ViViDex: Learning Vision-based Dexterous Manipulation from Human Videos

Zerui Chen[1], Shizhe Chen[1], Etienne Arlaud[1], Ivan Laptev[2] and Cordelia Schmid[1]

***Abstract*— In this work, we aim to learn a unified vision-based policy for multi-fingered robot hands to manipulate a variety of objects in diverse poses. Though prior work has shown benefits of using human videos for policy learning, performance gains have been limited by the noise in estimated trajectories. Moreover, reliance on privileged object information such as ground-truth object states further limits the applicability in realistic scenarios. To address these limitations, we propose a new framework ViViDex to improve vision-based policy learning from human videos. It first uses reinforcement learning with trajectory guided rewards to train state-based policies for each video, obtaining both visually natural and physically plausible trajectories from the video. We then rollout successful episodes from state-based policies and train a unified visual policy without using any privileged information. We propose coordinate transformation to further enhance the visual point cloud representation, and compare behavior cloning and diffusion policy for the visual policy training. Experiments both in simulation and on the real robot demonstrate that ViViDex outperforms state-of-the-art approaches on three dexterous manipulation tasks. Project website: `zerchen.github.io/projects/vividex.html`**

I. Introduction

People possess a remarkable ability to manipulate objects effortlessly with their hands, guided by visual perception. Despite significant progress, replicating the dexterity of human hands with multi-fingered robot hands remains challenging [1], [2], [3], [4], [5]. This is largely due to the complexity of translating visual signals into the high-dimensional control commands required for dexterous manipulation.

Recent developments in deep learning (DL) [6] and reinforcement learning (RL) [7], [8] have enabled significant progress in learning-based algorithms for dexterous manipulation such as in-hand rotation [9], [10], [11], solving Rubik's cube [12] and playing the piano [13], [14]. However, it proves challenging to train RL policies as RL training heavily relies on intricate reward engineering [8] and extensive computational resources [12]. Furthermore, unnatural behaviors with high rewards can emerge from RL training [15]. To address the limitations of RL, some previous works [16], [17], [18], [19], [20], [21] have turned to imitation learning using robot demonstrations collected by teleoperation. While these approaches enhance training efficiency, they demand substantial human efforts to collect diverse robot demonstrations and are, hence, difficult to scale.

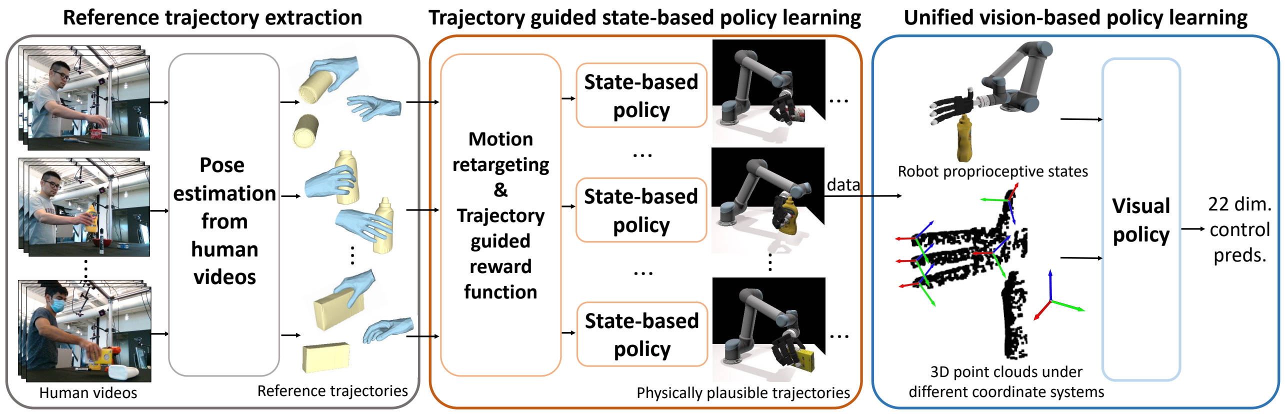
[1]Inria, École normale supérieure, CNRS, PSL Research University, 75005, Paris, France. {`firstname.lastname`}`@inria.fr`
[2]Mohamed bin Zayed University of Artificial Intelligence, Abu Dhabi, United Arab Emirates. {`firstname.lastname`}`@mbzuai.ac.ae`

As human videos capturing hands manipulating objects [22], [23] are abundant and easy to acquire, there is a growing trend in robotics research to utilize human video demonstrations for learning dexterous manipulation skills [24], [25], [26], [27], [28]. For example, DexMV [24] proposes to extract robot and object poses from human videos, and then employs the data to accelerate RL training via demonstration augmented policy gradient [16]. However, since the extracted robot and object poses are noisy, the method needs hundreds of human videos to learn the manipulation of a single object and requires reward engineering for different tasks. Moreover, existing methods [24], [26], [29] often leverage privileged information of objects in policy learning such as ground-truth object CAD models and object poses, which are non-trivial to obtain from visual sensor data in real-world scenarios.

In this work, we propose ViViDex, a new framework making use of human **Vi**deos for **Vi**sion-based **Dex**terous manipulation. ViViDex consists of three modules as illustrated in Figure 1. We first obtain human hand and object trajectories from video demonstrations [24]. Though such trajectories are noisy and not directly usable for robot control, they provide examples of natural hand-object interactions. Our second module then refines the reference trajectories to be physically plausible. We train a state-based policy with RL for each reference trajectory. A novel trajectory-guided reward is proposed to keep pose similarity in the reference trajectory while solving the task. We also augment trajectories during training to generalize to a broader range of object poses beyond the pose in the reference trajectory. Finally, we rollout episodes from optimized state-based policies and distill successful episodes to a unified vision-based policy using no privileged object information. To this end, we propose to improve visual representation by transforming input 3D point clouds into hand-centered coordinate systems. We evaluate ViViDex on three dexterous manipulation tasks *relocation*, *pour* and *placing inside* in simulation and demonstrate significant improvements over the state-of-the-art DexMV [24] while using significantly less human demonstrations. Our visual policy also achieves good performance both in simulation and on a real robot for both seen and unseen objects.

To summarize, our contributions are three-fold:

- We introduce a new framework ViViDex to learn vision-based dexterous manipulation policy from human videos.
- We improve the RL rewards for the state-based policy by imitating trajectories extracted from human videos and propose a novel model architecture for the vision-based policy.
- Extensive simulation and real robot experiments demon-



Reference trajectory extraction
Trajectory guided state-based policy learning
Unified vision-based policy learning
Pose estimation from human videos
Human videos
Reference trajectories
Motion retargeting & Trajectory guided reward function
State-based policy
State-based policy
State-based policy
Physically plausible trajectories
data
Robot proprioceptive states
3D point clouds under different coordinate systems
Visual policy
22 dim. control preds.

