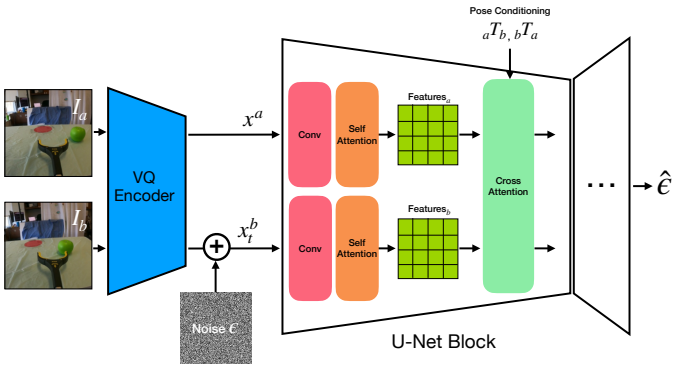

Fig. 3: DMD Architecture: We use the architecture introduced in [81], a U-Net diffusion model with blocks composed of convolution, self-attention, and cross attention layers. The conditioning image I_a, and noised target image I_b are processed in parallel except at cross-attention layers. The pose conditioning information is injected at cross-attention layers.

It is typically realized via a U-Net [55]. Our application requires *conditional image generation* where the conditions are source image I_a and transformation ${}_aT_b$. Thus, by using I_b as the diffusion target, and conditioning on I_a and ${}_aT_b$, the model ϵ_θ learns to denoise new views of the scene in I_a based on the transformation ${}_aT_b$. Finally, rather than directly denoising in the high-resolution pixel space, the denoising is done in the latent space of a VQ-GAN autoencoder, E [17, 54]. This gives the final training objective of:

$$\mathcal{L} = ||\epsilon - \epsilon_\theta(x_t^b, E(I_a), {}_aT_b, t)|| \quad \text{where} \quad x_0^b = E(I_b).$$

Following [81], the pose conditioning information ${}_aT_b$ is injected into the U-Net via cross-attention as shown in Figure 3.

2) Model Training: Training the model requires access to triples: $(I_a, I_b, {}_aT_b)$. We process data from $\mathbf{D}$ to generate these triples. Let each trajectory consist of images $I_1, \ldots, I_N$.

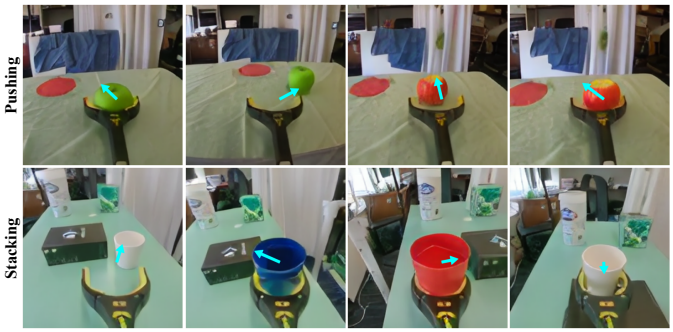

Fig. 4: Training Examples from the Diffusion Model and Computed Labels: We visualize generated examples, $\tilde{I}$, used to train our policies along with the computed action label (the arrow is a projection of the 3D action into the 2D image plane: the arrow pointing up means move the gripper forward, pointing to the right means move it right). The first row shows augmenting samples for the pushing task, while the second row shows those for the stacking task.

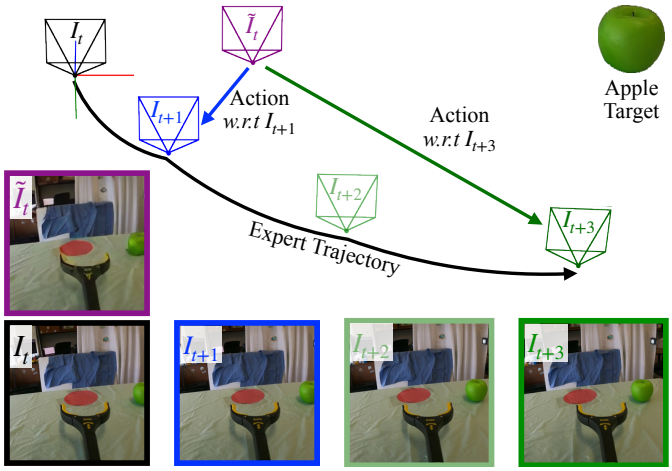

Fig. 5: The Overshooting Problem: When the generated image $\tilde{I}_t$ exceeds I_{t+1}, the inferred action for $\tilde{I}_t$ may direct the agent away from task success. We refer to this as the overshooting problem. At time step $t+1$, the view I_{t+1} has moved to the lower right of I_t. However, the synthesized sample $\tilde{I}_t$ has moved even further to the right than I_{t+1}, but not beyond I_{t+3}. Blue arrow represents action label for $\tilde{I}_t$ computed using I_{t+1} as the target; green arrow represents action label computed using I_{t+3} as the target. Since $\tilde{I}_t$ has *overshot* I_{t+1}, an action taken with I_{t+1} as the next intended target moves backward, away from the apple, and this labeling is not desirable. Computing the action with respect to a farther image, say I_{t+3}, does not have this issue.

We use structure from motion (SfM) algorithms [8, 15, 23, 59, 60, 69] to extract poses for the images in the trajectory τ. This associates each image I_t in the trajectory with the (arbitrary) world frame ${}_tT_w$. We can then compute relative pose between arbitrary images I_a and I_b from a trajectory via ${}_aT_b = {}_aT_w {}_wT_b$. We sample random pair of images from the trajectory to produce triples $(I_a, I_b, {}_aT_b)$ that are used to train the model ϵ_θ.

As $\mathbf{D}$ contains much fewer data than is typically required to train a diffusion models from scratch, we finetune the model from Yu et al. [81]. Finetuning with around 50 trajectories leads to realistic novel view synthesis for our tasks as shown in Figure 7. We only finetune the diffusion model weights. The pre-trained VQ-GAN codebooks is kept fixed.

The diffusion model can be finetuned on task data $\mathbf{D}$, task-agnostic play data [45, 79], or even combinations of task and play data. Play data in our setups involves moving the grabber within the workspace and randomly interacting with the task objects without emphasizing task completion. Access to random interactions in play data may make it easier for the diffusion model to synthesize diverse perturbations. We experiment with these different data sources for training the diffusion model (Section IV-A4d and Section IV-C2).

C. Generating Out-of-Distribution Images and Labels

1) Sampling Out-of-distribution Images: With the noise estimation model ϵ_θ trained, the image generation function $f(I_t, \Delta p)$ is realized by simply letting $I_a = I_t$, letting ${}_aT_b$ represent the desired Δp, and sampling from the conditional diffusion model using well-established sampling strategies [26, 54, 66].

We use $f(I_t, \Delta p)$ to generate out-of-distribution images with a very simple strategy for sampling Δp. For an image I_t from trajectory τ, we sample vectors in a random direction, with magnitudes drawn uniformly from a pre-defined range for each task. If the SfM reconstruction recovers the real-world scale, this range is set to $[2cm, 4cm]$; otherwise, it is $[0.2s, s]$, where s is the largest displacement between adjacent frames in τ. Δp simply corresponds to moving along this randomly chosen vector.

2) Labeling Generated Images: For each image $\tilde{I}_t = f(I_t, \Delta p)$ generated from an original image I_t, we use I_{t+k} as the target for generating the action label. The action label for $\tilde{I}_t$ is simply the action that conveys the agent from the pose depicted in $\tilde{I}_t$ to the pose in I_{t+k}.

