Hand-Object Interaction Pretraining from Videos

Himanshu Gaurav Singh* **Antonio Loquercio*** **Carmelo Sferrazza** **Jane Wu**

Haozhi Qi **Pieter Abbeel** **Jitendra Malik**

Abstract: We present an approach to learn *general robot manipulation priors* from 3D hand-object interaction trajectories. We build a framework to use in-the-wild videos to generate sensorimotor robot trajectories. We do so by lifting both the human hand and the manipulated object in a shared 3D space and retargeting human motions to robot actions. Generative modeling on this data gives us a task-agnostic base policy. This policy captures a general yet flexible manipulation prior. We empirically demonstrate that finetuning this policy, with both reinforcement learning (RL) and behavior cloning (BC), enables sample-efficient adaptation to downstream tasks and simultaneously improves robustness and generalizability compared to prior approaches. Qualitative experiments are available at: https://hgaurav2k.github.io/hop/.

Keywords: Learning from videos, dexterous manipulation.

1 Introduction

Reusable sensorimotor representations have the potential to give robots access to the versatility of their sensorimotor apparatus, thereby enabling them to achieve a wide variety of goals. Similar to advancements in other AI domains [1, 2], such representations are likely to be trained with unsupervised objectives on large datasets. In this work, we study the feasibility of training such representations using human videos in the context of dexterous manipulation.

Using videos as a data engine comes with several advantages: (1) they are abundant; (2) they cover a wide range of skills that we want robots to master; and (3) they reflect natural or socially acceptable behaviors that we want robots to emulate. However, training sensorimotor representations on videos is a challenging endeavor. First, videos only partially capture the nature of an agent's interaction with their surroundings. For instance, by looking at a person holding an object, it is almost impossible to estimate the force their fingers are exerting. In addition, the larger the embodiment gap between a human and a robot, the more their actions will differ to achieve the same objectives.

The difficulty of learning from videos led previous work to mostly focus on specific aspects of the problem. One line of research focused on training visual representations with off-the-shelf self-supervised vision algorithms on large vision datasets [3, 4, 5, 6, 7]. While simple and effective, such pretrained representations lack a motor component, making them less effective on downstream tasks [8]. Another line of work aims to extract both sensory and motor information from videos by estimating human motions in 3D [9, 10, 11, 12, 13]. However, these approaches require alignment between the training videos and the robot's downstream tasks, which compromises the generality of the learned representations. Finally, recent works aim to use egocentric videos of human activities to learn an explicit hand-object interaction prior in the form of a contact-pose prediction model [14, 15]. While a contact-pose prior is potentially task-agnostic, useful information in hand-object trajectories extends beyond contact-poses, including but not limited to pre/post-contact trajectories, intuitive physics of the interaction and human preferences.

*denotes equal contribution. All authors are affiliated to UC Berkeley.