Fig. 1: The overall framework of our method for learning dexterous manipulation skills from human videos. It consists of three steps: extraction of reference trajectories from human videos, trajectory-guided state-based policy learning using RL, and vision-based policy learning using either the behavior cloning or the 3D diffusion policy.

strate the effectiveness of our proposed ViViDex approach.

II. Related Work

Dexterous manipulation. Multi-fingered robotic hands enable robots to perform delicate manipulation operations on objects. Previous work addressed dexterous manipulation by trajectory optimization [30], [31], [32], [33], [34], [35], [36] and data-driven learning methods [16], [24], [25], [29], [37], [38], [39], [40], [41], [42], [43]. Trajectory optimization methods usually require well-defined dynamic models for the robot and the manipulated object, which are sometimes hard to obtain in practice. Data-driven methods instead directly train neural policies given robot data. Most of the works [44], [45], [16], [29], [38], [46], [47], [48] use RL algorithms to train policies. Nagabandi *et al.* [47] and Hansen *et al.* [48] propose to improve the sample efficiency in RL by jointly learning the dynamic model and the control policy. To accelerate the convergence of RL, Rajeswaran *et al.* [16] introduce demonstration augmented policy gradients (DAPG) and train RL with expert demonstrations. Instead of manually collecting expert demonstrations, ILAD [29] derives demonstration data via an affordance model [49] and trajectory optimization [50]. Xu *et al.* [38] and Wan *et al.* [46] decompose the dexterous grasping into two sub-tasks: the grasp proposal generation and the goal-conditioned policy training and largely improve the generalization of the learned policy. However, RL is data-inefficient, requires well-designed rewards, and often results in unrealistic object manipulation. In this work, we leverage human videos to improve RL training of state-based policies and then train a unified visual policy that enables object manipulation given 3D point cloud inputs.

Learning robotic manipulation from human videos. Human videos naturally provide rich human hands interactions with diverse objects [51], [52], [53], [54], [55]. Equipping robots with the capacity to acquire manipulation skills by simply watching human videos has been an attractive direction [56], [57], [58], [59], [60]. One line of works [37], [61], [62], [63], [64] aims to learn generic visual representations from large-scale human video data [65] and then makes use of such pre-trained representations to learn control policies. Another line of works [66], [67], [68] focuses on learning reward functions from human videos. Chen *et al.* [66] introduce DVD, a domain-agnostic video discriminator, to learn multi-task reward functions. Alakuijala *et al.* [67] propose a task-agnostic reward function by using unlabeled human videos. Some other works [24], [25], [26], [27], [69] explore how to explicitly leverage human videos to facilitate learning dexterous manipulation skills. Mandikal *et al.* [25], [69] propose to utilize object affordances priors and human grasping priors to improve the dexterous grasping policy. DexMV [24] and DexRepNet [26] extract demonstrations data from human videos and use DAPG algorithm [16] to launch the RL training with demonstrations. PGDM [28] proposes to initialize the robot hand configuration at the given pre-grasp derived from human motions and enable efficient RL training. Compared to prior work [24], [26], our method allows to solve more complex manipulation tasks with fewer videos by using a unified trajectory guided reward learned from videos and automatic trajectory augmentation.

III. Method

This work aims to train vision-based dexterous manipulation policies. However, jointly learning the visual representation and the control policy is challenging. Therefore, we first train the state-based policy and then distill the learned policy to a vision-based policy. To train the state-based policy, as the pure RL training is ineffective [24], [26], [29], we extract reference trajectories from human videos and propose trajectory-guided RL to improve the performance. In the following, we first present the reference trajectory generation in Section III-A. Then, we introduce our state-based policy learning algorithm in Section III-B. Finally, we describe the visual policy and its training in Section III-C.

A. Reference Trajectory Extraction

To guide our policy training, we extract hand and object poses from video demonstrations. The poses and shapes of hands are represented by MANO [70] and result in 3D locations of joints $\psi_h \in \mathbb{R}^{21\times3}$. To account for differences in geometry, we retarget the human hand pose to the robot hand pose. Following [24], [71], [72], we formulate the hand

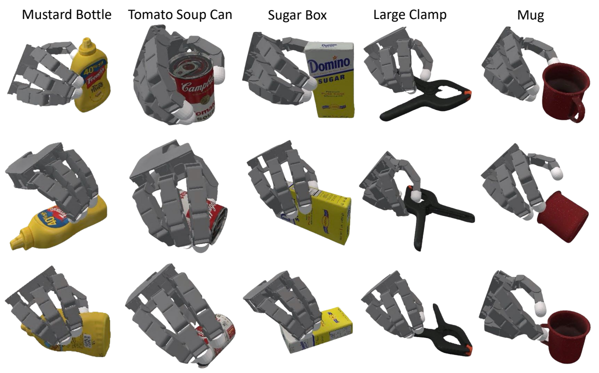


Fig. 2: Motion retargeting results for the Allegro hand and objects under different poses for selected DexYCB videos.

motion retargeting from a video of length T as an optimization problem and define its objective function as:

$$\min_{\boldsymbol{q}_r^t} \sum_{t=1}^{T} \left\| \hat{\boldsymbol{x}}_{rj}^t(\boldsymbol{q}_r^t) - \boldsymbol{\psi}_{hj}^t \right\|_2^2 + \alpha \left\| \boldsymbol{q}_r^t - \boldsymbol{q}_r^{t-1} \right\|_2^2, \tag{1}$$

where $\boldsymbol{q}_r^t$ represents robot joint rotation angles. $\boldsymbol{\psi}_{hj}^t$ denotes human hand tip and middle phalanx positions. We solve their robot counterparts $\hat{\boldsymbol{x}}_{rj}^t$ via forward kinematics according to given $\boldsymbol{q}_r^t$. The first term aims to minimize the difference between the 3D locations of robot joints to those in the human hand. We also use a regularization term to avoid sudden changes in the robot pose, where we empirically set α to 4e-3 and $\boldsymbol{q}_r^0$ to the mean pose of its motion limits. We use the NLopt solver [73] for the optimization and port robot and object motions into the simulator to generate reference trajectories, which are visually plausible but not physically plausible. Figure 2 illustrates results of the motion retargeting for the Allegro robot hand.

