DemoDiffusion: One-Shot Human Imitation using pre-trained Diffusion Policy

Author Names Omitted for Anonymous Review.

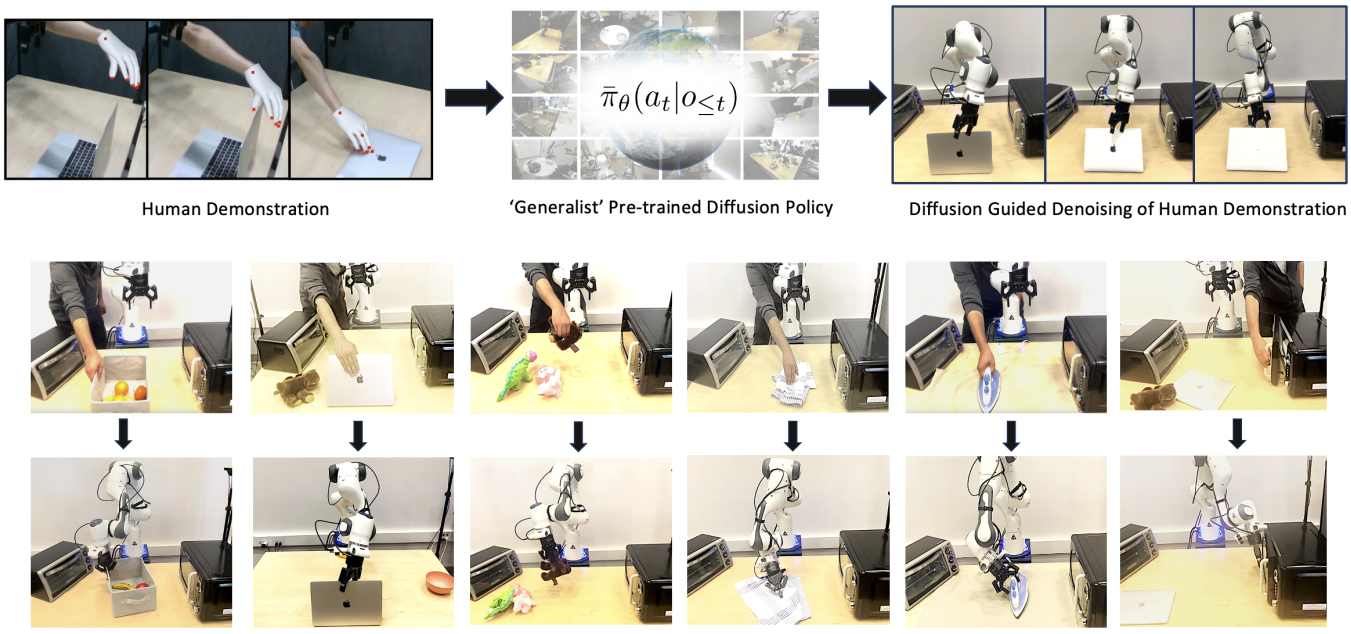

Fig. 1: **Overview of *DemoDiffusion*.** We show how *generalist* pre-trained diffusion policies can be used for following a generic human demonstration showing a manipulation task during deployment. Our real-world manipulation results encompass a wide diversity of manipulation tasks involving everyday objects.

***Abstract*—We propose *DemoDiffusion*, a simple and scalable method for enabling robots to perform manipulation tasks in natural environments by imitating a single human demonstration. Our approach is based on two key insights. First, the hand motion in a human demonstration provides a useful prior for the robot's end-effector trajectory, which we can convert into a rough open-loop robot motion trajectory via kinematic re-targeting. Second, while this re-targeted motion captures the overall structure of the task, it may not align well with plausible robot actions in-context. To address this, we leverage a pre-trained *generalist* diffusion policy to modify the trajectory, ensuring it both follows the human motion and remains within the distribution of plausible robot actions. Our approach avoids the need for online reinforcement learning or paired human-robot data, enabling robust adaptation to new tasks and scenes with minimal manual effort. Experiments in both simulation and real-world settings show that *DemoDiffusion* outperforms both the base policy and the re-targeted trajectory, enabling the robot to succeed even on tasks where the pre-trained *generalist* policy fails entirely. Project page: https://demodiffusion-anonymous.github.io/**

I. Introduction

How do we build robot manipulation systems that can be readily deployed in unstructured human environments? One possible answer is to learn 'generalist' policies that are capable of accomplishing any generic task (specified via some description language or image goal) in any environment. Indeed, there is a general optimism about this paradigm, reflected in several ongoing efforts to collect large-scale demonstration datasets to train such policies. While these efforts have led to impressive results in the form of unified policies capable of performing diverse tasks [1, 2, 3, 4, 5], these policies still struggle to perform meaningfully when deployed zero-shot to novel environments or asked to perform unseen tasks. Deployment thus often requires additional fine-tuning using task-and-scene-specific robot demonstrations [1], but this is a non-trivial overhead as collecting robot demonstrations in the real world can be time-consuming and beyond the expertise of a average user. In this work, we propose an alternate deployment mechanism – leveraging a generalist policy to perform a task via imitating a single *human* demonstration.

We are of course not the first to consider this goal of allowing robots to imitate any given human demonstration. One common approach [6, 7, 8, 9] is to instantiate this as kinematic re-targeting task and compute open-loop robot actions that maximize a manually defined similarity between the achieved robot end-effector configuration and observed human hand poses (matching locations of fingertips). However, the human-robot embodiment mismatch makes it difficult for

the re-targeted actions to achieve precisely the same effects as the human ones, and the open-loop execution further makes the approach brittle to noise and scene variations. Another line of work for human imitation attempts to learn robot policies through online reinforcement learning [10, 11, 12, 13, 14], where the human demonstration helps define reward functions. While this can overcome the embodiment gap, such test-time online RL requires hours of online interaction and resets - making it difficult to adopt such methods for generic real-world manipulation tasks, especially in safety-critical scenarios. We instead seek to develop an approach for demonstration following that, akin to kinematic re-targeting, can be deployed 'one-shot' without any test-time training while still benefitting from generic learning-based priors for more precise closed-loop interaction.

Our approach builds on the insight that pre-trained diffusion policies can act as priors for robot action. Inspired by prior work in leveraging pre-trained diffusion models for image editing [15], we present *DemoDiffusion*–a formulation to utilize diffusion policy trained on robot interaction data for synthesizing coherent robot actions from a human demonstration. Specifically, we first perform kinematic re-targeting by extracting human hand poses from the demonstration and obtain an open-loop robot action trajectory. While this trajectory is typically suboptimal due to embodiment differences and lack of closed-loop feedback, it serves as an effective initialization that can be improved via a diffusion policy. We do so by injecting gaussian noise and appling the pre-trained diffusion policy to iteratively denoise the trajectory conditioned on robot observations, yielding a refined, executable sequence of robot actions. *DemoDiffusion* thus enables the robot to use the pre-trained policy as a prior and adapt the human-derived trajectory to its own embodiment and environment in a closed-loop manner.

In summary, we propose *DemoDiffusion*: a framework for robotic manipulation that allows a robot to perform generic tasks in natural environments by following a human demonstration with guidance from a pre-trained diffusion policy. Importantly, our framework doesn't require any robot demonstration of the target task in the target environment and doesn't require any online interaction or fine-tuning. Our experiments across simulation and real-world environments show that *DemoDiffusion* surpasses the performance of both, the base policy and re-targeted trajectory, even allowing the robot to perform tasks where pre-trained *generalist* policy completely fails.