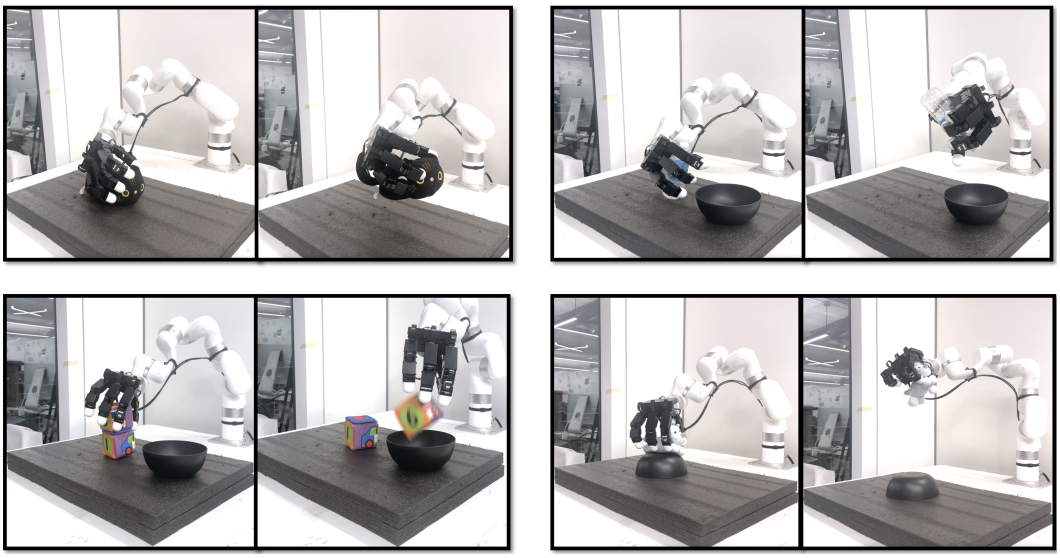

Figure 1: Real world rollouts of the policy finetuned from HOP using less than 50 demonstrations. HOP enables sample-efficient downstream adaptaion by learning a general manipulation prior from human videos.

In this paper, we present an approach to capture a general manipulation prior from in-the-wild videos. Such a prior is implicitly embedded in the weights of a causal transformer, pretrained with a conditional distribution matching objective on sensorimotor robot trajectories. These trajectories are generated by mapping 3D hand-object interactions to the robot's embodiment via a physically grounded simulator. The choice of an implicit prior, aligned with the current paradigm in vision and language research, has the potential advantage of becoming more and more expressive as the quality and diversity of the data increases. The resulting prior can be quickly adapted to any manipulation task either with reinforcement learning or behavioral cloning. After adaptation, the prior takes the form of an *end-to-end* policy mapping the robot's multi-modal sensory stream to low-level joint commands. This policy can be directly executed on a physical robot (see Fig. 1).

We empirically study the advantages brought forward by pretraining with hand-object interactions in both simulation and real-world experiments. The findings of this study indicate that our manipulation prior considerably speeds up skill acquisition compared to previous methods, even if such skills are not represented in the training videos. Additionally, it improves generalization and robustness to disturbances in the downstream policy.

2 Overview

The objective of **H**and-**O**bject interaction **P**retraing (HOP) is to capture general hand-object interaction priors from videos. In contrast to previous work, we do not assume a strict alignment of the human's intent in the video and the downstream robot tasks. Our key intuition is that the basic skills required for manipulation lie on a manifold whose axes are well covered by unstructured human-object interactions.

We extract sensorimotor information from videos by lifting the human hand and the manipulated object in a shared 3D space. We then bring such 3D representations to a physics simulator, where we map human motion to robot actions. There are several advantages to using a simulator as an intermediary between videos and robot sensorimotor trajectories: (i) we can add physics, inevitably lost in videos, back to the interactions; (ii) it enables the synthesis of large training datasets without putting the physical platform in danger; and (iii) we can add diversity to the data by randomizing the simulation environment, e.g., varying the friction between the robot's joints, the scene's layout, and the object's location relative to the robot.

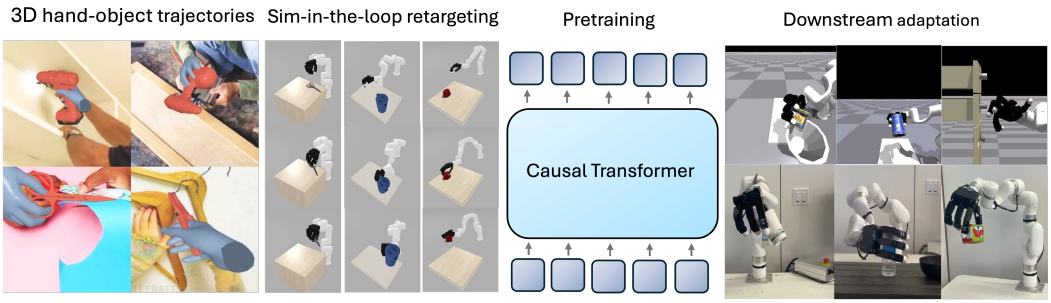

Figure 2: 3-D hand-object trajectories from in-the-wild human manipulation videos are re-targeted to a robot embodiment within a physics simulator, resulting in physically grounded robot data. General manipulation priors are learnt from this using generative modelling of trajectories. Such representation enables sample-efficient adaptation for new downstream tasks.