Using SfM, we obtain camera pose information ${}_tT_w$ for time step t and ${}_{t+k}T_w$ for time step $t+k$. The synthesized view $\tilde{I}_t$ can be treated as an image taken by a virtual camera whose pose is represented as ${}_tT_{\tilde{t}}$. The diffusion model synthesizes perturbed view conditioned on I_t and ${}_tT_{\tilde{t}}$, or $\tilde{I}_t = f(I_t, {}_tT_{\tilde{t}})$ (Section III-B). The action label $\tilde{a}_t$ for $\tilde{I}_t$ is then computed as $\tilde{a}_t = {}_{\tilde{t}}T_{t+k} = {}_{\tilde{t}}T_t \, {}_tT_w \, ({}_{t+k}T_w)^{-1}$. Examples of $(\tilde{I}_t, \tilde{a}_t)$ are shown in Figure 4.

One might wonder why we don't just use $k = 1$? When sampling Δp, we assume that the perturbed observation $\tilde{I}_t$ is within a local region around I_t. However, some generated samples can move past the target image I_{t+1} (described in Figure 5). This causes conflicting supervision, as the computed action does not make progress toward completing the task. Using a large k guards against this, leading to stronger performance downstream, as our experiments verify (Section IV-A3b).

IV. Experiments

We test DMD across four tasks: non-prehensile pushing, stacking, pouring, and hanging a shirt. The task setups are shown in Figure 6. Together these tasks test DMD in different settings: 3DoF action space (pushing, stacking), 6DoF action

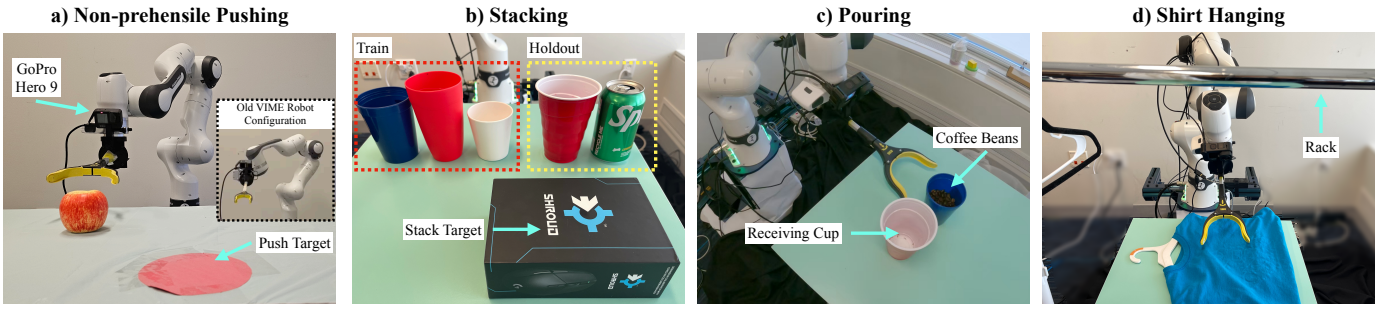

Fig. 6: DMD Robotic Experiments: There are four tasks in total: pushing apple to target location (Section IV-A), stacking five different cups on box (Section IV-B), pouring coffee beans into a cup (Section IV-C), and hanging shirt on a rack (Section IV-D). We conduct our experiments on a Franka Research 3 robot with a wrist-mounted GoPro Hero 9. We modified VIME's [78] grabber mount for Franka, allowing the robot to reach end-effector poses without reaching joint limits.

spaces (pouring, hanging a shirt), generalization to new objects (stacking), precision in reaching goal locations (pouring, hanging a shirt).

On the pushing task, we present visual comparisons to NeRF-based synthesis approach SPARTN [86] in Section IV-A2 and in-depth quantitative analysis (ablation of design choices, offline evaluations) in Section IV-A3. We then show a variety of online comparisons for the pushing task in Section IV-A4. Moreover, we compare DMD to behavior cloning for stacking (Section IV-B), pouring (Section IV-C), and hanging a shirt (Section IV-D). Finally, we test whether DMD improves generalization to novel objects and environment when provided with a diverse task dataset, as described in Section IV-E. We adopt randomized A/B testing when making comparisons between different methods to minimize the effects of unknown environmental factors.

A. *Non-prehensile Pushing*

1) Experiment Setup:

a) Task, Observation Space, Action Space: We use the non-prehensile pushing task from VIME [78], which involves reaching a target object and pushing it to the target location marked by a red circle. Following Young et al. [78], we use a grabber as the robot end-effector as shown in Figure 6. This allows for easy collection of expert demonstrations using a GoPro mounted on a grabber. Due to the difference in joint limits between the XArm (used by VIME) and the Franka Research 3 robot, we redesign the attachment between the grabber and the robot flange (Figure 6(a)). In this way, the robot can execute planar push in the typical top-down configuration, extending the workspace to a wider range of the table. Observations come from the eye-in-hand GoPro camera. For pushing, we use the action space used in VIME's [78] publicly available code: relative 3D translations of the end-effector.

b) Task and Play Data: We collect task and play data using a GoPro mounted on a grabber. The task data include 74 demonstration, and the play data include 36 trajectories. For task data, an expert pushes the object to a target location. For play data, we move the grabber around the apple and push it in different directions. From the GoPro videos, we obtain image sequences and use structure from motion [23, 69] to extract camera poses ${}_tT_w$. Action a_t is computed by computing the relative translation between I_{t+1} and I_t in I_t's camera frame. Since COLMAP only gives reconstructions up to an unknown scale factor, we normalize the action to unit length and interpret it as just a direction.

c) Policy Architecture and Execution: We adopt the architecture for the task policy π from VIME [78] but replace the AlexNet [33] backbone with an ImageNet pre-trained ResNet-18 backbone [24]. The policy consists of the ResNet backbone followed by 3 MLPs that predict the action. This task policy accepts a center cropped image from the GoPro camera and outputs the direction in which the camera should move. The policy is trained by L1 regression to the (unit length) actions using the Adam optimizer with a learning rate of 1e-4. Actions are executed on the robot by commanding the robot to go 1cm in the predicted direction.

d) Baselines: We use vanilla behavior cloning on the expert data as the baseline as done in past work [78], we refer to this as BC. We also compare to SPARTN [86] and evaluate the various design choices. Since there is no publicly available code from SPARTN, we replicate their procedure using NerfStudio [68].

2) Visual Comparison of Generated Images:

Figure 7 shows the high visual quality of samples generated by the diffusion model. The model is able to faithfully synthesize images where the camera is moved left / right, up / down, and front / back with minimal artifacts.

As shown in Figure 9, we also compare the quality of diffusion-generated images to those generated using the NeRF based approach from SPARTN [86]. The moving gripper breaks the static scene assumption made by NeRF. Thus, [86] masks out the gripper before training the NeRF. While this strategy works for the pre-grasp trajectory that Zhou et al. [86] seek to imitate, it fails when the gripper manipulates the scene, as in our tasks. Thus, we investigate different masking schemes: a) no masking, b) masking just the gripper, c) masking a larger region around the gripper; and show visualizations in Figure 9.