II. Related Work

a) Generalist Manipulation Policies.: There has been a growing trend in developing '*generalist*' robotic policies that can perform multiple tasks based on a specified goal in the form of an image or a language instruction. Collecting large-scale robot interaction datasets [16, 17] combined with behavior cloning and vision-language pre-training [5, 2, 18, 1, 4, 3] is the predominant recipe for training such multi-task policies. However, since collecting large-scale robot interaction datasets in natural settings like homes and offices is challenging due to operational constraints, current best models still struggle to perform manipulation tasks zero-shot and thus are not yet deployable in-the-wild. Our work enables using such *generalist* diffusion policies for following a human demonstration, thereby being able to perform tasks that the pre-trained policy might be unable to zero-shot.

b) Robotic Manipulation with Non-Robot Datasets.: Instead of requiring large-scale robot interaction datasets, a growing body of works have begun utilizing human videos and large-scale web videos for robotics. A lot of these approaches have been enabled by recent advances in computer vision for representation learning [19, 20, 21], predicting tracks [22, 23], and reconstructing hand-object interactions from monocular videos [24, 25, 26, 27, 27]. Earlier works learned self-supervised visual representations from such non-robotic datasets [28, 29, 30] that can serve as the visual backbone of robotic policies [31, 32, 33, 34, 35, 36]. Recent works in this paradigm predict manipulation-relevant cues from web videos in the form of motion trajectories [37, 38] and object affordances [39, 40] and combine these predictive models with a limited amount of robot interaction data for training policies via conditional behavior cloning. Our work is orthogonal to these approaches in that instead of using human videos for training, we require a single human demonstration during deployment for guiding a pre-trained generalist policy to perform a new task.

c) One-Shot Imitation from Robot Demonstration.: One-shot imitation aims to enable robots to perform a new task with guidance from a single demonstration. Prior works have explored this in the context of a robot demonstration provided at test time, via training demonstrateion-conditioned policies [41]. Recent works have developed learning-based visual servoing systems [42] and algorithms for identifying object invariances [43] to replicate the robot end-effector actions from the demonstration onto novel configurations of objects. However, requiring a robot demonstration for imitation is restrictive for ubiquitous deployment as end-users might find it challenging to tele-operate the robot to collect demonstrations in natural human environments.

d) One-Shot Imitation from Human Demonstration.: To imitate a human demonstration for robot manipulation, a straightforward approach is kinematic re-targeting of the human hand pose to the robot end-effector pose per time-step, as demonstrated by some recent works [6, 7]. Although simple to implement, the human-robot embodiment mismatch typically introduces errors in the re-targeting and the open-loop execution is brittle to object pose variations during deployment. To remedy this, another line of works seek to learn this re-targeting implicitly by training a closed-loop policy from paired human-robot datasets [44, 45] enabling more flexible deployments. Since such paired datasets are difficult to collect, other works [10, 11, 46, 12] perform reinforcement learning with reward functions derived from video comparison. However, such methods require online interaction and resets, limiting their practicality. We propose an alternate paradigm:

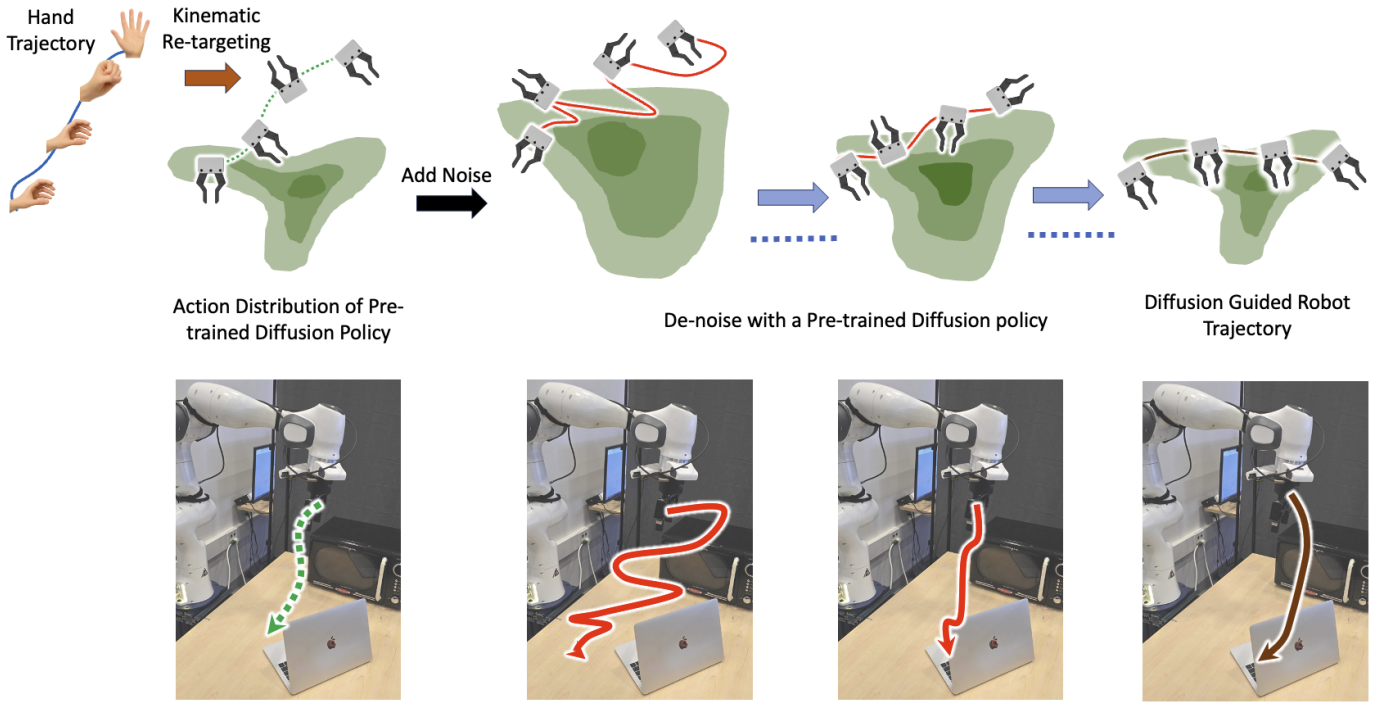

Fig. 2: **Re-targeted human hand trajectory to closed-loop robot action sequence, for the task $\mathcal{T}$: *"close the laptop"***. The dotted line shows the trajectory of robot end-effector poses after kinematic re-targeting. The olive contour plot depicts the distribution of trajectories from a pre-trained diffusion policy. Given a kinematic re-targeting, we first perturb it with Gaussian noise and progressively remove the noise by simulating the reverse SDE with the diffusion policy. This process gradually projects a potentially unfeasible but approximately correct re-targeting to the manifold of plausible robot actions that can perform real-world manipulation, in this case closing the laptop without missing the edge.

given the human demonstration, we refine re-targeted human behavior using a pre-trained robot policy, avoiding brittle replays and enabling closed-loop control. This formulation doesn't rely on expensive paired human-robot data collection and also doesn't require cumbersome test-time fine-tuning via interaction.

III. Method

We target the problem of one-shot visual imitation. Given a human demonstration $\mathbf{D}$ depicting a manipulation task with description $\mathcal{T}$, we want a robot manipulator to perform the same task. Unlike prior work that relies on test-time RL training [10] or paired human-robot datasets [44], we wish to enable such imitation 'one-shot' assuming access to a closed-loop diffusion policy $\bar{\pi}_\theta(\mathbf{a}_t|\mathbf{o}_{\leq t}, \mathcal{T})$ pre-trained on some broad robot interaction dataset $\mathcal{D}_{robot}$. Our assumption on the human demonstration $\mathbf{D}$ is also very minimal: it can either be an RGBD video or a multi-view video of a human doing a task, such that 3D hand poses per-timestep can be reliably extracted from it.