We generate a dataset of robot-object interactions $\mathcal{D} = \{\tau_1, \tau_2, \ldots, \tau_N\}$ where $\tau = \{(\boldsymbol{o}[0], \boldsymbol{a}[0]), (\boldsymbol{o}[1], \boldsymbol{a}[1]), \ldots, (\boldsymbol{o}[T], \boldsymbol{a}[T])\}$ are the observation-action pairs of a single sensorimotor trajectory. An observation $\boldsymbol{o}[k] \in \mathbb{O}$ at time $k \in [0, \ldots, T]$ consists of visual scene information (a depth image or a pointcloud) and robot's joint angles $\phi[k]$, *i.e.*, its proprioception. The action $\boldsymbol{a}[k] \in \mathbb{A}$ consists of continuous joint angles, which are converted to joint torques with a low-level PD controller. We use $\mathcal{D}$ to train a base policy $\boldsymbol{\pi}_b$ on the unsupervised objective of next-action prediction from a history of sensory observations, *i.e.*, $\hat{\boldsymbol{a}} = \boldsymbol{\pi}_b(\boldsymbol{o}[t : t - L])$, where L is a fixed context length. We finetune $\boldsymbol{\pi}_b$ to generate task-specific policies $\boldsymbol{\pi}_t$ either optimizing a reward with reinforcement learning or a behavioral cloning objective on few task-specific demonstrations. The next section presents each aspect of our method in detail.

3 Method

3.1 Lifting Hand-Object Interaction Videos to 3D

Recovering the underlying 3D structure of hand-object interactions from in-the-wild monocular videos is inherently ambiguous. To alleviate such ambiguity, previous work leveraged the insight that the human hand can be used as an anchor for the 3D location and scale of the manipulated object [16, 17, 18, 19, 20, 21]. Our setup to estimate hand-object interaction trajectories from videos builds upon recent advances in 3D vision. Our approach closely follows MCC-HO [16] with a few modifications to adapt it to our use case.

Given a single RGB image and an estimate of the 3D hand geometry from HaMeR [22], MCC-HO jointly infers hand-object geometry as point clouds. To fine-tune the quality of the prediction, MCC-HO finetunes the object's pose by fitting it to a CAD model. However, this finetuning assumes knowledge of the object the human is interacting with. To increase generality, we wave this assumption and skip the CAD-based post-processing. This simplification comes at the cost of reduced reconstruction quality and temporal smoothness. While we find the first problem not critical for pre-training, we increase temporal smoothness by anchoring object reconstructions to time-smoothed hand detections [22]. In addition, we make the simplifying assumption that the camera from which the video is collected is static. More details of our 3D estimation pipeline are provided in the appendix. The result of this pipeline is a sequence of 3D hand-object poses.

3.2 Mapping 3D Human-Object Interactions to Robot-Object Interactions

We formulate a non-linear optimization problem to generate a sensorimotor trajectory τ from a sequence of 3D hand-object poses. At each step k, we find the action $\boldsymbol{a}[k]$ by optimizing the following cost function:

$$\min_{\boldsymbol{a}[k]} \frac{1}{2}\|\boldsymbol{x}_h[k] - f(\boldsymbol{a}[k])\|^2 + \lambda\|\boldsymbol{a}[k] - \boldsymbol{\phi}[k-1]\|^2 \quad \text{s.t.} \quad \boldsymbol{a}[k] \in \mathbb{A}, \tag{1}$$

where f is the robot's forward kinematics, and $\boldsymbol{x}_h[k]$ are the 3D coordinates of a set of keypoints on the human hand. The first term of Eq. 1 represents the difference between the robot's and the human keypoints as a function of the robot's desired joints $\boldsymbol{a}[k]$. The second term is proportional to the energy required to execute the action $\boldsymbol{a}[k]$, which we minimize to favor smoothness.