No masking leads to the worst results as expected. Masking just the grabber fixes the grabber, but causes the object being

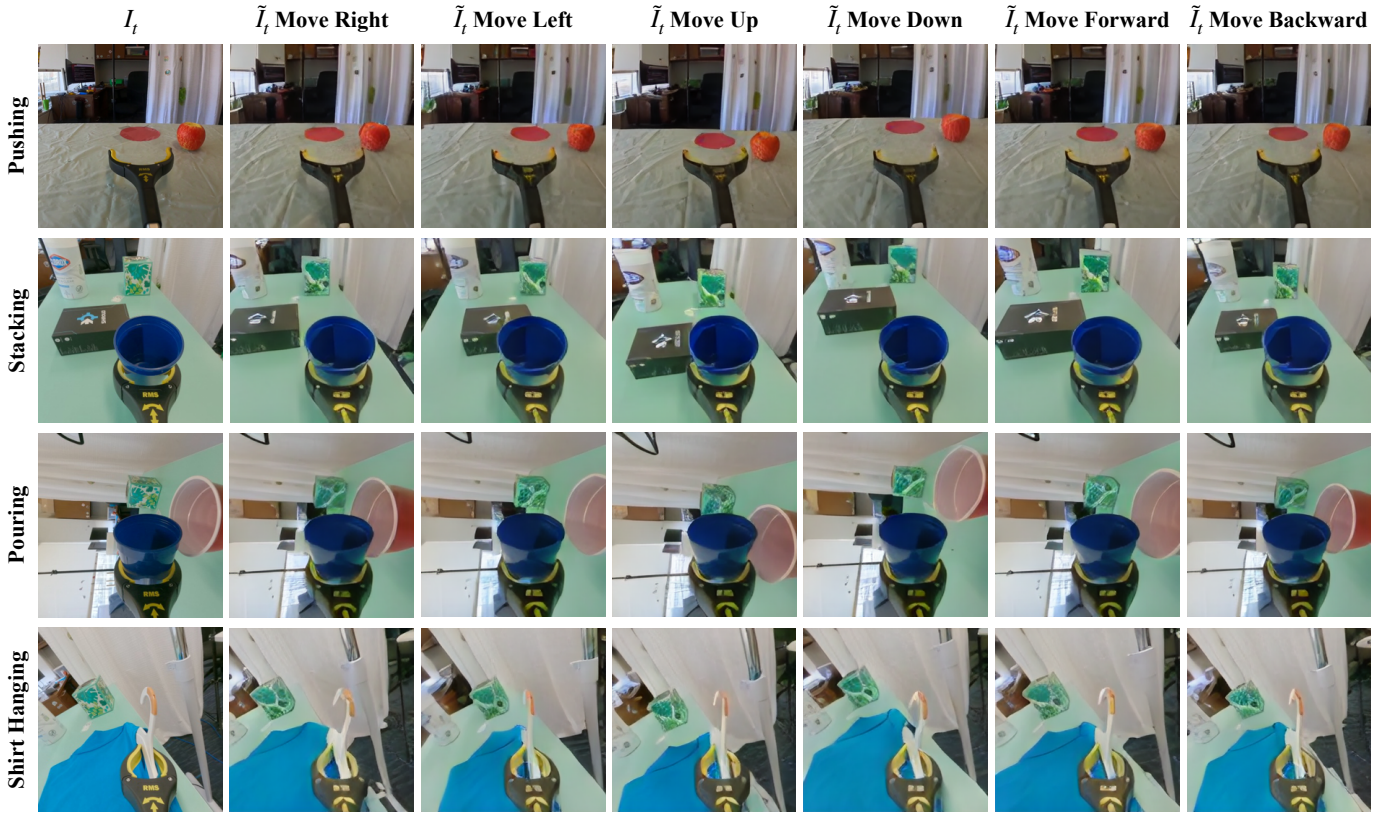

Fig. 7: Visualization of Perturbed Images Generated by Diffusion Model for Augmenting Policy Learning: In each row, we show $\tilde{I}_t$ generated from I_t through different camera translations for the tasks of pushing, stacking, pouring, and hanging a shirt.

TABLE I: Comparisons between Diffusion-generated Augmented Dataset and Standard Augmentation Schemes. The table shows median angle error (in radians) between predicted and ground-truth translations. Adding $\tilde{\mathbf{D}}$ generated by diffusion models improve performance on top of other augmentation techniques.

Methods	BC	DMD
No Aug	0.376	0.349
w/ Color jitter	0.381	0.350
w/ Flip	0.356	0.338
w/ Color jitter & Flip	0.355	**0.328**

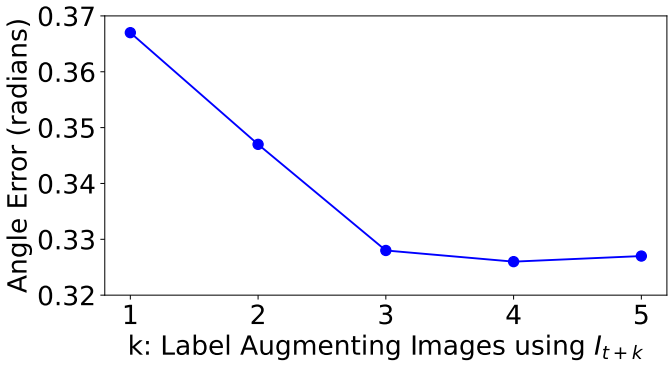

Fig. 8: Effect of Using Different Future Frames for Labeling Augmenting Images: We experiment with using different future frame I_{t+k} for labeling the diffusion-generated images. Error decreases as k increases and plateaus around $k = 3$. Therefore, we use frame I_{t+3} for labeling $\tilde{I}_t$

manipulated to blur out. Masking a larger region around the grabber fixes the blurring but also eliminates the object being manipulated. In contrast, images synthesized using our diffusion model move the camera as desired without creating any artifacts.

3) **Offline Validation:** Out of the 74 expert demonstrations, we use 24, 23, and 27 trajectories for train, validation, and test, respectively. As done in past work [78], the baseline is vanilla behavior cloning on the expert data. Our offline validation evaluates the policy's predictions on the test set using the median angle between the predicted and ground truth translations as the error metric.

a) Comparison with other data augmentation schemes.: The goal of this experiment is to compare the effectiveness of augmenting with diffusion-generated images to standard techniques, such as color jitter and horizontal flip. Table I shows that augmenting with out-of-distribution images leads to larger improvement compared to standard techniques. Comparing the two columns, we see that adding diffusion-generated images improve performance in all settings.

b) Overshooting problem, what frame to use?: In order to mitigate the overshooting problem, we use image $I_{t+k}, k > 1$ to label the augmenting image instead of I_{t+1}. Figure 8 shows the effect of this parameter k on policy performance. We find that labeling with I_{t+1}'s pose works worse than labeling with I_{t+3}'s pose and beyond. As the curve plateaus for $k > 3$