A. Overview

Our approach is based on two key insights. The first is that the trajectory of the hand pose in the human demonstration $\mathbf{D}$ provides useful information of the approximate trajectory the robot end-effector should follow, and we can perform *kinematic re-targeting* of the hand trajectory $\{\mathbf{h}_t\}_{t=0}^T$ to an open-loop robot end-effector trajectory $\{\hat{\mathbf{a}}_t\}_{t=0}^T$. The second insight is that the kinematically re-targeted robot trajectory has the correct form of motion, but these actions may not be very precise in the distribution of plausible robot actions given the current observation. A diffusion policy models this likelihood, and we can use a pre-trained '*generalist*' diffusion policy $\bar{\pi}_\theta(\mathbf{a}_t|\mathbf{o}_{\leq t}, \mathcal{T})$ to refine the retargeted robot actions in a closed loop manner, thus inferring actions that, while still similar to the human demonstration, are more likely under the policy and better aligned with the robot embodiment (see Fig. 2). We describe both these steps in detail below.

B. Kinematic Re-Targeting of Human Hand Trajectories to Robot End-Effector Poses

Given a human demonstration $\mathbf{D}$ of a manipulation task $\mathcal{T}$, our first step is to extract the 3D hand pose trajectory $\{\mathbf{h}_t\}_{t=0}^T$. Each hand pose $\mathbf{h}_t \in \mathbb{R}^{3\times J}$ corresponds to the 3D locations of J keypoints (e.g., wrist and fingertips), which we estimate using a pre-trained monocular hand pose estimator applied to each video frame [47]. These keypoints encode the motion of the human performing the task and form the basis for re-targeting to a robotic end-effector.

To translate this human motion into a robot-executable trajectory, we define a simple geometric mapping function

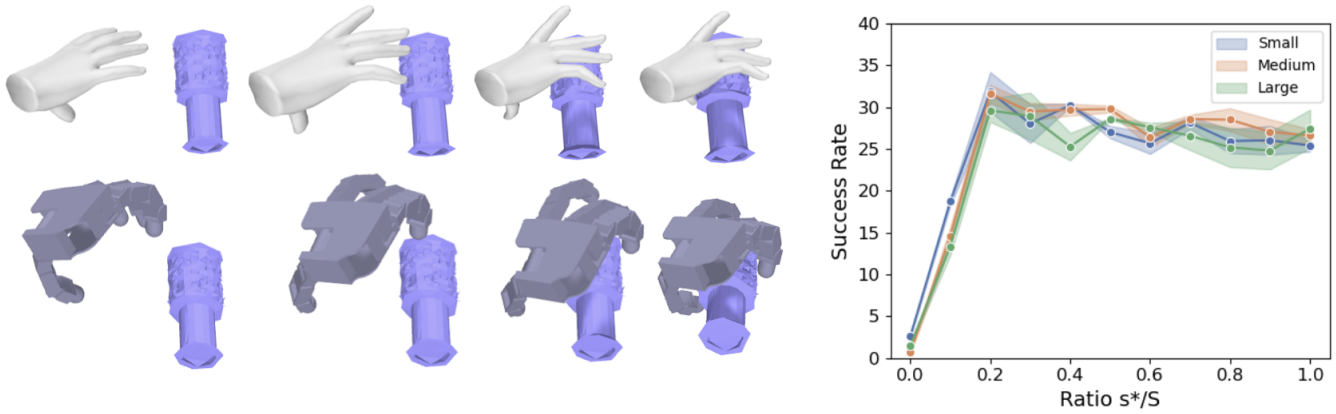

Fig. 3: **Dexterous Grasping Results in simulation.** On the left, we show a human demonstration followed by a rollout with *DemoDiffusion* for dexterous grasping. On the right, we plot the average and standard deviation of success rates over 3 seeds for dexterous grasping, corresponding to different levels of the diffusion step s^*. Here, $s^*/S = 0$ corresponds to kinematic retargeting and $s^*/S = 1$ corresponds to the robot policy.

Method	*Small*	*Medium*	*Large*	***Average***
Robot Policy	25.4	26.6	27.4	26.5
Kinematic Retargeting	2.6	0.7	1.5	1.6
DemoDiffusion	**31.8**	**31.6**	**29.6**	**31.0**

TABLE I: **Simulated Dexterous Grasping Results.** The numbers show the average and standard deviation of success rates over 3 seeds. We use $s^*/S = 0.2$, where the robot policy uses $S = 1000$. *DemoDiffusion* consistently outperforms both baselines, across all groups.

$f_{\text{retarget}} : \mathbb{R}^{3\times J} \to \mathbb{R}^6$ that converts the human hand pose $\mathbf{h}_t$ into a robot configuration $\hat{\mathbf{a}}_t = f_{\text{retarget}}(\mathbf{h}_t)$. The mapping aligns the wrist pose of the human to the robot end-effector pose. For a two-finger gripper, we use the distance between the thumb and the remaining fingers of the hand mesh to infer a binary robot grasp, and for a dexterous robotic hand we match the robot hand's fingertip positions to those of a human hand using inverse kinematics.

This kinematic re-targeting procedure yields a full trajectory $\{\hat{\mathbf{a}}_t\}_{t=0}^T$ in the robot's configuration space, which can be executed as an open-loop policy. However, due to differences in morphology and embodiment between humans and robots, the absence of environment feedback, and the inaccuracy of the hand estimation module, this trajectory often leads to suboptimal or unstable behavior during open-loop execution. We thus treat this as the basis for searching for a plausible robot action trajectory via denoising with a (pre-trained) closed-loop diffusion policy.

C. Closed-Loop Denoising of Robot Actions with a Pre-Trained Diffusion Policy

A diffusion policy $\bar{\pi}_\theta(\mathbf{a}_t|\mathbf{o}_{\le t}, \mathcal{T})$ [48, 49] pre-trained on diverse offline robot interaction data reliably models the distribution of plausible robot actions $\mathbf{a}_t$ given previous observations $\mathbf{o}_{\le t}$. To predict actions $\mathbf{a}_t$, diffusion policies start with Gaussian noise $\tilde{\mathbf{a}}_t^{(S)} \sim \mathcal{N}(0, I)$ and iteratively denoise it at different diffusion steps s:

$$\tilde{\mathbf{a}}_t^{(s-1)} = \bar{\pi}_\theta\left(\tilde{\mathbf{a}}_t^{(s)}, \mathbf{o}_{\le t}\right), \quad s = S, S-1 \ldots, 0 \tag{1}$$

We use this to modify the kinematically re-targeted trajectory $\{\hat{\mathbf{a}}_t\}_{t=0}^T$ for obtaining robot actions that follow the high-level motion in the human trajectory and still lie within the distribution of plausible actions under $\bar{\pi}_\theta$. To do this, we modify this typical reverse diffusion process so that instead of starting with pure noise $\tilde{\mathbf{a}}_t^{(S)}$ at step S, we start from an intermediate time-step s^* such that $0 < s^* < S$. We define $\{\tilde{\mathbf{a}}_t^{(s^*)}\}_{t=0}^T$ to be a noisy version of the kinematically re-targeted trajectory as follows:

$$\tilde{\mathbf{a}}_t^{(s^*)} = \sqrt{\alpha_{s^*}}\hat{\mathbf{a}}_t + \sqrt{1-\alpha_{s^*}}\epsilon_t, \quad \epsilon_t \sim \mathcal{N}(0, \sigma^2\mathbf{I}) \tag{2}$$

Here α corresponds to the diffusion schedule of the pre-trained policy. This procedure, inspired by SDEdit [15] which adopted a similar approach for image editing, relies on the assumption that $\hat{\mathbf{a}}_t$ is an approximate version of the ideal trajectory that should be executed by the robot, and it potentially lies outside the distribution of feasible trajectories under the pre-trained diffusion policy $\bar{\pi}_\theta$. The diffusion policy $\bar{\pi}_\theta$ then performs iterative denoising steps, conditioned on the robot's observations $o_{\le t}$, to refine this noisy trajectory into feasible robot actions, based on equation (2) above.