While there are approaches that include the object dynamics in the optimization [23, 24, 25, 26], they are challenging to apply to in-the-wild videos due to noise in object pose estimates. In addition, in-the-wild videos do not have reliable information about objects' physical properties, *e.g.*, mass or friction. Therefore we disregard the dynamics of the manipulated object and place it on every step at the location observed in the video. While this can lead to physical implausibility in the object motion and possibly lack of force-closure grasps, the data quality does not deteriorate much in practice, as we can still learn useful behaviors. We empirically show that without object-trajectories (Section 5.3), the quality of the base policy decreases.

The optimization is performed independently on each timestep k, and the resulting actions $\boldsymbol{a}[k]$ are executed in a high-fidelity simulator to generate $\boldsymbol{o}[k]$. We refer to this method of mapping human motion to robot sensorimotor trajectories τ as *simulator-in-the-loop retargeting*. The primary advantage of this approach is the optimization in (1) can be conducted using a simplified forward kinematics f, reducing the computational burden. Despite this simplification, the actions are executed in a high-fidelity simulator, ensuring realistic behaviors and high-quality observations.

We randomize the simulated scene to increase data diversity. Specifically, we add obstacles like tables and walls to the scene and vary their positions relative to the robot. This allows us to add random constraints to this optimization problem, which increases the overall diversity in the extracted joint trajectories. This is particularly important for robots with kinematic redundancies, since they have multiple joint position trajectories for the same end-effector trajectory. Note that this approach to retargeting differs from previous methods, disregarding other objects in the scene and optimizing actions via physics-based constraints, *e.g.*, minimum jerk [12, 27, 9] or minimum velocity [13].

The quality of the resulting robot trajectories decreases as the difference between the environment where the video was collected, the simulated scene, and f grows. However, given the non-convex optimization landscape of (1), we can obtain good trajectories by running the optimization multiple times with various initial positions and scene layouts. High-quality data is then obtained by pruning the trajectories on metrics like collision with obstacles and the tracking error between the hand's and the robot's keypoints.

3.3 Robot Trajectory Pretraining

The resulting trajectory dataset $\mathcal{T}$ contains knowledge that could be valuable to any manipulation tasks. For instance, $\mathcal{T}$ has information about object affordance, *i.e.*, where and how to grasp; some intuitive (although rudimentary) physics, *e.g.*, an object should be reached upon before being lifted; or wrist-hand coordination, *i.e.*, the behavior of orienting and shaping the hand simultaneously while moving the wrist to maximize efficiency [28].

We aim to incorporate this knowledge as useful behavioral priors into a policy π_b that can be finetuned to downstream tasks. Similar to previous work in language [2], vision [29], and robotics [8, 30, 31], we instantiate π_b as a transformer [32] and train it on a generative modeling objective. Specifically, we train π_b to capture the conditional distribution $\Pi(\boldsymbol{a}[t-L:t]|\boldsymbol{o}[t-L:t])$ by optimizing the following loss:

$$\mathcal{L}(\tau;\theta) = \mathbb{E}_{t\sim[1\ldots T]}\left[\|\boldsymbol{a}[t-L:t] - \pi_b(\boldsymbol{o}[t-L:t])\|_1\right]. \tag{2}$$

However, unlike previous work, our pretraining dataset $\mathcal{T}$ contains neither real-world demonstrations nor complete task executions. This is because our data is generated from unstructured 3D hand-object interactions, and we disregard the dynamics of the manipulated object during retargeting (Sec. 3.2). Yet, we find that the pre-training paradigm in (2) leads to the emergence of useful representations in π_b.

Downstream Finetuning. The pretrained policy π_b exhibits primitive manipulation skills, *e.g.*, reaching an object with a reasonable grasp pose, while occasionally grasping successfully. We finetune these skills to a task by optimizing a reward with reinforcement learning or a behavior cloning loss on limited demonstrations. We finetune the whole model for the task. Empirically, we find that finetuned policies use the information in π_b to train faster, are more robust to disturbances, and generalize better than policies trained from scratch and a set of baselines. In addition, we find that the finetuning process re-utilizes the information in π_b even for tasks not explicitly represented in the training videos.