B. *State-based Policy*

We train a state-based policy with RL to recover physically plausible trajectories, where the above generated reference trajectories are employed in the reward function to guide robot hand and object motions. The network architecture for the state-based policy consists of actor and critic MLPs [74]. It takes both robot and object states as inputs and predicts the robot control commands. In the following, we will describe our reward functions and the trajectory augmentation for RL.

Trajectory-guided reward functions. During RL training, we propose to divide the reference trajectory into two stages: the pre-grasp stage and the manipulation stage. However, at test time, the policy executes without stage distinctions.

During the pre-grasp stage, the robot hand needs to approach the object without making physical contacts. We require that the robot approaches the object in a similar way as humans and define the following reward function as:

$$R_p = \sum_{t=1}^{T_p} 10 \cdot \exp(-10 \cdot \left\| \boldsymbol{x}_{rt}^t(\boldsymbol{q}_r^t) - \hat{\boldsymbol{x}}_{rt}^t \right\|_2^2), \tag{2}$$

where T_p is the length of pre-grasp steps, and $\hat{\boldsymbol{x}}_{rt}^t$ denotes the robot finger tip positions at the timestep t in the reference trajectory. $\boldsymbol{x}_{rt}^t$ is the current robot finger tip positions.

When the robot successfully reaches its pre-grasp configuration, the episode starts its manipulation stage. In this stage, the robot aims to manipulate the object and bring it to the desired target configuration. Here, we define the reward function to constrain the robot and object motions jointly:

$$R_m = \sum_{t=T_p+1}^{T_r} \lambda_1 R_m^h + \lambda_2 R_m^o + \lambda_3 \mathbb{1}_{\text{cont}} + \lambda_4 \mathbb{1}_{\text{lift}}, \tag{3}$$

where T_r is the length of the reference trajectory. The first term R_m^h constrains the hand motions similar to (2). The second term $R_m^o = \exp(-\alpha_1(\left\| \boldsymbol{x}_o^t - \hat{\boldsymbol{x}}_o^t \right\|_2^2 + \alpha_2 \phi(\boldsymbol{\theta}_o^t,\ \hat{\boldsymbol{\theta}}_o^t)))$ constrains the object motions, where $\boldsymbol{x}_o^t$ and $\hat{\boldsymbol{x}}_o^t$ are the current object position and its reference at timestep t. $\phi(\cdot)$ computes the angular distance between the current object orientation with its reference $\hat{\boldsymbol{\theta}}_o^t$. The third term computes the number of finger tips in contact with the object. The forth term assigns bonus points when the object is lifted off the table. We empirically set $\lambda_1 = 4, \lambda_2 = 10, \lambda_3 = 0.5, \alpha_1 = 50$ and $\alpha_2 = 0.1$. Since our reward function is primarily derived from trajectories in videos, it can be applied across various manipulation tasks, reducing the need for task-specific reward engineering as in DexMV [24].

Reference trajectory augmentation. Though the state-based policy can successfully imitate the hand and object motions from a video, it remains challenging to generalize to different initial object positions, rotations and target positions. To make our policy applicable to different initial and target object configurations, we introduce our reference trajectory augmentation strategies during RL training. Specifically, we randomly set initial object positions or rotations and then transform the whole reference trajectory accordingly. In order to further increase the diversity of target object positions, we augment the original object trajectory through interpolations between the original last object pose and the target pose. We similarly interpolate hand motions and use proposed reward functions to train the state-based policy.

C. *The Vision-based Policy*

Though our state-based policy can work well for different manipulation tasks, it requires robot proprioceptive states and object states as inputs. However, reliably estimating object states is often not trivial in practice. To alleviate this issue, we propose a vision-based policy that only takes robot states and 3D scene point clouds as inputs. To generate training data for the visual policy, we rollout the optimized state-based policy and generate diverse physically plausible trajectories. During the rollout process, we compute the 3D point clouds $\mathrm{PC_w} \in \mathbb{R}^{N \times 3}$ from the depth camera, where $\mathrm{PC_w}$ is represented under the world coordinate system (*i.e.*, the center of the table) and N is the number of points.

Coordinate transformation. As shown in Figure 1, inspired by [75], we propose to transform $\mathrm{PC_w}$ into the desired target coordinate system $\mathrm{PC_t}$, which makes the model more aware of the target position for control predictions. To capture dense interaction features between the robot and the object, we additionally transform $\mathrm{PC_w}$ into different robot joint coordinate systems (*i.e.*, palm and finger tips). Finally, we combine point clouds representations under different coordinate systems $\mathrm{PC} \in \mathbb{R}^{N \times 3(j+3)}$ and feed them into PointNet [76], where j is the number of finger tips. Our visual policy predicts control commands based on extracted visual features and robot proprioceptive states.

Training. We train our visual policy using either the behavior cloning (BC) or the recently proposed 3D diffusion policy [77], [78] from generated physically plausible trajectories. The BC model directly takes transformed point clouds and robot states as inputs and predicts robot actions. The diffusion policy employs the extracted 3D features from PointNet [76] as the global condition for the denoising model and recovers actions from Gaussian noise. We train both models with $\ell 2$ loss between the predicted and ground-truth actions.

IV. Experiments

We conduct extensive experiments and evaluate state-based and visual policies learned by our approach on three manipulation tasks for five seen and ten unseen objects.

A. Experimental Setting

Video dataset. The DexYCB [51] dataset contains human demonstration videos of hand-object interactions. Following [24], we focus on five objects including *mustard bottle*, *tomato soup can*, *sugar box*, *large clamp* and *mug*.

For each object, we choose three videos which are most similar to those used in DexMV[1] in our experiments. Figure 2 presents motion retargeting results for these videos. Two evaluation protocols are used for benchmarking the performance of policies. Protocol #1 trains the policy for each object separately and uses initial object poses from the first row of Figure 2. For Protocol #2, we train a unified policy for all five objects and adopt three poses for each object shown in Figure 2. Furthermore, we evaluate the performance on ten unseen objects, namely *master chef can*, *tuna fish can*, *pudding box*, *gelatin box*, *potted meat can*, *banana*, *pitcher base*, *bleach cleanser*, *wood block* and *foam brick*.

Simulation environment. We follow previous works [16], [24], [34] by using the the Adroit robot hand and the MuJoCo simulator for fair comparison. However, this setup is less realistic, as the robotic hand is not attached to an arm and can move freely. To address this, we attach an Allegro robot hand to a UR5 arm, mirroring our actual hardware configuration. We use the SAPIEN [79] simulator for this setup, which simplifies creating this more realistic environment and allows to speed up simulation. Unless specified otherwise, the Adroit hand in MuJoCo is used for benchmarking against state-of-the-art methods, while the Allegro hand in SAPIEN is employed for training visual policies and performing ablation studies.