After s^* denoising steps, the final output $\mathbf{a}_t = \tilde{\mathbf{a}}_t^{(0)}$ is deployed on the robot. Importantly, this process is carried out

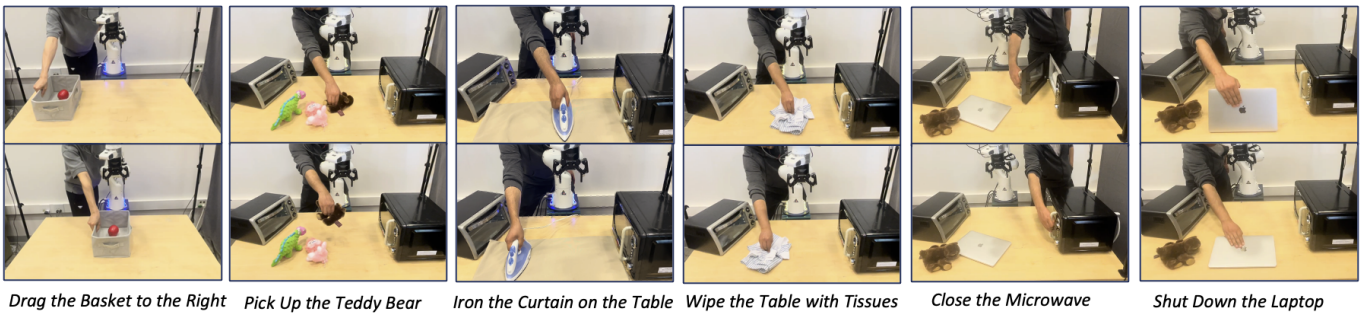

Fig. 4: **Real-World Manipulation Tasks.** Each column corresponds to the respective human demonstrations for the real-world manipulation tasks, showing two frames from each demonstration. We also show the respective language task descriptions $\mathcal{T}$ that go as input to the pre-trained diffusion policy.

Method	Shut Laptop	Close Microwave	Drag Basket	Wipe Table	Iron Curtain	Pick up Bear	Avg.
Pi-0	0	0	**100**	60	**20**	0	30
Kinematic Retargeting	0	20	60	0	0	40	20
DemoDiffusion	**20**	**40**	**100**	**80**	**20**	**60**	**53**

TABLE II: **Quantitative Results for Real-World Robot Manipulation.** Success rates (%) over 5 trials per task. Please refer to the Appendix for more comprehensive quantitative comparisons.

in a closed-loop manner: the policy uses real-time observations from the cameras in the scene to iteratively improve its predictions, thereby compensating for embodiment mismatch and external perturbations (e.g., object slippage or occlusion). The key hyperparameter in this process for *DemoDiffusion* is the diffusion step s^* that trades off between the faithfulness to the demonstration and the likelihood under the robot policy – in the limit $s^* = S$ we recover a rollout from the base policy $\bar{\pi}_\theta$ and in the limit $s^* = 0$ we recover the kinematically re-targeted trajectory $\{\hat{\mathbf{a}}_t\}_{t=0}^{T}$.

IV. Experiments

We evaluate our approach on dexterous grasping in simulation and across diverse real-world tasks comprising of prehensile and non-prehensile table-top manipulation. Through experiments, we aim to understand the following research questions:

- Can *DemoDiffusion* outperform pure kinematic re-targeting from the human demonstration?
- Do the human demonstrations allow *DemoDiffusion* to perform new tasks where the pre-trained diffusion policy fails?
- How to effectively tradeoff between faithfulness to the human demonstration and performing the task reliably, with varying noise level s^*/S?

A. Dexterous Grasping in Simulation

To verify our intuition, we first consider a simulation environment where the target task is restricted to picking up a generic object with a 16-DOF four-fingered Allegro hand. Specifically, we train a dexterous grasping policy across a small set of generic objects in simulation, and test our method on human hand grasping trajectories on a different set of objects. This serves as the pre-trained diffusion model for this experiment.

For training the robot policy, we collect total 985 grasping trajectories of Allegro hand over 58 training objects[50] (randomly sampled 26 and 32 objects from ShapeNet [51] and PartNet [52], respectively), resulting in 22.2 success rate. We use a variant of 3D Diffusion Policy [53], which takes 3D point tracks of an object instead of point clouds. At test time, we provide human grasping trajectories on a subset of Objvaerse dataset [54], which are unseen during training. The test set contains total 1220 objects, with 1 human grasping trajectory per object, provided by GraspXL [50]. Note that there is no error from hand estimation or initial configuration of object in this setup. We use ground-truth human hand 3d keypoints, and the object is located at the same location to that of human demonstration.

a) Results.: As in GraspXL [50], we group the objects based on their size, resulting in small, medium, and large objects. We compare *DemoDiffusion* with two baselines. The first is open-loop kinematic re-targeting obtained from the human demonstration. The second is directly deploying the robot policy we trained.

Table. I shows the main results. Our method outperforms baselines, with more performance gain for small-sized objects. We hypothesize this roots from that larger objects are easier to grasp, supported by robot policy performance improving for larger sized objects. This experiment shows that even in a highly controlled setup where the robot environment is initialized identically to that of human demonstration, kinematic retargeting cannot solve the task, while slightly refining with a generalist grasping policy significantly improves the performance. Using kinematic retargeting to initiate the denoising