4 Experimental Setup

Robot. We use a low-cost 7-DoF xArm robot with a 16-DoF Allegro hand [33] vertically mounted at its end effector. The proprioception observation ϕ_k includes joint position from both robots. While we don't make any specific assumption about the robot embodiment, we use a multi-fingered hand instead of a parallel joint gripper since demonstration quality increases as the embodiment gap between the robot and the human decreases. We empirically found that since the robot base is fixed, a 7-DoF arm can track much better human motions than a 6-DoF arm, which often encounters singularities during such trajectories. Visual sensing comes from a single stereo camera (Zed-2) mounted on the robot's right side.

Simulation Setup. Our simulation environment is developed with the IsaacGym [34] simulator. The robot morphology and action space are identical to the real setup. However, since rendering depth images is prohibitively expensive, we give the agent access to the ground-truth object point-could instead of a depth image (see Section 2). Specific details about the task setup and reward design can be found in the appendix.

Video Datasets. Our pretraining dataset of 3D hand-object trajectories consists of sequences from two datasets: DexYCB [35] and 100 Days of Hands [36]. We use 250 videos from the DeXYCB dataset (right-hand only) annotated with ground truth hand-object trajectories as a source of high-quality data. We additionally use approximately 200 videos from the 100 Days of Hands dataset. Sixty percent of these videos were previously annotated with hand-object interaction trajectories [11], which we directly use. We annotate the remaining videos with our 3D estimation pipeline (Sec. 3.1). Overall, our combined dataset contains approximately 450 videos. We retarget these videos to obtain a pretraining dataset $\mathcal{T}$ of approximately $70,000$ trajectories.

Retargeting. We use low-storage BFGS [37] from the NLOpt library [38] for optimization. We perform simulation-in-the-loop retargeting in a simple simulated scene with a ground floor on which the robot and a static table are placed 65cm apart. Objects start their trajectories above the table with a random pose. We run the optimization 700 times for each video, randomizing the table location and the robot's initial joint state. We add a trajectory to $\mathcal{T}$ only if, at any time, their retargeting error (See Eq. (1)) is below 3cm and the arm does not collide with the table or the floor. Our code is built upon the implementation of Qin et al. [39].

Transformer. Similar to previous work [40], we represent the policy π_b with a GPT-2-style causal transformer. The policy takes proprioception and observation input from the past 16 timesteps and predicts the next action. Details about the architecture can be found in the appendix.

Pretraining. We train the transformer with the objective in Eq. (2) on $\mathcal{T}$. While we could make the prediction autoregressive and add decoding heads and proxy losses for future proprioception and images (as in [13]), we empirically found these changes to be not very helpful in practice to our tasks. Therefore, we predict only future actions for simplicity. We use as optimizer AdamW [41] with initial learning rate of 10^{-4} and weight decay of 10^{-2}. We trained two distinct base policies—one with depth observations and the other with point cloud observations. The former is used for real-world, and the latter for simulation experiments. However, it's important to note that both policies were trained on exactly the same trajectories; only the associated sensor observations differed.

Finetuning. *In simulation*, we finetune the transformer with PPO [42] using the default hyperparameters from [43]. However, we add a few modifications inspired by [44] for effective fine-tuning: (1) we use a small initial exploration noise of 0.1; (2) the value and policy networks share the observation tokenizer, but the tokenizer's weights are not updated by the value function's gradients; (3) we warm up the value function's parameters for the first 200 gradient steps, keeping the actor parameters fixed. Since reinforcement learning requires up to 1 billion steps to convergence, and our simulator does not offer fast multi-gpu rendering, we use pointclouds as visual information, as they can be efficiently simulated. *In the real world*, we finetune the entire π_b on limited demonstrations with the same objective and hyperparameters used for pretraining. In these experiments, we use depth images as input to our policy since pointcloud estimation in the real world generally requires multiple cameras, while our real-world setup has a single camera.

Inference. At test time, the model operates in *closed-loop*: it receives the past and current observations as input and predicts the next action to execute. The prediction loop runs at 20Hz. The predicted action is sent to the xArm and Allegro low-level controllers, which operate at 120Hz and 300Hz, respectively.