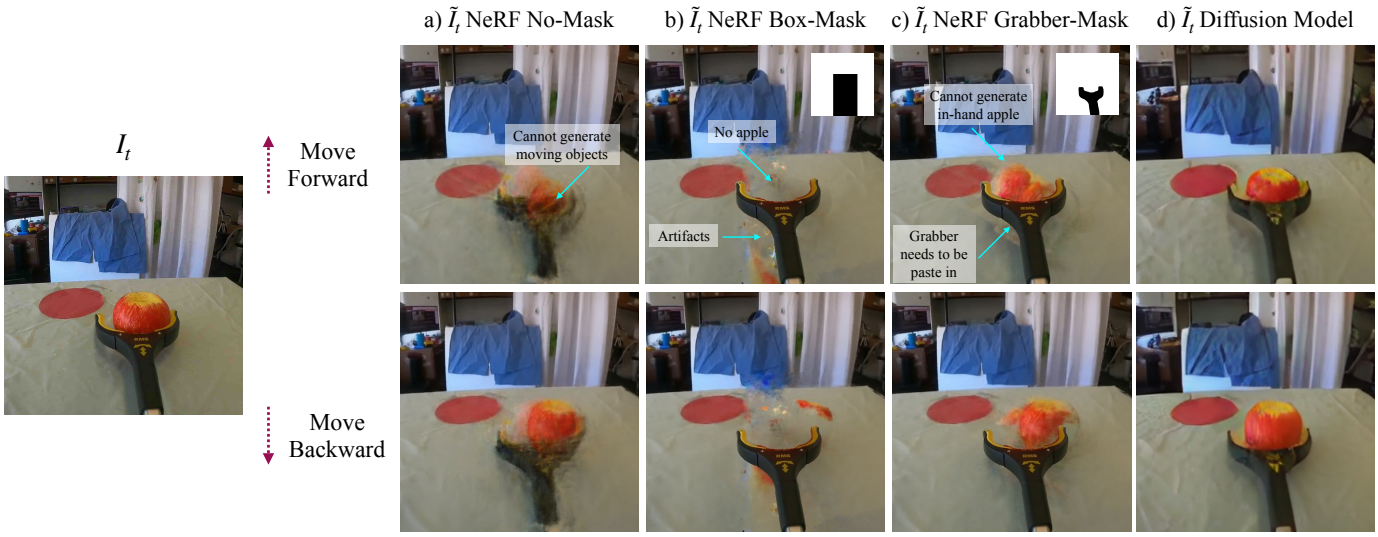

Fig. 9: Diffusion vs NeRF We visualize perturbed samples generated using DMD and NeRF with different masking strategies. The top row shows images generated for a forward movement relative to I_t; the bottom row shows images for a backward movement **(a)** With no masking, the NeRF reconstructions of the gripper and apple are degenerate, as they violate the static scene assumption in NeRFs. **(b)** With a box-mask, the apple is occluded by the mask, and consequently, missing from the reconstruction. **(c)** Even with a mask around the end-effector (Grabber-Mask), there are major artifacts in reconstructing the apple. **(d)** With ours (Diffusion Model), the model faithfully reconstructs the grabber and apple.

TABLE II: Pushing Online Evaluation Results. (a) DMD outperforms BC across all settings. DMD achieves a 100% success rate when pushing an apple, greatly exceeding BC's 30%. It also maintains an 80% success rate with only 8 demonstrations, whereas BC drops to 20%. (b) As shown in Figure 9, our diffusion model synthesizes higher quality images than NeRFs, especially when scenes undergo deformations. This advantage results in higher task performance: DMD achieves a 100% success rate, while SPARTN [86] achieves only 50%. (c) Training the diffusion model with additional play data boosts the task success rate to 100%, compared to 80% when using the model trained only on task data.

(a) DMD *vs.* BC

Method	Apple			White Cup
	24 Demo.	16 Demo.	8 Demo.	16 Demo.
BC	30%	50%	20%	80%
DMD	100%	90%	80%	90%

(b) DMD *vs.* SPARTN [86]

Method	Apple
	24 Demo.
SPARTN [86]	50%
DMD	100%

(c) Utility of Play Data

Method	Apple
DMD Task Only	80%
DMD Task & Play	100%

and using larger k leads to the exclusion of the last $k-1$ frames in each trajectory, we use $k = 3$ for our online experiments.

4) **Online Validation:** We conduct 10 trials per method and randomize the apple start location between trials. To prevent human biases, we choose the apple start location before sampling the method to run next. Execution trajectories can be found on the project website.

a) DMD vs. Behavior Cloning: We take the best model for BC and DMD from Table I and evaluate them on the robot. We find the vanilla BC model to get a 30% success rate while DMD achieves 100% (Table IIa). This significant difference shows that minor error at each step can accumulate and result in task failure.

b) DMD with Few Demonstrations: Motivated by the 100% performance of DMD with 24 demonstrations, we ask how low can the number of demonstrations be? We consider 2 low data settings with 8 and 16 demonstrations. For these settings, the diffusion model is trained with only play data. Success rate for these two models on the robot is shown in Table IIa. Once again DMD outperforms BC and achieves 80% success even when trained with only 8 demonstrations.

c) DMD vs. SPARTN [86]: Following up on the visual comparison of augmenting samples from Section IV-A2, we conduct a head-to-head comparison against SPARTN on the real robot. SPARTN achieves a success rate of 50% *vs.* DMD succeeds 100% of the time (Table IIb). In Figure 10, we show that while SPARTN diverges from the expert course and pushes the apple off the table, DMD brings the apple to the target successfully.

d) Utility of Play Data for Training Diffusion Model: We also experimentally validate what the choice of the dataset necessary for training the view-synthesis diffusion model. We repeat the experiment with 24 demos but vary the diffusion model that generated the augmenting samples. Using the diffusion model trained on just task data lead to an 80% success rate *vs.* 100% success rate with the diffusion model trained on a combination of task and play data (Table IIc). Thus, while just training the diffusion model on task data is effective (a nice result in practice), performance can be boosted further by training the diffusion model on play data.

e) Scaling to Other Objects: We test DMD on another object, a cup. The diffusion model is trained with only cup play data. Task data include 16 demonstrations. BC is trained

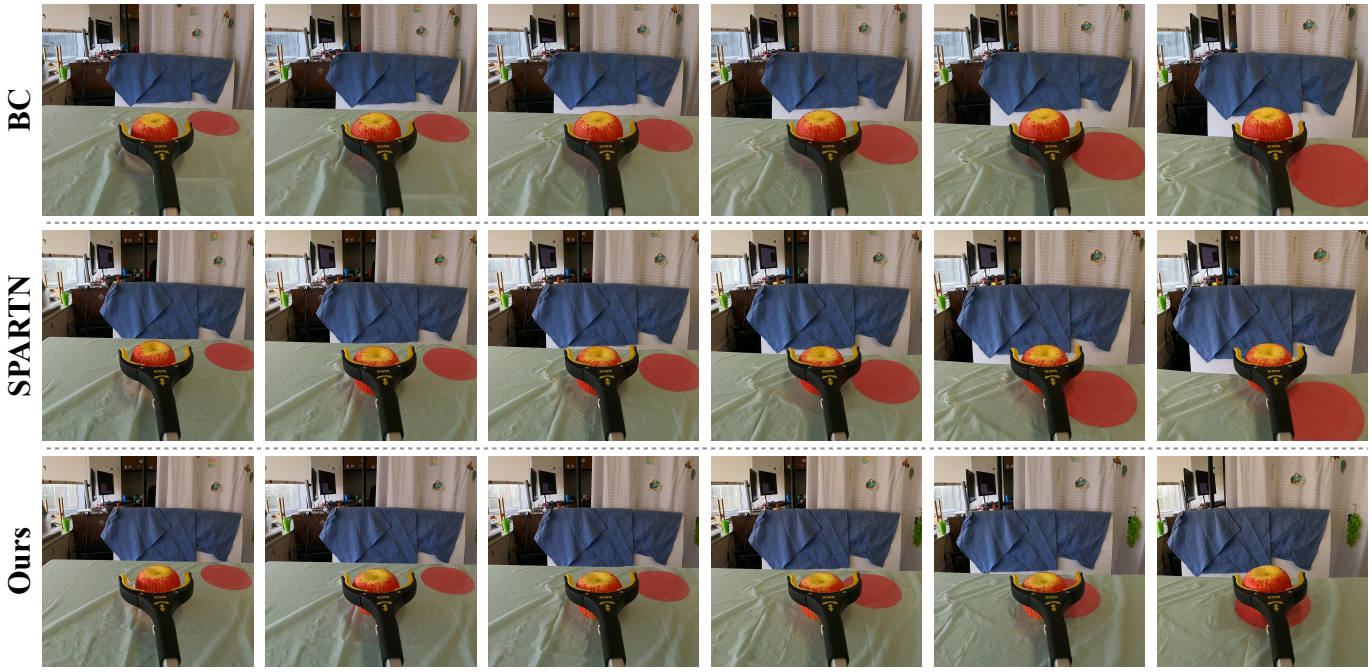

Fig. 10: Comparison of BC, SPARTN, and DMD for Staying on Course. We show the trajectories executed by BC, SPARTN [86], and DMD over several steps towards the target. Unlike BC and SPARTN, which gradually deviate from the intended path, DMD maintain its course towards the right and reach the target successfully. See project website for more execution videos.