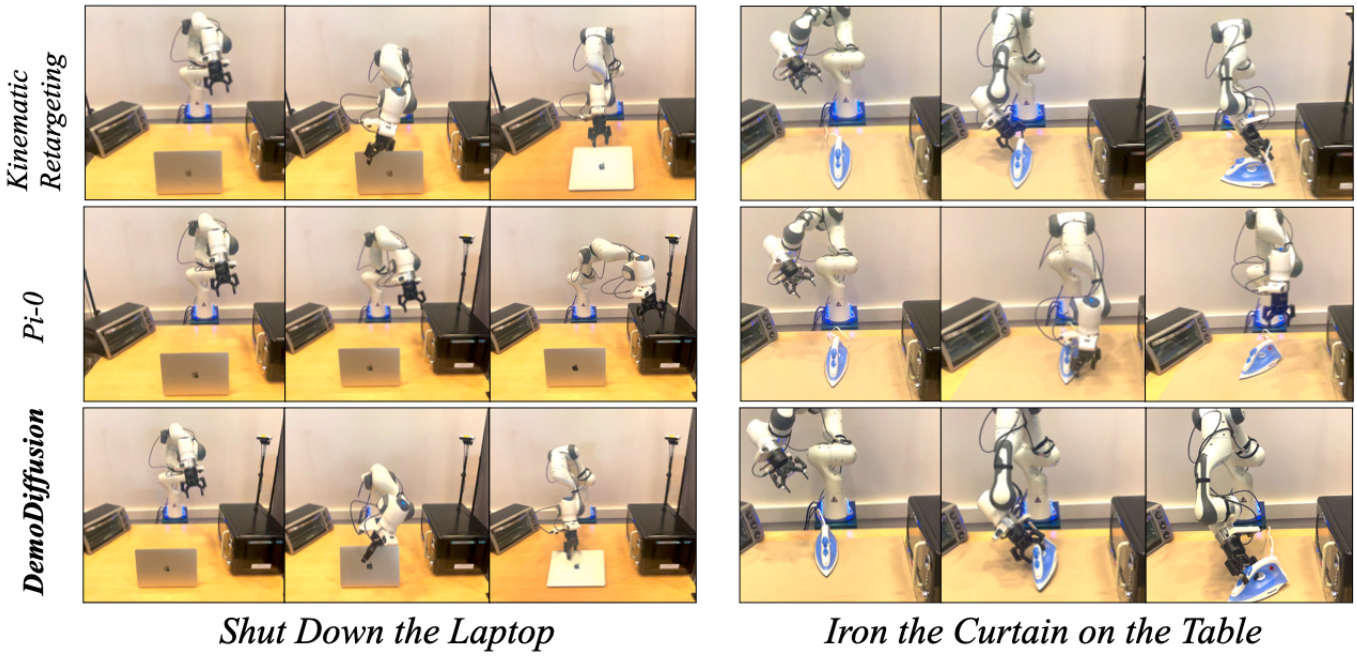

Fig. 5: **Qualitative comparisons for real-robot manipulation.** We show rollouts from two tasks for *DemoDiffusion* and the baselines. The respective strips show progression of rollouts from left to right including the start frame, an intermediate frame, and the final frame. Frames are cropped for better visibility. Please refer to the supplementary material for detailed visualizations.

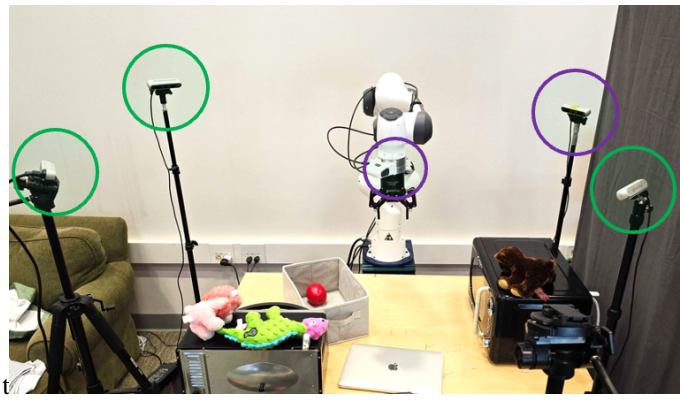

Fig. 6: **Workspace with 5 cameras.** We use the four external cameras for triangulation to obtain the global pose of the hand mesh from a human demonstration. The pre-trained diffusion policy uses the two cameras marked in purple.

process also helps the denoising process itself, outperforming the robot policy. Additionally, *DemoDiffusion* achieves higher inference speed, by the factor of S/s^*.

b) Ablation studies.: We analyze the influence of s^* on *DemoDiffusion* in Fig. 3. Overall, we observe a consistent trend of performance gain from $s^*/S = 1$ to $s^*/S = 0.2$, showing that *DemoDiffusion* is robust to the choice of hyper-parameter.

B. Real-World Manipulation

For the real-world experiments, we use a Franka Emika Panda arm equipped with a two-finger gripper from Robotiq. We use the pre-trained diffusion policy $\bar{\pi}_\theta(\mathbf{a}_t|\mathbf{o}_{\leq t}, \mathcal{T})$ called Pi-0 released by Physical Intelligence [1]. This policy takes as input a language task specification $\mathcal{T}$, and observations from two cameras in the scene, and outputs per-timestep joint velocity. This policy was trained on a large offline dataset of tele-operated robot demonstrations, and is a *generalist* policy that can perform a broad set of tasks. We do not fine-tune or adapt this policy in any way and directly use it for *DemoDiffusion*. We use an off-the-shelf monocular hand reconstruction model, Hamer [47] to obtain per-timestep hand mesh reconstructions from each of the external cameras given a human demonstration in the scene. Our experiments follow the protocol of a human first demonstrating an object manipulation in the scene, followed by robot execution after re-setting the scene.

a) Workspace.: Fig. 6 describes our real-robot manipulation setup. The scene has 5 cameras including 4 Real-sense external cameras and 1 Zed-mini wrist camera mounted on the robot gripper. We calibrate the four external cameras and use them for triangulating the 3D hand pose so that we can obtain the position and orientation of a human hand in the world coordinates, with origin at the base of the robot. The pre-trained diffusion policy Pi-0 requires just two cameras (marked in purple) and the policy does not use any calibration information.

b) Tasks.: We perform evaluations for 6 different manipulation tasks, the langauge description and the human demonstration for which are described in Fig. 4. Each of these tasks require a different manipulation strategy involving either prehensile or non-prehensile manipulation. We collect just a single human demonstration per task but for robot execution, we make organic variations in the location of objects in the scene. Additionally, we add other objects in the scene as visual distractors, in order to see whether *DemoDiffusion* can generalize under such settings.

c) Baselines.: For the real-world experiments we compare *DemoDiffusion* with two baselines. The first is open-loop kinematic re-targeting obtained from the human demonstration. The second is directly deploying the Pi-0 model with the language instruction for the respective task. Since Pi-0 operates in the join velocity space, for *DemoDiffusion* we first convert the kinematically re-targeted trajectory to the joint velocity space with IK and then add noise to the trajectory, followed by denoising with Pi-0. We perform comparisons of *DemoDiffusion* with noise level $s^*/S = 0.1$. Note that in the limit of 0 noise added, we recover the first baseline, kinematic re-targeting, and in the limit of maximum noise 1 we obtain the second baseline Pi-0.

d) Qualitative Visualizations.: In Fig. 5 we show rollouts for two tasks across all the methods. We observe that Pi-0 is only able to perform reaching behaviors but doesn't follow the required manipulation trajectory necessary to perform these tasks. On the other hand, we observe that (open-loop) kinematic re-targeting from the human demonstration has a plausible motion of the robot end-effector, but it completely misses grasping the iron, and loses contact with the laptop halfway. Compared to these, our approach *DemoDiffusion* is able to reach the respective object and also follow the approximate motion of the re-targeted trajectory while remaining in contact with the object in order to solve the tasks.

e) Quantitative Comparisons.: In Table II we compare *DemoDiffusion* and the baselines across all the real-world manipulation tasks. We observe that *DemoDiffusion* either performs comparably or outperforms all the baselines in all the tasks. Interestingly, even for tasks like closing the laptop where both the baselines fail, *DemoDiffusion* is able to leverage the best of both the baselines (that represent extremes of our approach under the hyper-parameter s^*) in order to achieve a non-zero success rate.

V. Discussion

We presented *DemoDiffusion*, a simple approach to leverage *generalist* diffusion models for imitating a human demonstration. By re-targeting hand motion from the human demonstration into an approximate robot end-effector trajectory and modifying it through de-noising with a pre-trained diffusion policy, *DemoDiffusion* enables the robot to produce actions that are both aligned with the demonstrated intent and contextually plausible. Our method consistently outperforms both the base diffusion policy and the raw re-targeted trajectory across diverse tasks in simulation and the real world, without requiring additional training, paired demonstrations, or reward functions. We envision that this simple approach can serve as a starting point for future efforts at human imitation across generic scenarios and perhaps also yield an improved exploration strategy for methods pursuing policy adaptation with online RL.