5 Experimental Results

We design an experimental procedure to analyze the advantages brought forward by HOP in terms of finetuning efficiency, generalization, and robustness to perturbations. Specifically, we ask the following questions: (i) *How does HOP compare to vision-only pre-training approaches for robot learning?* (ii) *How does HOP compare to existing demonstration-guided reinforcement learning algorithms [45, 46]?* (iii) *How does learning from hand-object interaction trajectories compare to learning hand-pose priors only* [47, 48, 9]?. We answer these questions via controlled experiments in simulation and the physical world.

5.1 Comparison to visual pre-training baselines (real-world).

Baselines. We compare our approach to visual pre-training systems. Such systems are trained on large image or video datasets but lack a motor component. Specifically, we compare to methods using the following pre-training data:

- **ImageNet** We encode the depth image with a VIT-B network [49] pre-trained on ImageNet and pass the resulting CLS token embeddings to our transformer. The latter is then trained with real-world data. We consider two variants: using the VIT features zero-shot (*Imagenet ZS*) and finetuning them on the downstream dataset (*ImageNet F*).
- **Internet Videos** We use off-the-shelf visual features from R3M [7], VIP [3], and MVP [8]. These features were obtained with unsupervised contrastive learning objectives on large video datasets, e.g. Ego4D [50]. Conversely to ours, these baselines don't use depth but RGB images as input.

We additionally compare to Diffusion Policies [23] using a UNet backbone since our tasks exhibit temporally smooth desired action sequences. Similarly to ours, this baseline uses depth as input. With the above baselines, we want to understand how classic methods for behavior cloning work in our setting, where a single camera and a limited number of demonstrations are available. Indeed, the previous approaches are generally applied with a large number of demonstrations and multiple RGB cameras.

Tasks We evaluate our approach in the real world on three tasks of increasing complexity. In the first task, *Grasp and Drop*, the robot needs to unstack a cube and put it in a bowl. The second is the *Grasp and Pour* task, where the robot needs to pick a bottle and point it towards a bowl. In the third task, *Grasp and Lift*, the robot must pick up one of 4 different-looking objects, all requiring different spatial affordances. One single model is trained to pick up all objects. In this task, we evaluate the ability of the approach to adapt with a few demonstrations on very different object

shapes. We collect 15 demonstrations for the first two tasks and 50 demonstrations for the third task. We encourage the reader to check our project page for visualizing the tasks. More details about the scene setup and evaluation criteria can be found in the appendix.

Results Table 1 summarizes the result of our study. The findings indicate that for tasks with a single object (Grasp & Drop, Grasp & Pour), all methods perform comparably and achieve, with some sporadic exceptions, a close-to-perfect success rate. However, in the hardest task (Grasp and Lift), where a single policy needs to pick four objects with different affordances, our approach has a margin of 30 percentage points to ImageNet Finetuned, the best-performing baseline. We additionally find that the baselines using RGB data are much more successful with a single object than with multiple ones. This is likely because there are not enough demonstrations to learn object-specific affordances. Overall, these results empirically validate the value of our sensorimotor pretraining strategy.

	Depth-input				**RGB-input**		
Task	Ours	Diffusion-Policy	Imagenet-ZS	Imagenet-F	R3M	VIP	MVP
Grasp & Drop	0.80	**0.90**	0.90	0.80	0.0	0.40	0.20
Grasp & Pour	**1.0**	0.20	0.80	0.70	1.0	1.0	1.0
Grasp & Lift	**0.65**	0.30	0.30	0.35	0.0	0.0	0.0

Table 1: Real-robot results (success rate % averaged over 20 rollouts). Note that RGB baselines are not directly comparable to our approach since our policy takes depth images as input.

5.2 Comparison to demonstration-guided reinforcement learning strategies (simulation)

Our simulation experiments investigate the effectiveness of HOP as a base model for adaptation to downstream tasks using RL. The simulation agent is identical in morphology to the real robot.