Manipulation tasks. We follow [24] to evaluate on three tasks. The first ***relocate*** task requires the robot to move an object to a target position. The ***pour*** task aims to grasp a mug filled with particles and pour the particles into a container. The success rate is measured by the percentage of particles in the container. The ***place inside*** task aims to grasp a banana and place it into a mug. The success rate is measured by the percentage of the banana mesh in the mug.

[1]The videos in DexMV are not publicly released.

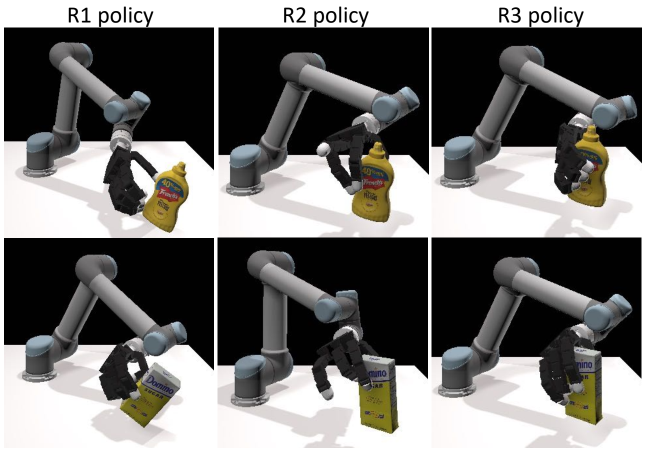


Fig. 3: Qualitative comparison of state-based policies using different rewards for Protocol #1 and the Allegro hand. R1 (w/o hand reward in pre-grasp) leads to unstable grasps. R2 (w/o hand reward in manipulation) results in unnatural hand actions. Our proposed approach R3 uses hand rewards at both stages and achieves the best performance.

TABLE I: Performance of our state-based policy using different hand rewards (Allegro hand, Protocol #1). It follows the extracted human hand trajectory to manipulate objects (R1), reach the pre-grasp location (R2) or combine both (R3).

	Hand Reward		$\mathrm{E_o} \downarrow$	$\mathrm{E_h} \downarrow$	$\mathrm{SR_o} \uparrow$	$\mathrm{SR_h} \uparrow$	$\mathrm{SR_3} \uparrow$
	Pre-grasp	Manipulation					
R1	×	✓	0.048	0.21	0.35	0.00	0.00
R2	✓	×	0.0033	0.077	0.95	0.32	1.00
R3	✓	✓	**0.0019**	**0.032**	**0.97**	**0.79**	**1.00**

Evaluation metrics. The success rate (SR) is the major metric for the above three tasks. For the *relocate* task, we adopt the 10cm threshold (SR_{10}) following [24], [29] to define the success rate. To measure the accuracy more precisely, we add a more rigorous threshold of 3cm and compute the success rate SR_3. For the state-based policy, we additionally evaluate how well the physically plausible trajectory matches the reference trajectory using four metrics [28]: $\mathrm{E_o}$ computes the average object position error between the current object trajectory and its reference trajectory over time; $\mathrm{E_h}$ computes the average finger tips position error between the current robot hand trajectory and its counterpart over time; $\mathrm{SR_o}$ reports the fraction of timesteps where $\mathrm{E_o}$ is below 1cm; $\mathrm{SR_h}$ computes the fraction of timesteps where $\mathrm{E_h}$ is lower than 5cm.

Implementation details. We use PPO [74] to optimize the state-based policy. The RL training takes around 2 hours on a single A100 GPU for each video. We rollout 100 successful trajectories from the state-based policy for each video, where trajectories differ in initial object positions, orientations and target positions. During rollouts, we render 3D scene point clouds from the depth camera. These data are used to train our visual policy, which takes robot joint positions and 3D point clouds as inputs. Training our visual policy on a single A100 GPU takes around 10 hours using behavior cloning

TABLE II: Comparison with state-of-the-art methods on *relocate* task for state-based policies. We optimize a state-based policy using a single human video for each object (S6, S7, S8) (Protocol #1), while previous methods usually need about 100 videos for each object (S2-S4). * indicates that we re-train the method from [24] with 20 DexYCB videos for each object.

	Methods	Robot Hands	Training algorithms	Num. videos	Mustard bottle		Tomato can		Sugar box		Large clamp		Mug		Avg.	
					SR_{10}	SR_3	SR_{10}	SR_3	SR_{10}	SR_3	SR_{10}	SR_3	SR_{10}	SR_3	SR_{10}	SR_3
S1	DexMV [24]	Adroit	TRPO [80]	0	0.06	-	0.67	-	0.00	-	0.51	-	0.49	-	0.35	-
S2	DexMV [24]	Adroit	SOIL [81]	97	0.33	-	0.98	-	0.67	-	0.89	-	0.71	-	0.72	-
S3	DexMV [24]	Adroit	GAIL+ [82]	97	0.06	-	0.66	-	0.00	-	0.52	-	0.53	-	0.50	-
S4	DexMV [24]	Adroit	DAPG [16]	97	0.93	-	1.00	-	0.00	-	1.00	-	1.00	-	0.79	-
S5	DexMV [24]*	Adroit	DAPG [16]	20	1.00	0.63	1.00	0.56	0.09	0.00	0.08	0.00	0.02	0.00	0.44	0.24
S6	Ours	Adroit	PPO [74]	1	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
S7	Ours	Allegro	PPO [74]	1	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**
S8	Ours w/o rot.	Allegro	PPO [74]	1	0.85	0.85	1.00	0.99	0.99	0.97	0.98	0.95	0.97	0.94	0.96	0.94

TABLE III: Performance of visual policies for each object on *relocate* task with SR_3 using Protocol #1 and Allegro hand.

	Algorithms	#Points	Mb.	Tc.	Sb.	Lc.	Mug	Avg.
V1	BC	512	0.99	0.82	0.82	0.78	0.90	0.86
V2	BC	2048	1.00	**1.00**	0.87	0.91	**1.00**	0.96
V3	Diffusion	2048	**1.00**	0.97	**0.98**	**0.99**	0.99	**0.99**

and about 20 hours using the diffusion policy, both based on data from 15 videos. For testing, we run 300 episodes with different initial configurations and report the average success rate. The length of a single episode is 60, 80 and 100 for the *relocate* task, *place inside* task and *pour* task, respectively. The real robot takes around 1 minute to execute an episode.