only with task data, and DMD is trained additionally with synthesized images. Similar to apple, we repeat each method 10 times and randomize the cup start location. We choose the start location before sampling the method to execute. The BC baseline achieves a success rate of 80% vs DMD achieves a success rate of 90% (Table IIa).

B. *Stacking*

1) **Experiment Setup:** Stacking involves reaching a cup, grasping it, placing it on a black box, and opening the grabber. This task is similar to the stacking task in VIME [78] and requires precision in the height direction to move the cup on the box. The action space is the same as in VIME's publicly available code: relative 3D translations of the end-effector.

We use a grabber and collect 50 demonstrations across three training cups (red dotted box in Figure 6(b)). We obtain cameras poses using ORB-SLAM3 [8, 15], which recovers the correct metric scale by using the IMU data from GoPro.

We train the diffusion model on the 50 expert trajectories and don't use any play data. Unlike pushing, we do not need to scale by the largest displacement between adjacent frames because of metric scale from IMU. To sample out-of-distribution images, we sample a small translation (with length between $2cm$ and $4cm$) in a random direction and a small rotation obtained by sampling yaw, pitch, and roll uniformly from the range $[-10°, 10°]$, $[-10°, 10°]$, and $[-10°, 15°]$, respectively.

The policy architecture is the same as that of pushing and is trained with L1 loss to the normalized ground-truth translation.

TABLE III: Stacking Online Evaluation Results. In the task of stacking a cup on a box, DMD achieves over 90% success rate for all training cups and over 80% for the two cups unseen cups and outperforms BC. The diffusion model is only trained on task data/expert demonstrations. See all execution videos on project website.

	Seen Cups			Unseen Cups	
Method	**Blue**	**Red-1**	**White**	**Red-2**	**Sprite**
BC	10%	10%	90%	40%	50%
DMD (Task Only)	**100%**	**90%**	**100%**	**80%**	**90%**

We annotate frames at which the grabber open/close and train a gripper policy that is shared across all the methods.

2) **Online Validation:** We evaluate on the three training cups plus two holdout cups: a larger red cup and a Sprite can (yellow dotted box in Figure 6(b)). The starting locations of the objects are randomized and do not necessarily match the training data.

As shown in Table III, DMD performs better than BC across both seen and unseen instances. BC often fails to lift the tall cups above the box and pushes the box forward continuously. See videos on project website for failure modes. This behavior might explain the higher success rate of the white cup, which is shorter and thus easier to lift above the box. By training on synthesized data, DMD learns to lower the grabber if approaching the cups too high, and it grasps closer to the cup bottom so that moving above the box becomes easier. It succeeds more than BC across all cups.

TABLE IV: Pouring and Hanging-Shirt Online Evaluation Results. DMD outperforms BC across both tasks. See all execution videos on project website. *: the two BC numbers are different because they are from two different pairwise randomized A/B tests: 1) BC *vs.* DMD (Task & Play) and 2) BC *vs.* DMD (Task Only), and experimental conditions (*e.g.* lighting) may have been different between when these two tests were conducted.

	Pouring		Hanging
Method	**Task & Play**	**Task Only**	**Task & Play**
BC	0%*	30%*	20%
DMD	**80%**	**70%**	**90%**

C. *Pouring*

1) **Experiment Setup:** As shown in Figure 6(c), this task requires the robot to reach a blue cup filled with 57 grams of coffee beans, grasp it, and pour all the beans into a red cup. Success is determined by the weight of the coffee beans being the same at the end of each trial. Action space is full 6-DoF.

Task data contain 49 demonstrations, and play data contain 16 trajectories. Diffusion model is trained on both types of data. Except for an additional MLP-head to predict rotation, the policy architecture is the same as that of stacking; the model is trained with L1 loss to the ground-truth translation and rotation.

During execution, the robot executes the translation and rotation predicted by the model. Locations of the two cups are randomized, but the blue cup is always on the same side of the red cup.

2) **Online Validation:** DMD succeeds 80% of the time while BC fails completely. A common failure case for BC is that as the robot rotates the cup with coffee beans, it does not move the cup closer to the receiving cup; the blue cup then drifts further and further from the one on the table. In contrast, even in the worst trial, DMD can pour 53 grams (*i.e.* more than 90%) of the coffee beans. See execution videos on project website for clear differences between BC and DMD.

We also validate the use of play data for learning an effective diffusion model. We finetune the diffusion model with only task data and conduct the same experiment as the one in the last paragraph. As shown in Table IV, without training the diffusion model with play data, the downstream policy maintains a 70% success rate, higher than BC's 30%. This shows that, training on task data alone also enables the diffusion model to synthesize good out-of-distribution images that improve task performance. Note that the two BC numbers are different because they are from two different pairwise randomized A/B tests: 1) BC *vs.* DMD (Task & Play) and 2) BC *vs.* DMD (Task Only), and experimental conditions (*e.g.* lighting) may have been different between when these tests were conducted.

D. *Hanging a Shirt*

1) **Experiment Setup:** To hang the shirt, the robot reaches the hanger and slides the shirt across the table to the edge; it then grasps the part of the hanger sticking out of the table and hangs the shirt on the clothing rack. The setup is shown in Figure 6(d). Action space is full 6-DoF.

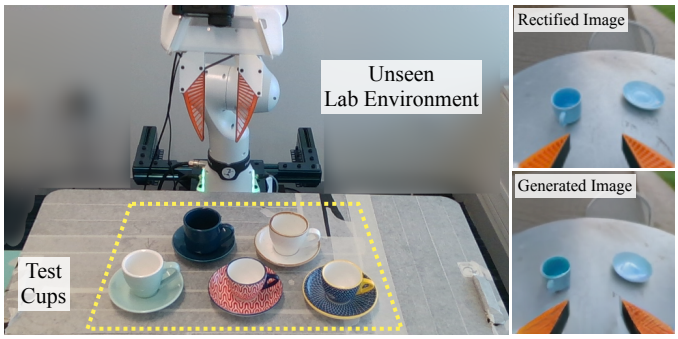

Fig. 11: In-the-Wild Cup Arrangement Setup: We evaluate DMD on the in-the-wild cup arrangement task from UMI [15]. The training set for this task contains 1447 demonstrations across 30 locations and 18 cups. Because our pre-trained diffusion model operates on non-fish eye images, we work with center-cropped and rectified images from the Fisheye lens. Baseline and baseline+DMD operate on these center-cropped rectified images. Evaluation is done on 5 novel cups in a novel environment (our lab).