Limitations

Our approach has several limitations. First, it assumes that the robot should act similarly to the human in order to successfully complete the task, which may not hold in scenarios requiring different strategies due to embodiment or environmental constraints. Second, while effective for one-shot imitation, the method does not produce a reusable task policy that can generalize across variations of the task. Third, the quality of the re-targeted trajectory is crucial—accurate 3D human motion capture is challenging and errors in retargeting can impact downstream performance. Additionally, the method implicitly assumes that the timing and speed of human and robot actions are aligned; extending the approach to allow for temporal alignment at test time is a promising direction. Finally, for long-horizon tasks, the robot may gradually drift away from the intended behavior, as the model's ability to stay close to the human demonstration weakens over extended sequences.

References

[1] Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

[2] Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pages 2165–2183. PMLR, 2023.

[3] 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 *Conference on Robot Learning*, pages 991–1002. PMLR, 2022.

[4] Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

[5] Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Joseph Dabis, Chelsea Finn, Keerthana Gopalakrishnan, Karol Hausman, Alex Herzog, Jasmine Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[6] Jinhan Li, Yifeng Zhu, Yuqi Xie, Zhenyu Jiang, Mingyo Seo, Georgios Pavlakos, and Yuke Zhu. Okami: Teaching

humanoid robots manipulation skills through single video imitation. In *CoRL*, 2024.

[7] Georgios Papagiannis, Norman Di Palo, Pietro Vitiello, and Edward Johns. R+ x: Retrieval and execution from everyday human videos. *arXiv preprint arXiv:2407.12957*, 2024.

[8] Chen Wang, Haochen Shi, Weizhuo Wang, Ruohan Zhang, Li Fei-Fei, and C Karen Liu. Dexcap: Scalable and portable mocap data collection system for dexterous manipulation. In *RSS*, 2024.

[9] Sirui Chen, Chen Wang, Kaden Nguyen, Li Fei-Fei, and C Karen Liu. Arcap: Collecting high-quality human demonstrations for robot learning with augmented reality feedback. *arXiv preprint arXiv:2410.08464*, 2024.

[10] Shikhar Bahl, Abhinav Gupta, and Deepak Pathak. Human-to-robot imitation in the wild. *arXiv preprint arXiv:2207.09450*, 2022.

[11] Irmak Guzey, Yinlong Dai, Georgy Savva, Raunaq Bhirangi, and Lerrel Pinto. Bridging the human to robot dexterity gap through object-oriented rewards. *arXiv preprint arXiv:2410.23289*, 2024.

[12] Yuanpei Chen, Chen Wang, Yaodong Yang, and C Karen Liu. Object-centric dexterous manipulation from human motion data. *arXiv preprint arXiv:2411.04005*, 2024.

[13] Xueyi Liu, Jianibieke Adalibieke, Qianwei Han, Yuzhe Qin, and Li Yi. Dextrack: Towards generalizable neural tracking control for dexterous manipulation from human references. *arXiv preprint arXiv:2502.09614*, 2025.

[14] Xueyi Liu, Kangbo Lyu, Jieqiong Zhang, Tao Du, and Li Yi. Quasisim: Parameterized quasi-physical simulators for dexterous manipulations transfer. In *ECCV*, 2024.

[15] 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.

[16] Hao-Shu Fang, Hongjie Fang, Zhenyu Tang, Jirong Liu, Junbo Wang, Haoyi Zhu, and Cewu Lu. Rh20t: A robotic dataset for learning diverse skills in one-shot. *arXiv preprint arXiv:2307.00595*, 2023.

[17] Homer Rich Walke, Kevin Black, Tony Z Zhao, Quan Vuong, Chongyi Zheng, Philippe Hansen-Estruch, Andre Wang He, Vivek Myers, Moo Jin Kim, Max Du, et al. Bridgedata v2: A dataset for robot learning at scale. In *Conference on Robot Learning*, pages 1723–1736. PMLR, 2023.

[18] Homanga Bharadhwaj, Jay Vakil, Mohit Sharma, Abhinav Gupta, Shubham Tulsiani, and Vikash Kumar. Roboagent: Generalization and efficiency in robot manipulation via semantic augmentations and action chunking. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024.

[19] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16000–16009, 2022.

[20] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[21] Lingting Zhu, Guying Lin, Jinnan Chen, Xinjie Zhang, Zhenchao Jin, Zhao Wang, and Lequan Yu. Large images are gaussians: High-quality large image representation with levels of 2d gaussian splatting. *arXiv preprint arXiv:2502.09039*, 2025.

[22] Nikita Karaev, Ignacio Rocco, Benjamin Graham, Natalia Neverova, Andrea Vedaldi, and Christian Rupprecht. Cotracker: It is better to track together. *arXiv preprint arXiv:2307.07635*, 2023.

[23] Carl Doersch, Ankush Gupta, Larisa Markeeva, Adrià Recasens, Lucas Smaira, Yusuf Aytar, João Carreira, Andrew Zisserman, and Yi Yang. Tap-vid: A benchmark for tracking any point in a video. *Advances in Neural Information Processing Systems*, 35:13610–13626, 2022.

[24] Yu Rong, Takaaki Shiratori, and Hanbyul Joo. Frankmocap: A monocular 3d whole-body pose estimation system via regression and integration. In *CVPR*, 2021.

[25] Georgios Pavlakos, Dandan Shan, Ilija Radosavovic, Angjoo Kanazawa, David Fouhey, and Jitendra Malik. Reconstructing hands in 3d with transformers. In *CVPR*, 2024.

[26] Yufei Ye, Abhinav Gupta, and Shubham Tulsiani. What's in your hands? 3d reconstruction of generic objects in hands. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3895–3905, 2022.

[27] Yufei Ye, Poorvi Hebbar, Abhinav Gupta, and Shubham Tulsiani. Diffusion-guided reconstruction of everyday hand-object interaction clips. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 19717–19728, 2023.

[28] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, et al. Scaling egocentric vision: The epic-kitchens dataset. In *ECCV*, 2018.

[29] Kristen Grauman, Andrew Westbury, Eugene Byrne, Zachary Chavis, Antonino Furnari, Rohit Girdhar, Jackson Hamburger, Hao Jiang, Miao Liu, Xingyu Liu, et al. Ego4d: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18995–19012, 2022.

[30] Kristen Grauman, Andrew Westbury, Lorenzo Torresani, Kris Kitani, Jitendra Malik, Triantafyllos Afouras, Kumar Ashutosh, Vijay Baiyya, Siddhant Bansal, Bikram Boote, et al. Ego-exo4d: Understanding skilled human activity from first-and third-person perspectives. In *Proceedings of the IEEE/CVF Conference on Computer Vision and*

Pattern Recognition, pages 19383–19400, 2024.

[31] Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022.

[32] Simone Parisi, Aravind Rajeswaran, Senthil Purushwalkam, and Abhinav Gupta. The unsurprising effectiveness of pre-trained vision models for control. In *International Conference on Machine Learning*, pages 17359–17371. PMLR, 2022.

[33] Siddharth Karamcheti, Suraj Nair, Annie S Chen, Thomas Kollar, Chelsea Finn, Dorsa Sadigh, and Percy Liang. Language-driven representation learning for robotics. *arXiv preprint arXiv:2302.12766*, 2023.

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

[35] Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. Vip: Towards universal visual reward and representation via value-implicit pre-training. *arXiv preprint arXiv:2210.00030*, 2022.

[36] Yecheng Jason Ma, Vikash Kumar, Amy Zhang, Osbert Bastani, and Dinesh Jayaraman. Liv: Language-image representations and rewards for robotic control. In *International Conference on Machine Learning*, pages 23301–23320. PMLR, 2023.