B. Evaluation of the State-based Policy on the Relocate Task

We evaluate the state-based policy on the *relocate* task using both the Adroit and Allegro hands. We follow Protocol #1 to train the state-based policy then evaluate each policy on the same object with novel placement of the object. Specifically, we randomly change the initial object position and rotation around z-axis for each testing episodes.

1) Ablation Studies: We first perform ablations to validate the importance of our hand rewards at the pre-grasp and the manipulation stages in state-based policy learning. Table I presents the averaged performance over all objects for policies with different reward functions for the Allegro hand. The policy in R1 uses the proposed hand reward R_m^h in the manipulation stage but follows the previous reward function [24], [29], [16] in the pre-grasp stage which is to minimize the distance between the robot and the object. It achieves poor performance due to the lack of guidance from the human hand trajectory to approach the object. As shown in the first column of Figure 3, the policy has difficulties in arriving at a plausible pre-grasp configuration to stably lift the object. The policy in R2 employs the proposed reward function R_p in the pre-grasping stage but follows PGDM [28] (*i.e.*, R_m^o and $\mathbb{1}_{\text{lift}}$) in the manipulation stage. The reward function in PGDM constrains the object motions to be close to the reference trajectory but does not add any constraints on the hand motions, which results in unnatural robot actions as shown in the second column of Figure 3. Our proposed policy in R3 utilizes hand rewards in both stages, which outperforms R1 and R2 on all the metrics and produces more robust and realistic grasps as shown in the last column of Figure 3.

2) State-of-the-art comparison: Table II presents the quantitative comparison of our state-based policy and state-of-the-art models [24]. S1 to S4 denote four model variants reported in [24] for the Adroit hand. As their training data is not released, we re-train their best model DAPG [16] (S4) using 20 DexYCB videos and report the results in S5 for fair comparison. We can see that the performance drops significantly for some objects, *i.e.*, *large clamp* and *mug*. This suggests that their approach needs a large number of videos to effectively learn how to handle objects with complex shapes. Our proposed policy in S6 significantly outperforms all previous methods despite only using one training video per object. Though S4 also performs well for *mustard bottle*, *tomato soup can*, *large clamp* and *mug*, it easily knocks down thin objects (*e.g.*, *sugar box*) when the robot approaches the object and achieves poor performance. Our hand reward addresses this problem by mimicking the human pre-grasping trajectory. As shown in S7, we achieve the same performance with the Allegro hand. S8 is a variant of our model S7 without rotation augmentation in the z-axis during training. The performance only drops insignificantly when tested for such rotations. This demonstrates the robustness of our method.

C. Evaluation of the Visual Policy on the Relocate Task

Here, we evaluate our proposed visual policy on the *relocate* task and present a detailed experimental analysis.

1) Learning from a single video per object: We first investigate whether replacing the ground-truth object state in the state-based policy with visual point cloud inputs can affect the performance. For each video, we follow Protocol #1 to train a separate visual policy using the rollout data obtained from the state-based policy S7 in Table II, and evaluate under novel object placements as in the evaluation of the state-based policy. The results are presented in Table III. Compared with V1 that uses 512 points, V2 samples 2048 points and improves the overall performance, which shows the importance of point clouds densities. By comparing V2 and V3, the 3D diffusion policy [78] can achieve more robust performance than the plain behavior cloning algorithm. Our visual policies only perform slightly worse than our state-based policies.

2) Learning a unified multi-object visual policy from multiple videos: Finally, we adopt Protocol #2 and train a single policy for all objects with the Allegro hand and report its performance in Table IV. We test the policy under three

TABLE IV: Performance of unified visual policies on *relocate* task under SR_3 using Protocol #2 and Allegro hand. We compare the performance using different number of human videos and point clouds under different coordinate systems.

	Algo.	Vid.	Coor. sys.		Seen						Unseen avg.
			target	hand	Mb.	Tc.	Sb.	Lc.	Mug	Avg.	
V4	BC	1×5	×	×	0.33	0.33	0.33	0.30	0.42	0.34	0.31
V5	BC	2×5	×	×	0.55	0.57	0.36	0.56	0.66	0.54	0.33
V6	BC	3×5	×	×	0.69	0.80	0.83	0.86	0.88	0.81	0.38
V7	BC	3×5	✓	×	0.93	0.92	0.99	0.93	0.99	0.95	0.37
V8	BC	3×5	✓	✓	0.93	0.97	0.97	0.98	**1.00**	0.97	0.41
V9	Diff.	3×5	✓	✓	**0.97**	**0.99**	**1.00**	**0.99**	0.99	**0.99**	**0.50**

TABLE V: Comparison with state-of-the-art methods on the *pour* and the *place inside* tasks using the Adroit hand. Our approach uses a single human video for each task, while previous methods need more than 91 human videos.

	Methods	Training algorithms	Num. videos	Pour success rate	Place inside success rate
L1	DexMV [24]	TRPO [80]	0 / 0	0.01	0.03
L2	DexMV [24]	SOIL [81]	101 / 91	0.04	0.28
L3	DexMV [24]	GAIL+ [82]	101 / 91	0.03	0.16
L4	DexMV [24]	DAPG [16]	101 / 91	0.27	0.31
L5	Ours	PPO [74]	1 / 1	0.97	0.68
L6	Ours	Diffusion	1 / 1	**0.97**	**0.68**

initial poses for each object shown in Figure 2. From V4 to V6, we gradually increase the number of videos for each object from one to three and observe a significant improvement in performance. Then, V7 additionally transforms the 3D point clouds into the target coordinate system. As a result, the model becomes more aware of its target and largely improves the average performance for seen objects from 81% to 95%. To further incorporate fine-grained hand-object interaction features, we further transform 3D points into hand joint coordinate systems in V8. Benefiting from extracting rich hand-object interaction features, V8 achieves even better performance than V7. Different from V8, V9 trains the visual model using 3D diffusion policy and achieves an average success rate of 99% under the challenging test scenario.

To investigate the generalization of our visual policies, we evaluate their performance using three different initial poses for ten unseen YCB objects. In Figure IV from V4 to V6, our model better generalizes to novel objects when using more videos. V8 incorporates fine-grained interaction features and enhances generalization. Compared with the behavior cloning, V9 significantly improves the unseen success rate from 41% to 50% by using the 3D diffusion policy, which demonstrates better robustness and generalization abilities.