TABLE V: In-the-Wild Cup Arrangement Online Evaluation Results. DMD generalizes better to novel cups in novel environment than vanilla Diffusion Policy [14], achieving a 64% overall success rate compared to Diffusion Policy's 16%. It succeeds in same or more trials across all 5 cups. See all execution videos on project website.

Method	**Navy**	**White**	**Mint**	**Red**	**Blue**	**Overall**
Diffusion Policy [14]	1/5	1/5	1/5	0/5	1/5	4/25 (16%)
DMD (Task Only)	**3/5**	**3/5**	**3/5**	**3/5**	**4/5**	**16/25 (64%)**

Task data contains 55 demonstrations, and play data contains 24 trajectories. The diffusion model is trained on both types of data. We divide the hanging task into two sub-tasks: grabbing the shirt and putting it on the rack. A separate policy is learned for each task. We use same policy architecture and loss function as those used in pouring.

During execution, the robot starts with the grabbing policy. When the network predicts to close the grabber, the grabbing policy terminates, and the robot switches to the putting-on-rack policy. The entire process terminates when the top of the hanger makes contact with the rack pole, signaled by when the end-effector senses a force greater than $4N$. We add some clay padding to the grabber tips because the hanger is thin, and the motor is not strong enough to close the grabber completely.

2) **Online Validation:** As shown in Table IV, DMD achieves a 90% success rate while BC only succeed in 20% cases. BC often fails because the robot raises the shirt too high, and the shoulder part of the hanger hits the pole before the hook part reaches the pole. On the other hand, DMD learns to bring the shirt down onto the rack.

E. *In-the-Wild Cup Arrangement*

The goal of this experiment is to understand a) can DMD improve performance in the presence of a large number of diverse demonstrations, b) can DMD improve generalization of policies to novel objects in novel environments, and c) if DMD is compatible with policies trained with diffusion [14].

1) **Experiment Setup:** We leverage a diverse in-the-wild dataset from the recent Universal Manipulation Interface (UMI) paper [15]. It introduces a hand-held data collection

device and an imitation learning framework utilizing Diffusion Policy [14]. The device captures image observations using a wrist-mounted GoPro camera.

We adopt the same task definition as the in-the-wild generalization experiment in UMI: placing a cup on a saucer with its handle facing the left side of the robot. UMI [15] collected 1447 demonstrations across 30 locations and 18 training cups. We use their publicly available demonstration data and conduct evaluation in our lab (*i.e.* novel location) with and without DMD.

Our evaluation setup is identical to that used in UMI, with one difference. UMI uses wide field-of-view images obtained by putting an additional Fisheye lens on the GoPro. Our diffusion model denoises in the latent space of a VQ-GAN autoencoder [16] that was pre-trained on the RealEstate10K dataset [87], which contains non fish-eye images. Therefore, our diffusion model is constrained to only work well on non-fish eye images. Thus, we conduct comparisons where both the baseline and baseline+DMD operate on center-cropped rectified images shown in Figure 11. We retrain the baseline diffusion policy from UMI to consume these center-cropped rectified images. Other than this change, the task policy uses UMI's implementation of Diffusion Policy [14] and denoises 6DoF actions.

We test on 5 held-out cups shown in Figure 11. These cups are not used for expert demonstrations in dataset from [15]. For each cup, we test 5 different start configurations. We follow the experiment protocol outlined in [15]: we use pixel masks to make sure that the starting locations of the cups and saucers are the same across the two methods.

2) **Online Validation:** DMD improves upon the vanilla Diffusion Policy [14] across all novel cups, achieving an overall success rate of 64%, compared to Diffusion Policy's 16% success rate (Table V). This improvement shows that DMD is effective even in the presence of large number (1447) of diverse demonstrations and also leads to improved performance when testing on novel objects in novel environments. Furthermore, DMD-created data also improves policies trained via diffusion. One common failure case for the baseline is that the policy doesn't reorient the cup correctly, because it fails to make contact with the cup or its handle. DMD, on the other hand, successfully pushes the handle to the left side more often and grasps the cup without tipping it over.

V. Discussion

We developed DMD, a data creation framework that improves the sample efficiency of eye-in-hand imitation learning. DMD uses a diffusion model to synthesize out-of-distribution views and assigns them corrective labels. Additional use of these out-of-distribution views for policy training leads to more performant policies. DMD leads to large benefits over traditional behavior cloning across 4 diverse tasks: pushing, stacking, pouring, and hanging a shirt. DMD attains 80% success rate from as few as 8 demonstrations in pushing and reaches an average of 92% success rate across 5 different cups in stacking. When pouring, it transfers coffee beans successfully 80% of the time; finally, it can slide a shirt off the table and put it on the rack with 90% success rate. Together these tasks test DMD on different aspects: 3DoF action space (pushing, stacking), 6DoF action spaces (pouring, hanging a shirt), generalization to new objects (stacking), and precision in reaching goal locations (pouring, hanging a shirt).

Our experiments reveal the benefits of the various parts of DMD over past approaches. The use of diffusion models to synthesize images allows DMD to generate augmenting images for manipulation tasks involving a non-static scene. Prior work [86] used NeRFs instead of diffusion models, restricting generation to samples from only the pre-grasp part of the demonstration. We showcase the advantages of diffusion models over NeRFs through qualitative results in Figure 9 and real robot experiments in Section IV-A4c.

We also tested which type of data: task data, play data, or a combination of both, should be used for training the diffusion model. We found all versions to outperform behavior cloning (Section IV-A4d, Table III, and Table IV). This suggests two practically useful results. First, DMD can be directly used to improve a task policy, *i.e.* the task data itself can be used to finetune a novel-view synthesis diffusion model, augmenting the task dataset to improve performance. Second, we can utilize play data, which may be simpler to collect than task data.

DMD also has several limitations. DMD assumes that the generated states are recoverable, *i.e.* there exists an action that can return to the expert distribution. This is not always true, for example, when the robot rotates the cup at the wrong location and the coffee beans spill. However, DMD can still be useful if it can bring the system back to in-distribution states before the system spirals into irrecoverable state.

Possible future directions include: learning diffusion models that can predict highly discontinuous object dynamics, generating out-of-distribution states of non-visual modalities (*e.g.* forces), and finding ways to assign action labels in situations where they cannot be derived from relative camera transformation Code, data and models are publicly available on the project website.

Acknowledgments

We give special thanks to Kevin Zhang for 3D printing the attachment for us. This material is based upon work supported by the USDA-NIFA AIFARMS National AI Institute (USDA #2020-67021-32799) and NSF (IIS-2007035).

References

[1] Suzan Ece Ada, Erhan Oztop, and Emre Ugur. Diffusion policies for out-of-distribution generalization in offline reinforcement learning. *IEEE Robotics and Automation Letters*, 2024.

[2] Michal Adamkiewicz, Timothy Chen, Adam Caccavale, Rachel Gardner, Preston Culbertson, Jeannette Bohg, and Mac Schwager. Vision-only robot navigation in a neural radiance world. *IEEE Robotics and Automation Letters*, 7(2):4606–4613, 2022.

[3] Anurag Ajay, Yilun Du, Abhi Gupta, Joshua Tenenbaum, Tommi Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision-making? *arXiv preprint arXiv:2211.15657*, 2022.

[4] Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and autonomous systems*, 57 (5):469–483, 2009.

[5] Yogesh Balaji, Seungjun Nah, Xun Huang, Arash Vahdat, Jiaming Song, Karsten Kreis, Miika Aittala, Timo Aila, Samuli Laine, Bryan Catanzaro, et al. ediffi: Text-to-image diffusion models with an ensemble of expert denoisers. *arXiv preprint arXiv:2211.01324*, 2022.