[37] Homanga Bharadhwaj, Roozbeh Mottaghi, Abhinav Gupta, and Shubham Tulsiani. Track2act: Predicting point tracks from internet videos enables generalizable robot manipulation. In *European Conference on Computer Vision*, pages 306–324. Springer, 2024.

[38] Kenneth Shaw, Shikhar Bahl, and Deepak Pathak. Videodex: Learning dexterity from internet videos. In *CoRL*, 2023.

[39] Mohan Kumar Srirama, Sudeep Dasari, Shikhar Bahl, and Abhinav Gupta. Hrp: Human affordances for robotic pre-training. In *RSS*, 2024.

[40] Shikhar Bahl, Russell Mendonca, Lili Chen, Unnat Jain, and Deepak Pathak. Affordances from human videos as a versatile representation for robotics. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13778–13790, 2023.

[41] Sudeep Dasari and Abhinav Gupta. Transformers for one-shot visual imitation. In *Conference on Robot Learning*, pages 2071–2084. PMLR, 2021.

[42] Eugene Valassakis, Georgios Papagiannis, Norman Di Palo, and Edward Johns. Demonstrate once, imitate immediately (dome): Learning visual servoing for one-shot imitation learning. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8614–8621. IEEE, 2022.

[43] Xinyu Zhang and Abdeslam Boularias. One-shot imitation learning with invariance matching for robotic manipulation. *arXiv preprint arXiv:2405.13178*, 2024.

[44] Vidhi Jain, Maria Attarian, Nikhil J Joshi, Ayzaan Wahid, Danny Driess, Quan Vuong, Pannag R Sanketi, Pierre Sermanet, Stefan Welker, Christine Chan, et al. Vid2robot: End-to-end video-conditioned policy learning with cross-attention transformers. *arXiv preprint arXiv:2403.12943*, 2024.

[45] Sungjae Park, Seungho Lee, Mingi Choi, Jiye Lee, Jeonghwan Kim, Jisoo Kim, and Hanbyul Joo. Learning to transfer human hand skills for robot manipulations. *arXiv preprint arXiv:2501.04169*, 2025.

[46] Tyler Ga Wei Lum, Olivia Y Lee, C Karen Liu, and Jeannette Bohg. Crossing the human-robot embodiment gap with sim-to-real rl using one human demonstration. *arXiv preprint arXiv:2504.12609*, 2025.

[47] Georgios Pavlakos, Dandan Shan, Ilija Radosavovic, Angjoo Kanazawa, David Fouhey, and Jitendra Malik. Reconstructing hands in 3d with transformers. In *CVPR*, 2024.

[48] Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, page 02783649241273668, 2023.

[49] Moritz Reuss, Maximilian Li, Xiaogang Jia, and Rudolf Lioutikov. Goal conditioned imitation learning using score-based diffusion policies. In *Robotics: Science and Systems*, 2023.

[50] Hui Zhang, Sammy Christen, Zicong Fan, Otmar Hilliges, and Jie Song. Graspxl: Generating grasping motions for diverse objects at scale. In *European Conference on Computer Vision*, pages 386–403. Springer, 2024.

[51] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[52] Kaichun Mo, Shilin Zhu, Angel X Chang, Li Yi, Subarna Tripathi, Leonidas J Guibas, and Hao Su. Partnet: A large-scale benchmark for fine-grained and hierarchical part-level 3d object understanding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 909–918, 2019.

[53] Yanjie Ze, Gu Zhang, Kangning Zhang, Chenyuan Hu, Muhan Wang, and Huazhe Xu. 3d diffusion policy: Generalizable visuomotor policy learning via simple 3d representations. *arXiv preprint arXiv:2403.03954*, 2024.

[54] Matt Deitke, Dustin Schwenk, Jordi Salvador, Luca Weihs, Oscar Michel, Eli VanderBilt, Ludwig Schmidt, Kiana Ehsani, Aniruddha Kembhavi, and Ali Farhadi. Objaverse: A universe of annotated 3d objects. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 13142–13153, 2023.

[55] Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A

large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

[56] Javier Romero, Dimitrios Tzionas, and Michael J Black. Embodied hands: Modeling and capturing hands and bodies together. *arXiv preprint arXiv:2201.02610*, 2022.

APPENDIX

A. Hardware Setup

We replicate DROID [55] setup for the hardware setup. Due to hardware availability, we use RealSense D455 instead of ZED 2 for the rear view camera.

B. Task Setup Description

The overview of scenes for each task at test time is shown in Fig. 7. Overall, kinematic retargeting may fail due to small misalignment of object placement, inaccurate 3D hand keypoints, or camera extrinsics, etc. Pi-0 may fail if it cannot identify the object to interact with and manipulate other objects in the scene. It may also reach the target object but fail to manipulate in the desired way. Note that previous approaches [6, 7, 8, 9] use kinematic retargeting to transfer human demonstration to the robot.

a) Drag the Basket to the Right.: The task is to drag the basket to the right side of the table. The human demonstration grasps the edge of the basket and pulls it to the right. The episode is considered successful if the robot grasps the basket/ puts the gripper inside the basket and move it to the right side of the table.

b) Pick Up the Teddy Bear.: The task is to pick up the teddy bear located at the center of the table. The human demonstration grasps the neck of the bear and picks it up. The episode is considered successful if the robot picks up the bear by the end of the episode.

c) Iron Curtain on the Table.: The task is to grasp the ironing machine located at the center of the table and slide it to the right side of the curtain. The episode is considered successful if the robot grasps/pushes the ironing machine to the right part of the curtain.

d) Wipe the Table with Tissues.: The task is to pinch/poke the given tissue and wipe the table. The episode is considered successful if the robot pinches/pokes the tissue and performs one stroke to any side of the table without missing it.

e) Close the Microwave.: The task is to close the microwave located at the corner of the table. The episode is considered successful if the handle part touches the body part of the microwave at the end of the episode.

f) Shut Down the Laptop.: The task is to close the laptop located in front of the robot. The episode is considered successful if the laptop is closed at the end of the episode.

C. Text Prompt.

We used the following text conditions for Pi-0 [1], which are also used for *DemoDiffusion*.

Drag the Basket to the Right: *move the basket to the right*
Pick Up the Teddy Bear: *pick up the teddy bear*
Iron Curtain on the Table: *move the ironing machine to the right*
Wipe the Table with Tissues: *wipe the table*
Close the Microwave: *close the microwave*
Shut Down the Laptop: *close the laptop*

TABLE III: Hyperparameters in Real-world.

Hyperparameter	Value
Open Loop Horizon	8
Predict Action Horizon	10
Denoising Steps at Inference (Pi-0)	20

D. Hyperparameters

To show our method's applicability across diverse setups, we do not change the speed of the given human demonstration when performing kinematic retargeting. At inference, we use the following hyperparameters for Pi-0 and *DemoDiffusion*, with only difference on the number of denoising steps, as *DemoDiffusion* starts denoising from intermediate steps. Specifically, we use the following hyperparameters. Depending on the noise level s^*/S, *DemoDiffusion* uses $20 \times s^*/S$ denoising steps.

E. Additional Results

To test *DemoDiffusion*'s robustness to noise level s^*/S, we additionally test *DemoDiffusion* with noise level $s^*/S = 0.2$ in the real world. The results are included in Table. IV. *DemoDiffusion* shows overall similar performance across two different noise levels, showing its applicability without much effort for tuning the noise level.