D. Evaluation of our policy on Pour and Place inside Tasks

We further evaluate our approach on the *pour* and *place inside* tasks and report results in Table V. Rows L1-L4 show results for state-based policies [24] using the Adroit hand as well as 91 and 101 video demonstrations for *pour* and *place inside* tasks respectively. While we use only single human video for training, our state-based policy L5 for the Adroit hand achieves 97% and 68% success rates on these two tasks respectively, significantly outperforming 27% and

TABLE VI: Success rate of visual policies on *relocate* task using the real robot. We evaluate 10 episodes for each object.

	Methods	Unified policy	Seen						Unseen Avg.
			Mb.	Tc.	Sb.	Lc.	Mug	Avg.	
R1	BC	×	10/10	8/10	9/10	8/10	9/10	0.88	-
R2	BC	✓	9/10	7/10	7/10	5/10	8/10	0.72	0.58
R3	Diffusion	✓	10/10	7/10	8/10	7/10	8/10	0.80	0.68

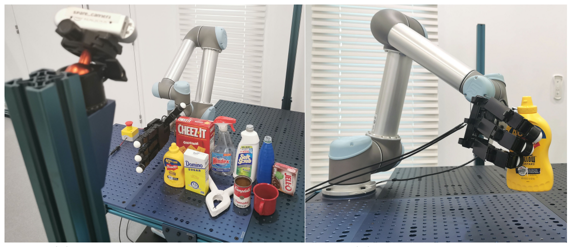

(a) Real-world experiments setup (b) Visual policy execution

Fig. 4: Illustrations of our real-world robot experimental setup and the performance of our proposed ViViDex algorithm.

31% success rates corresponding to the best models of [24]. By taking advantage of high-quality trajectories generated by L5, our visual policy L6 also achieves high performance.

E. Real world experiments

To further demonstrate the advantages of ViViDex, we evaluate its performance on real-world dexterous manipulation. As shown in Figure 4(a), we use a UR5 robotic arm equipped with an Allegro hand and a single RealSense D435 RGB-D camera. Since point clouds for the real-world experiments are noisier than those in simulation, we use the state-based policy to collect data in the real world and then learn a unified visual policy based on real data. We run state-based policies in the simulator for specific object locations and run them on the real robot by placing the object in the same locations. This allows us to collect 3D scene point clouds and robot states for each execution step and build the real-robot training dataset. We collect five robot trajectories for each object and train the visual policy using our collected real-robot data. As shown in Figure 4(b), our visual policy can then manipulate the object. We summarize the quantitative results on the real robot in Table VI. We adopt the initial pose for seen objects from the first row of Figure 2 and include five unseen objects for evaluation: *cracker box*, *spray bottle*, *bleach cleanser*, *water bottle* and *pudding box*. R1 trains five policies separately and achieve an average success rate of 88%. R2 and R3 learn unified policies for five objects and demonstrate the ability to grasp unseen objects. The diffusion policy R3 also achieves better performance than BC in real experiments.

V. Conclusion

We introduce ViViDex, a new framework for learning vision-based dexterous manipulation from human videos. Our approach extracts reference trajectories from human videos and uses them as a reward in training state-based policies with RL. This generates diverse physically plausible trajectories. We rollout these trajectories for training a visual policy, which takes as inputs robot proprioceptive states and 3D point clouds. To enhence visual policies, we further transforms 3D point clouds into hand-centered coordinate systems. We conduct extensive ablation experiments to validate the effectiveness of ViViDex on different simulators and a real robot. We show that our visual policies outperform state-of-the-art accuracy on different tasks by a significant margin and generalize to unseen objects. Future work aims at leveraging internet videos for acquiring more general dexterous manipulation skills and investigating advanced 3D pose estimation algorithms.

Acknowledgement: We thank Ricardo Garcia Pinel for help with the camera calibration, Yuzhe Qin for clarifications about DexMV and suggestions on the real robot depolyment. This work was granted access to the HPC resources of IDRIS under the allocation AD011013147 made by GENCI. It was funded in part by the French government under management of Agence Nationale de la Recherche as part of the "France 2030" program, reference ANR-23-IACL-0008 (PR[AI]RIE-PSAI projet), and the ANR project VideoPredict (ANR-21-FAI1-0002-01). Cordelia Schmid would like to acknowledge the support by the Körber European Science Prize.

References

[1] S. James, Z. Ma, D. R. Arrojo, and A. J. Davison, "RLBench: The robot learning benchmark & learning environment," *RAL*, 2020.

[2] P.-L. Guhur, S. Chen, R. G. Pinel, M. Tapaswi, I. Laptev, and C. Schmid, "Instruction-driven history-aware policies for robotic manipulations," in *CoRL*, 2023.

[3] D. Driess, F. Xia, M. S. Sajjadi, C. Lynch, A. Chowdhery, B. Ichter, A. Wahid, J. Tompson, Q. Vuong, T. Yu *et al.*, "PaLM-E: An embodied multimodal language model," in *ICML*, 2023.

[4] S. Chen, R. Garcia, C. Schmid, and I. Laptev, "PolarNet: 3D point clouds for language-guided robotic manipulation," in *CoRL*, 2023.

[5] M. T. Mason, "Toward robotic manipulation," *Annual Review of Control, Robotics, and Autonomous Systems*, 2018.

[6] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, 2015.

[7] L. P. Kaelbling, M. L. Littman, and A. W. Moore, "Reinforcement learning: A survey," *Journal of artificial intelligence research*, 1996.

[8] R. S. Sutton, "Reinforcement learning: An introduction," *A Bradford Book*, 2018.

[9] H. Qi, A. Kumar, R. Calandra, Y. Ma, and J. Malik, "In-hand object rotation via rapid motor adaptation," in *CoRL*, 2022.

[10] Z.-H. Yin, B. Huang, Y. Qin, Q. Chen, and X. Wang, "Rotating without seeing: Towards in-hand dexterity through touch," in *RSS*, 2023.

[11] H. Qi, B. Yi, S. Suresh, M. Lambeta, Y. Ma, R. Calandra, and J. Malik, "General in-hand object rotation with vision and touch," in *CoRL*, 2023.

[12] I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas *et al.*, "Solving rubik's cube with a robot hand," *arXiv*, 2019.

[13] H. Xu, Y. Luo, S. Wang, T. Darrell, and R. Calandra, "Towards learning to play piano with dexterous hands and touch," in *IROS*, 2022.

[14] K. Zakka, P. Wu, L. Smith, N. Gileadi, T. Howell, X. B. Peng, S. Singh, Y. Tassa, P. Florence, A. Zeng *et al.*, "RoboPianist: Dexterous piano playing with deep reinforcement learning," in *CoRL*, 2023.