[6] Kevin Black, Mitsuhiko Nakamoto, Pranav Atreya, Homer Walke, Chelsea Finn, Aviral Kumar, and Sergey Levine. Zero-shot robotic manipulation with pre-trained image-editing diffusion models. *arXiv preprint arXiv:2310.10639*, 2023.

[7] Valts Blukis, Taeyeop Lee, Jonathan Tremblay, Bowen Wen, In So Kweon, Kuk-Jin Yoon, Dieter Fox, and Stan Birchfield. Neural fields for robotic object manipulation from a single image. *arXiv preprint arXiv:2210.12126*, 2022.

[8] Carlos Campos, Richard Elvira, Juan J. Gómez, José M. M. Montiel, and Juan D. Tardós. ORB-SLAM3: An accurate open-source library for visual, visual-inertial and multi-map SLAM. *IEEE Transactions on Robotics*, 37(6):1874–1890, 2021.

[9] Ang Cao and Justin Johnson. Hexplane: A fast representation for dynamic scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 130–141, 2023.

[10] Matthew Chang, Aditya Prakash, and Saurabh Gupta. Look ma, no hands! agent-environment factorization of egocentric videos. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

[11] Chang Chen, Fei Deng, Kenji Kawaguchi, Caglar Gulcehre, and Sungjin Ahn. Simple hierarchical planning with diffusion. *arXiv preprint arXiv:2401.02644*, 2024.

[12] Lili Chen, Shikhar Bahl, and Deepak Pathak. Playfusion: Skill acquisition via diffusion from language-annotated play. In *Conference on Robot Learning*, pages 2012–2029. PMLR, 2023.

[13] Zoey Chen, Sho Kiami, Abhishek Gupta, and Vikash Kumar. Genaug: Retargeting behaviors to unseen situations via generative augmentation. *arXiv preprint arXiv:2302.06671*, 2023.

[14] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Robotics: Science and Systems (RSS)*, 2023.

[15] Cheng Chi, Zhenjia Xu, Chuer Pan, Eric Cousineau, Benjamin Burchfiel, Siyuan Feng, Russ Tedrake, and Shuran Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. *arXiv preprint arXiv:2402.10329*, 2024.

[16] Patrick Esser, Robin Rombach, and Björn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12868–12878, 2021.

[17] Patrick Esser, Robin Rombach, and Björn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[18] Peter Florence, Lucas Manuelli, and Russ Tedrake. Self-supervised correspondence in visuomotor policy learning. *IEEE Robotics and Automation Letters*, 5(2):492–499, 2019.

[19] Alessandro Giusti, Jérôme Guzzi, Dan C. Cireşan, Fang-Lin He, Juan P. Rodríguez, Flavio Fontana, Matthias Faessler, Christian Forster, Jürgen Schmidhuber, Gianni Di Caro, Davide Scaramuzza, and Luca M. Gambardella. A machine learning approach to visual perception of forest trails for mobile robots. *IEEE Robotics and Automation Letters*, 1(2):661–667, 2016. doi: 10.1109/LRA.2015.2509024.

[20] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 27, 2014.

[21] Jiatao Gu, Alex Trevithick, Kai-En Lin, Joshua M Susskind, Christian Theobalt, Lingjie Liu, and Ravi Ramamoorthi. Nerfdiff: Single-image view synthesis with nerf-guided distillation from 3d-aware diffusion. In *International Conference on Machine Learning*, pages 11808–11826. PMLR, 2023.

[22] Shuyang Gu, Dong Chen, Jianmin Bao, Fang Wen, Bo Zhang, Dongdong Chen, Lu Yuan, and Baining Guo. Vector quantized diffusion model for text-to-image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10696–10706, 2022.

[23] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[24] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[25] Shashank Hegde, Sumeet Batra, KR Zentner, and Gaurav S Sukhatme. Generating behaviorally diverse policies with latent diffusion models. *arXiv preprint arXiv:2305.18738*, 2023.

[26] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

[27] Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, and Chrisina Jayne. Imitation learning: A survey of learning methods. *ACM Computing Surveys (CSUR)*, 50 (2):1–35, 2017.

[28] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1125–1134, 2017.

[29] Eric Jang, Alex Irpan, Mohi Khansari, Daniel Kappler, Frederik Ebert, Corey Lynch, Sergey Levine, and Chelsea Finn. BC-Z: Zero-shot task generalization with robotic imitation learning. In *Proceedings of the Conference on Robot Learning (CoRL)*, pages 991–1002. PMLR, 2022.

[30] Michael Janner, Yilun Du, Joshua B Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991*, 2022.

[31] Ivan Kapelyukh, Vitalis Vosylius, and Edward Johns. Dall-e-bot: Introducing web-scale diffusion models to robotics. *IEEE Robotics and Automation Letters*, 2023.

[32] Liyiming Ke, Jingqiang Wang, Tapomayukh Bhattacharjee, Byron Boots, and Siddhartha Srinivasa. Grasping with chopsticks: Combating covariate shift in model-free imitation learning for fine manipulation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 6185–6191. IEEE, 2021.

[33] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, page 1097–1105, 2012.

[34] Jeong-gi Kwak, Erqun Dong, Yuhe Jin, Hanseok Ko, Shweta Mahajan, and Kwang Moo Yi. Vivid-1-to-3: Novel view synthesis with video diffusion models. *arXiv preprint arXiv:2312.01305*, 2023.

[35] Michael Laskey, Sam Staszak, Wesley Yu-Shu Hsieh, Jeffrey Mahler, Florian T Pokorny, Anca D Dragan, and Ken Goldberg. Shiv: Reducing supervisor burden in dagger using support vectors for efficient learning from demonstrations in high dimensional state spaces. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 462–469. IEEE, 2016.

[36] Michael Laskey, Jonathan Lee, Roy Fox, Anca Dragan, and Ken Goldberg. Dart: Noise injection for robust imitation learning. In *Proceedings of the Conference on Robot Learning (CoRL)*, pages 143–156. PMLR, 2017.

[37] Wenhao Li, Xiangfeng Wang, Bo Jin, and Hongyuan Zha. Hierarchical diffusion for offline decision making. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 20035–20064. PMLR, 2023.

[38] Xiang Li, Varun Belagali, Jinghuan Shang, and Michael S Ryoo. Crossway diffusion: Improving diffusion-based visuomotor policy via self-supervised learning. *arXiv preprint arXiv:2307.01849*, 2023.

[39] Yunzhu Li, Shuang Li, Vincent Sitzmann, Pulkit Agrawal, and Antonio Torralba. 3d neural scene representations for visuomotor control. In *Conference on Robot Learning*, pages 112–123. PMLR, 2022.

[40] Zhixuan Liang, Yao Mu, Mingyu Ding, Fei Ni, Masayoshi Tomizuka, and Ping Luo. Adaptdiffuser: Diffusion models as adaptive self-evolving planners. *arXiv preprint arXiv:2302.01877*, 2023.

[41] Zhixuan Liang, Yao Mu, Hengbo Ma, Masayoshi Tomizuka, Mingyu Ding, and Ping Luo. Skilldiffuser: Interpretable hierarchical planning via skill abstractions in diffusion-based task execution. *arXiv preprint arXiv:2312.11598*, 2023.

[42] Ruoshi Liu, Rundi Wu, Basile Van Hoorick, Pavel Tokmakov, Sergey Zakharov, and Carl Vondrick. Zero-1-to-3: Zero-shot one image to 3d object. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9298–9309, 2023.