F. Task Setup Description

In our simulation experiment, the task is to pick up the object. The rollout is considered success when the object's z position is higher than 0.10 *m* at the end of the episode.

a) Training Data.: As we don't have a *generalist* pre-trained diffusion policy in simulation, we first train a diffusion policy while constraining the target task to picking up generic objects. Specifically, we collected total 985 grasping trajectories of Allegro hand over 58 objects in the RaiSim simulator for training. The trajectories were generated by rolling out the expert Allegro RL policy with random grasping directions. The 58 objects are randomly sampled from ShapeNet [51] (26 objects) and PartNet [52] (32 objects). The training objects and expert Allegro RL policy are provided by GraspXL[50].

b) Test Data.: At test time, we provide human grasping trajectories for a different set of objects. As it is challenging to provide human demonstration in simulation, we instead use MANO [56] hand RL policy to collect human demonstrations. The test set contains total 1220 objects, which is a subset of Objvaerse dataset [54]. The objects are divided in three groups based on its size: small scale $s \in [3, 5]cm$ medium scale $m \in [5, 7]cm$, and large scale $l \in [7, 9]cm$. As the object meshes don't contain material information, we assume fixed density, resulting in diverse mass. We additionally use the same friction coefficient for all objects. Again, test objects and expert MANO [56] RL policy are provided by GraspXL[50]. Additionally, we do not change the speed of the given human demonstration when performing kinematic retargeting.

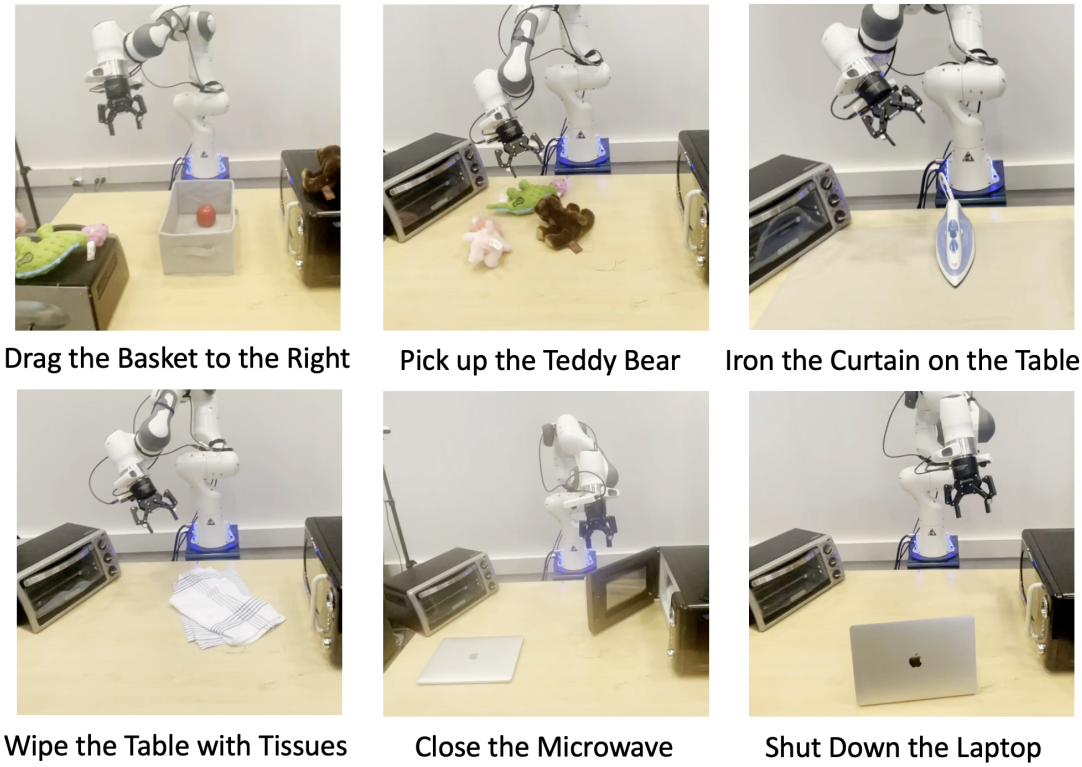

Fig. 7: **Real-World Manipulation Tasks at Test Time.** Each image corresponds to an example of the scene for each task at test time. To test our method in a natural setup, we add other objects in the scene as visual distractors. For each rollout, we use the same set of distractors across methods.

TABLE IV: **Additional Quantitative Results for Real-World Robot Manipulation.** Success rates (%) over 5 trials per task. $s^*/S = 0.1$ corresponds to results included in the main text, and $s^*/S = 0.2$ corresponds to new results.

Method	Shut Laptop	Close Microwave	Drag Basket	Wipe Table	Iron Curtain	Pick up Bear	Avg.
Pi-0	0	0	**100**	60	**20**	0	30
Kinematic Retargeting	0	20	60	0	0	40	20
*DemoDiffusion*w$(s^*/S = 0.1)$	20	**40**	**100**	80	**20**	**60**	**53**
DemoDiffusion$(s^*/S = 0.2)$	**60**	**40**	**100**	**100**	0	20	**53**

G. Robot Policy Implementation Details

We train a hierarchical policy, where the high-level planner predicts the robot's future goal state, and the low-level controller predicts the action given the desired future robot state. Both policies were trained using a variant of 3D Diffusion Policy [53], where we modify the point cloud encoder to take point tracks, for a richer representation of the object. Specifically, while the original point encoder takes object points at each timestep, the modified encoder takes a sequence of object points (object tracks) as input. High-level planner and low-level controller use a subset of the following as input and output. All rotations are represented in Euler angles.

a) Robot state: contains its wrist position in the world frame, rotation in the world frame, and absolute hand joint angles.

b) Object state: contains 3D points of its surface. For a history of robot state, we assume 3D point tracks to be given(i.e. each 3D point across timesteps corresponds to the same point of the object).

c) Robot goal state: contains its wrist position in the local frame(wrist frame at previous timestep), rotation in the local frame, and relative hand joint angles to the previous timestep.

d) Action: contains the desired robot's wrist position in the local frame(previous wrist frame), desired rotation in the local frame, and desired relative hand joint angles to the previous timestep. Note that the action and the robot's goal state are not the same. For instance, the desired hand joint angles can penetrate the object surface, while the robot's state cannot. Such a difference creates contact force, enabling a stable grasp.

e) High-level Planner: The high-level planner takes in a history of robot states and object states, and predicts a sequence of robot goal states. At test-time, we use the kinematic retargeting result as an initial estimate for the robot's goal state. We then apply *DemoDiffusion* to refine the robot goal

TABLE V: Hyperparameters in Simulation.

Hyperparameter	**Value**
H	4
N_{obs}	2
N_{act}	1
N_{goal}	2
$N_{latency}$	0
Number of Object Points	1024
Train Timesteps	1000
Inference Timesteps	10
Episode Length	155
Wrist Position Pgain	100.0
Wrist Position Dgain	0.1
Wrist Rotation Pgain	100.0
Wrist Rotation Dgain	0.2
Finger Rotation Pgain	100.0
Finger Rotation Dgain	0.2

state, and feed it to the low-level controller.

f) Low-level Controller: The low-level controller takes in a history of robot states and a sequence of robot goal states, and predicts a sequence of robot actions. The goal state encoder has the same architecture to robot state encoder. Once it outputs the action, the action is used as a position target for a PD controller provided in the RaiSim simulator.

H. Hyperparameters

For policy training, we use the default hyperparameters provided in the 3D Diffusion Policy [53] with minimal modifications. For the simulation environment, we use the default hyperparameters provided by GraspXL [50]. Specifically, we use the following same parameters for both the high-level planner and low-level controller.