[15] J. Merel, Y. Tassa, D. TB, S. Srinivasan, J. Lemmon, Z. Wang, G. Wayne, and N. Heess, "Learning human behaviors from motion capture by adversarial imitation," *arXiv*, 2017.

[16] A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov, and S. Levine, "Learning complex dexterous manipulation with deep reinforcement learning and demonstrations," in *RSS*, 2018.

[17] Y. Qin, W. Yang, B. Huang, K. Van Wyk, H. Su, X. Wang, Y.-W. Chao, and D. Fox, "AnyTeleop: A general vision-based dexterous robot arm-hand teleoperation system," in *RSS*, 2023.

[18] S. P. Arunachalam, S. Silwal, B. Evans, and L. Pinto, "Dexterous imitation made easy: A learning-based framework for efficient dexterous manipulation," in *ICRA*, 2023.

[19] S. P. Arunachalam, I. Güzey, S. Chintala, and L. Pinto, "Holo-Dex: Teaching dexterity with immersive mixed reality," in *ICRA*, 2023.

[20] C. Wang, H. Shi, W. Wang, R. Zhang, L. Fei-Fei, and C. K. Liu, "DexCap: Scalable and portable mocap data collection system for dexterous manipulation," in *RSS*, 2024.

[21] S. Yang, M. Liu, Y. Qin, D. Runyu, L. Jialong, X. Cheng, R. Yang, S. Yi, and X. Wang, "ACE: A cross-platfrom visual-exoskeletons for low-cost dexterous teleoperation," in *CoRL*, 2024.

[22] L. Yang, K. Li, X. Zhan, F. Wu, A. Xu, L. Liu, and C. Lu, "OakInk: A large-scale knowledge repository for understanding hand-object interaction," in *CVPR*, 2022.

[23] Y. Liu, Y. Liu, C. Jiang, K. Lyu, W. Wan, H. Shen, B. Liang, Z. Fu, H. Wang, and L. Yi, "HOI4D: A 4D egocentric dataset for category-level human-object interaction," in *CVPR*, 2022.

[24] Y. Qin, Y.-H. Wu, S. Liu, H. Jiang, R. Yang, Y. Fu, and X. Wang, "DexMV: Imitation learning for dexterous manipulation from human videos," in *ECCV*, 2022.

[25] P. Mandikal and K. Grauman, "DexVIP: Learning dexterous grasping with human hand pose priors from video," in *CoRL*, 2022.

[26] Q. Liu, Y. Cui, Q. Ye, Z. Sun, H. Li, G. Li, L. Shao, and J. Chen, "DexRepNet: Learning dexterous robotic grasping network with geometric and spatial hand-object representations," in *IROS*, 2023.

[27] K. Shaw, S. Bahl, and D. Pathak, "VideoDex: Learning dexterity from internet videos," in *CoRL*, 2022.

[28] S. Dasari, A. Gupta, and V. Kumar, "Learning dexterous manipulation from exemplar object trajectories and pre-grasps," in *ICRA*, 2023.

[29] Y.-H. Wu, J. Wang, and X. Wang, "Learning generalizable dexterous manipulation from human grasp affordance," in *CoRL*, 2022.

[30] A. Bicchi and V. Kumar, "Robotic grasping and contact: A review," in *ICRA*, 2000.

[31] L. Han and J. C. Trinkle, "Dextrous manipulation by rolling and finger gaiting," in *ICRA*, 1998.

[32] D. Rus, "In-hand dexterous manipulation of piecewise-smooth 3D objects," *IJRR*, 1999.

[33] I. Mordatch, Z. Popović, and E. Todorov, "Contact-invariant optimization for hand manipulation," in *SIGGRAPH*, 2012.

[34] A. Wu, M. Guo, and C. K. Liu, "Learning diverse and physically feasible dexterous grasps with generative model and bilevel optimization," *arXiv*, 2022.

[35] W. Yang and W. Jin, "ContactSDF: Signed distance functions as multi-contact models for dexterous manipulation," *arXiv*, 2024.

[36] X. Zhu, J. Ke, Z. Xu, Z. Sun, B. Bai, J. Lv, Q. Liu, Y. Zeng, Q. Ye, C. Lu *et al.*, "Diff-LfD: Contact-aware model-based learning from visual demonstration for robotic manipulation via differentiable physics-based simulation and rendering," in *CoRL*, 2023.

[37] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta, "R3M: A universal visual representation for robot manipulation," in *CoRL*, 2022.

[38] Y. Xu, W. Wan, J. Zhang, H. Liu, Z. Shan, H. Shen, R. Wang, H. Geng, Y. Weng, J. Chen *et al.*, "UniDexGrasp: Universal robotic dexterous grasping via learning diverse proposal generation and goal-conditioned policy," in *CVPR*, 2023.

[39] T. Chen, J. Xu, and P. Agrawal, "A system for general in-hand object re-orientation," in *CoRL*, 2021.

[40] C. Bao, H. Xu, Y. Qin, and X. Wang, "DexArt: Benchmarking generalizable dexterous manipulation with articulated objects," in *CVPR*, 2023.

[41] T. Lin, Y. Zhang, Q. Li, H. Qi, B. Yi, S. Levine, and J. Malik, "Learning visuotactile skills with two multifingered hands," *arXiv*, 2024.

[42] Y. Han, M. Xie, Y. Zhao, and H. Ravichandar, "On the utility of koopman operator theory in learning dexterous manipulation skills," in *CoRL*, 2023.

[43] J. Wang, Y. Qin, K. Kuang, Y. Korkmaz, A. Gurumoorthy, H. Su, and X. Wang, "CyberDemo: Augmenting simulated human demonstration for real-world dexterous manipulation," in *CVPR*, 2024.

[44] Y. Yuan, H. Che, Y. Qin, B. Huang, Z.-H. Yin, K.-W. Lee, Y. Wu, S.-C. Lim, and X. Wang, "Robot synesthesia: In-hand manipulation with visuotactile sensing," in *ICRA*, 2024.

[45] T. Chen, M. Tippur, S. Wu, V. Kumar, E. Adelson, and P. Agrawal, "Visual dexterity: In-hand reorientation of novel and complex object shapes," *Science Robotics*, 2023.

[46] W. Wan, H. Geng, Y. Liu, Z. Shan, Y. Yang, L. Yi, and H. Wang, "UniDexGrasp++: Improving dexterous grasping policy learning via geometry-aware curriculum and iterative generalist-specialist learning," in *ICCV*, 2023.