[43] Cong Lu, Philip J Ball, and Jack Parker-Holder. Synthetic experience replay. *arXiv preprint arXiv:2303.06614*, 2023.

[44] Andreas Lugmayr, Martin Danelljan, Andres Romero, Fisher Yu, Radu Timofte, and Luc Van Gool. Repaint: Inpainting using denoising diffusion probabilistic models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11461–11471, 2022.

[45] Corey Lynch, Mohi Khansari, Ted Xiao, Vikash Kumar, Jonathan Tompson, Sergey Levine, and Pierre Sermanet. Learning latent plans from play. In *Conference on robot learning*, pages 1113–1132. PMLR, 2020.

[46] Zhao Mandi, Homanga Bharadhwaj, Vincent Moens, Shuran Song, Aravind Rajeswaran, and Vikash Kumar. Cacti: A framework for scalable multi-task multi-scene visual imitation learning. *arXiv preprint arXiv:2212.05711*, 2022.

[47] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. Sdedit: Guided image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021.

[48] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021.

[49] Jyothish Pari, Nur Muhammad, Sridhar Pandian Arunachalam, and Lerrel Pinto. The surprising effectiveness of representation learning for visual imitation. *arXiv preprint arXiv:2112.01511*, 2021.

[50] Jung Yeon Park and Lawson L. S. Wong. Robust imitation of a few demonstrations with a backwards model. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2022.

[51] Dean A Pomerleau. Alvinn: An autonomous land vehicle in a neural network. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 1, 1988.

[52] Dean A Pomerleau. Efficient training of artificial neural networks for autonomous navigation. *Neural computation*, 3(1):88–97, 1991.

[53] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey

Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1(2):3, 2022.

[54] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[55] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI*, pages 234–241. Springer, 2015.

[56] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635, 2011.

[57] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems*, 35:36479–36494, 2022.

[58] Stefan Schaal. Learning from demonstration. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 9, 1996.

[59] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[60] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *European Conference on Computer Vision (ECCV)*, 2016.

[61] Nur Muhammad Shafiullah, Zichen Cui, Ariuntuya Arty Altanzaya, and Lerrel Pinto. Behavior transformers: Cloning k modes with one stone. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 35, pages 22955–22968, 2022.

[62] Nur Muhammad Mahi Shafiullah, Anant Rai, Haritheja Etukuru, Yiqian Liu, Ishan Misra, Soumith Chintala, and Lerrel Pinto. On bringing robots home. *arXiv preprint arXiv:2311.16098*, 2023.

[63] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015.

[64] Shuran Song, Andy Zeng, Johnny Lee, and Thomas Funkhouser. Grasping in the wild: Learning 6dof closed-loop grasping from low-cost demonstrations. *Robotics and Automation Letters*, 2020.

[65] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, 2019.

[66] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[67] Shih-Yang Su, Frank Yu, Michael Zollhöfer, and Helge Rhodin. A-nerf: Articulated neural radiance fields for learning human shape, appearance, and pose. volume 34, pages 12278–12291, 2021.

[68] Matthew Tancik, Ethan Weber, Evonne Ng, Ruilong Li, Brent Yi, Terrance Wang, Alexander Kristoffersen, Jake Austin, Kamyar Salahi, Abhik Ahuja, et al. Nerfstudio: A modular framework for neural radiance field development. In *Proceedings of the ACM SIGGRAPH Conference*, pages 1–12, 2023.

[69] Sebastian Thrun. Probabilistic robotics. *Communications of the ACM*, 45(3):52–57, 2002.

[70] Hung-Yu Tseng, Qinbo Li, Changil Kim, Suhib Alsisan, Jia-Bin Huang, and Johannes Kopf. Consistent view synthesis with pose-guided diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16773–16783, 2023.

[71] Yixuan Wang, Zhuoran Li, Mingtong Zhang, Katherine Driggs-Campbell, Jiajun Wu, Li Fei-Fei, and Yunzhu Li. d^3 fields: Dynamic 3d descriptor fields for zero-shot generalizable robotic manipulation. *arXiv preprint arXiv:2309.16118*, 2023.

[72] Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. *arXiv preprint arXiv:2208.06193*, 2022.

[73] Daniel Watson, William Chan, Ricardo Martin-Brualla, Jonathan Ho, Andrea Tagliasacchi, and Mohammad Norouzi. Novel view synthesis with diffusion models. *arXiv preprint arXiv:2210.04628*, 2022.

[74] Zhou Xian, Nikolaos Gkanatsios, Theophile Gervet, Tsung-Wei Ke, and Katerina Fragkiadaki. Chaineddiffuser: Unifying trajectory diffusion and keypose prediction for robotic manipulation. In *Conference on Robot Learning*, pages 2323–2339. PMLR, 2023.

[75] Tete Xiao, Ilija Radosavovic, Trevor Darrell, and Jitendra Malik. Masked visual pre-training for motor control. *arXiv preprint arXiv:2203.06173*, 2022.

[76] Jiawei Yang, Marco Pavone, and Yue Wang. Freenerf: Improving few-shot neural rendering with free frequency regularization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8254–8263, 2023.

[77] Yufei Ye, Xueting Li, Abhinav Gupta, Shalini De Mello, Stan Birchfield, Jiaming Song, Shubham Tulsiani, and Sifei Liu. Affordance diffusion: Synthesizing hand-object interactions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 22479–22489, 2023.

[78] Sarah Young, Dhiraj Gandhi, Shubham Tulsiani, Abhinav Gupta, Pieter Abbeel, and Lerrel Pinto. Visual imitation made easy. In *Proceedings of the Conference on Robot Learning (CoRL)*, pages 1992–2005. PMLR, 2021.

[79] Sarah Young, Jyothish Pari, Pieter Abbeel, and Lerrel Pinto. Playful interactions for representation learning. In *International Conference on Intelligent Robots and Systems*, pages 992–999. IEEE, 2022.

[80] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4578–4587, 2021.

[81] Jason J Yu, Fereshteh Forghani, Konstantinos G Derpanis, and Marcus A Brubaker. Long-term photometric consistent novel view synthesis with diffusion models. *arXiv preprint arXiv:2304.10700*, 2023.

[82] Tianhe Yu, Ted Xiao, Austin Stone, Jonathan Tompson, Anthony Brohan, Su Wang, Jaspiar Singh, Clayton Tan, Jodilyn Peralta, Brian Ichter, et al. Scaling robot learning with semantically imagined experience. *arXiv preprint arXiv:2302.11550*, 2023.

[83] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023.

[84] Tianhao Zhang, Zoe McCarthy, Owen Jow, Dennis Lee, Xi Chen, Ken Goldberg, and Pieter Abbeel. Deep imitation learning for complex manipulation tasks from virtual reality teleoperation. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 5628–5635, 2018.

[85] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

[86] Allan Zhou, Moo Jin Kim, Lirui Wang, Pete Florence, and Chelsea Finn. Nerf in the palm of your hand: Corrective augmentation for robotics via novel-view synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17907–17917, 2023.

[87] Tinghui Zhou, Richard Tucker, John Flynn, Graham Fyffe, and Noah Snavely. Stereo magnification: learning view synthesis using multiplane images. *ACM Trans. Graph.*, 37(4), 2018.

[88] Yuke Zhu, Ziyu Wang, Josh Merel, Andrei Rusu, Tom Erez, Serkan Cabi, Saran Tunyasuvunakool, János Kramár, Raia Hadsell, Nando de Freitas, et al. Reinforcement and imitation learning for diverse visuomotor skills. *arXiv preprint arXiv:1802.09564*, 2018.