[47] A. Nagabandi, K. Konolige, S. Levine, and V. Kumar, "Deep dynamics models for learning dexterous manipulation," in *CoRL*, 2020.

[48] N. Hansen, Y. Lin, H. Su *et al.*, "MoDem: Accelerating visual model-based reinforcement learning with demonstrations," in *ICLR*, 2023.

[49] H. Jiang, S. Liu, J. Wang, and X. Wang, "Hand-object contact consistency reasoning for human grasps generation," in *ICCV*, 2021.

[50] R. Rubinstein, "The cross-entropy method for combinatorial and continuous optimization," *Methodology and computing in applied probability*, 1999.

[51] Y.-W. Chao, W. Yang, Y. Xiang, P. Molchanov, A. Handa, J. Tremblay, Y. S. Narang, K. Van Wyk, U. Iqbal, S. Birchfield *et al.*, "DexYCB: A benchmark for capturing hand grasping of objects," in *CVPR*, 2021.

[52] Y. Hasson, G. Varol, D. Tzionas, I. Kalevatykh, M. J. Black, I. Laptev, and C. Schmid, "Learning joint reconstruction of hands and manipulated objects," in *CVPR*, 2019.

[53] Z. Chen, Y. Hasson, C. Schmid, and I. Laptev, "AlignSDF: Pose-Aligned signed distance fields for hand-object reconstruction," in *ECCV*, 2022.

[54] Z. Chen, S. Chen, C. Schmid, and I. Laptev, "gSDF: Geometry-Driven signed distance functions for 3D hand-object reconstruction," in *CVPR*, 2023.

[55] Y. Ye, A. Gupta, and S. Tulsiani, "What's in your hands? 3D reconstruction of generic objects in hands," in *CVPR*, 2022.

[56] L. Smith, N. Dhawan, M. Zhang, P. Abbeel, and S. Levine, "AVID: Learning multi-stage tasks via pixel-level translation of human videos," in *RSS*, 2020.

[57] K. Zakka, A. Zeng, P. Florence, J. Tompson, J. Bohg, and D. Dwibedi, "XIRL: Cross-embodiment inverse reinforcement learning," in *CoRL*, 2021.

[58] F. Ebert, Y. Yang, K. Schmeckpeper, B. Bucher, G. Georgakis, K. Daniilidis, C. Finn, and S. Levine, "Bridge data: Boosting generalization of robotic skills with cross-domain datasets," in *RSS*, 2022.

[59] K. Schmeckpeper, O. Rybkin, K. Daniilidis, S. Levine, and C. Finn, "Reinforcement learning with videos: Combining offline observations with interaction," in *CoRL*, 2020.

[60] L. Shao, T. Migimatsu, Q. Zhang, K. Yang, and J. Bohg, "Concept2Robot: Learning manipulation concepts from instructions and human demonstrations," in *RSS*, 2020.

[61] T. Xiao, I. Radosavovic, T. Darrell, and J. Malik, "Masked visual pre-training for motor control," *arXiv*, 2022.

[62] I. Radosavovic, T. Xiao, S. James, P. Abbeel, J. Malik, and T. Darrell, "Real-world robot learning with masked visual pre-training," in *CoRL*, 2023.

[63] Y. Ze, Y. Liu, R. Shi, J. Qin, Z. Yuan, J. Wang, and H. Xu, "H-InDex:visual reinforcement learning with hand-informed representations for dexterous manipulation," in *NeurIPS*, 2023.

[64] E. Chane-Sane, C. Schmid, and I. Laptev, "Learning video-conditioned policies for unseen manipulation tasks," in *ICRA*, 2023.

[65] K. Grauman, A. Westbury, E. Byrne, Z. Chavis, A. Furnari, R. Girdhar, J. Hamburger, H. Jiang, M. Liu, X. Liu *et al.*, "Ego4D: Around the world in 3,000 hours of egocentric video," in *CVPR*, 2022.

[66] A. S. Chen, S. Nair, and C. Finn, "Learning generalizable robotic reward functions from" in-the-wild" human videos," in *RSS*, 2021.

[67] M. Alakuijala, G. Dulac-Arnold, J. Mairal, J. Ponce, and C. Schmid, "Learning reward functions for robotic manipulation by observing humans," in *ICRA*, 2023.

[68] H. Xiong, Q. Li, Y.-C. Chen, H. Bharadhwaj, S. Sinha, and A. Garg, "Learning by watching: Physical imitation of manipulation skills from human videos," in *IROS*, 2021.

[69] P. Mandikal and K. Grauman, "Learning dexterous grasping with object-centric visual affordances," in *ICRA*, 2021.

[70] J. Romero, D. Tzionas, and M. J. Black, "Embodied Hands: Modeling and capturing hands and bodies together," *TOG*, 2017.

[71] A. Handa, K. Van Wyk, W. Yang, J. Liang, Y.-W. Chao, Q. Wan, S. Birchfield, N. Ratliff, and D. Fox, "DexPilot: Vision-based teleoperation of dexterous robotic hand-arm system," in *ICRA*, 2020.

[72] D. Antotsiou, G. Garcia-Hernando, and T.-K. Kim, "Task-oriented hand motion retargeting for dexterous manipulation imitation," in *ECCV*, 2018.

[73] G. J. Steven, "The nlopt nonlinear-optimization package," 2019.

[74] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv*, 2017.

[75] M. Liu, X. Li, Z. Ling, Y. Li, and H. Su, "Frame Mining: a free lunch for learning robotic manipulation from 3D point clouds," in *CoRL*, 2022.

[76] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *CVPR*, 2017.

[77] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," in *RSS*, 2023.

[78] Y. Ze, G. Zhang, K. Zhang, C. Hu, M. Wang, and H. Xu, "3D diffusion policy: Generalizable visuomotor policy learning via simple 3D representations," in *RSS*, 2024.

[79] F. Xiang, Y. Qin, K. Mo, Y. Xia, H. Zhu, F. Liu, M. Liu, H. Jiang, Y. Yuan, H. Wang, L. Yi, A. X. Chang, L. J. Guibas *et al.*, "SAPIEN: A simulated part-based interactive environment," in *CVPR*, 2020.

[80] J. Schulman *et al.*, "Trust region policy optimization," in *ICML*, 2015.

[81] I. Radosavovic, X. Wang, L. Pinto, and J. Malik, "State-only imitation learning for dexterous manipulation," in *IROS*, 2021.

[82] B. Kang, Z. Jie, and J. Feng, "Policy optimization with demonstrations," in *ICML*, 2018.