Baselines. We compare our approach to three baselines: (1) training from scratch (*PPO*); (2) demonstration-guided reinforcement learning with a proxy imitation objective [45] (*DAPG*); and (3) using adversarial objectives to keep the policy close to the demonstrations [46] (*AMP*). DAPG is the closest to our work, as it trains on a weighted sum of behavioral cloning and reinforcement learning losses. However, it assumes access to expert demonstrations in the downstream task. Our pre-training dataset does not fulfill this assumption. Indeed, humans might not behave optimally according to the reward, or the task might not be well represented in the pre-training dataset. Similarly to previous work [51], we found that training from scratch is unsuccessful using joint-position control as action space, consistently leading the PPO baseline to fail. Therefore, we use the moving-average action space proposed by Petrenko et al. [43] to improve its performance. All baselines use the same environment settings and training strategy, *e.g.*, domain randomization parameters, as our approach.

Tasks and Metrics. We evaluate approaches on three tasks. The first requires picking objects and placing them at a specific location (*Grasp and Lift*). The second is to grasp objects and throw them in a basket (*Grasp and Throw*). In the final task, the robot is required to open a cabinet. We evaluate performance using success over 256 environments with different objects and report the mean and standard deviation over three seeds per approach. More details can be found in the appendix.

HOP enables sample-efficient RL and effective exploration In Figure 3, it is demonstrated that our approach outperforms all baselines by a large margin, especially when the pretraining corpus is not closely related to the task. This is expected because DAPG strongly biases exploration in the neighborhood of the pre-training trajectories, which may potentially be misaligned with the downstream task. Furthermore, we observed that the adversarial training scheme of AMP is unstable and does not scale well with the amount of data. Finally, we find that using HOP leads to a 2-5X improvement in sample efficiency compared to training from scratch. Overall, these experiments

Figure 3: **Comparison of HOP-initialized actor with baselines.** HOP improves sample-efficiency of online RL across multiple tasks, particularly when the downstream task and the behaviors in the data are less aligned, as in *Lift & Throw*. Runs are averaged across three randomly chosen seeds.

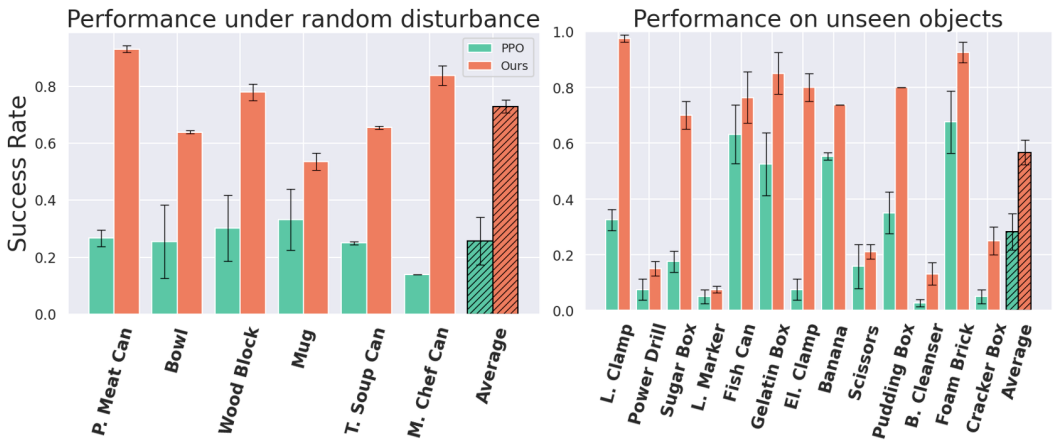

Figure 4: **Evaluating RL finetuning under out-of-distribution scenarios** (Left) To test grasp robustness in the task *Grasp & Lift*, we apply to the grasped objects, forces in random direction equal to their weights. When initialized with HOP, the resulting policy is more than $3\times$ more robust compared to training PPO from scratch. (Right) We evaluate grasp success on multiple objects from the YCB dataset that were not part of the training set. When initialized with HOP, the resulting policy is more than $2\times$ more robust compared to training PPO from scratch.

show that initializing with HOP enables more informed exploration than the baselines and reduces variance in policy gradients.

HOP learns robust and general behaviors Policies fine-tuned from HOP can potentially bias exploration toward human-like behavior, leading to more robustness against forces. This is shown in Fig. 4. Agents trained with our approach perform better when subject to forces than the ones trained from scratch. In addition, we show in Fig. 4 that our approach generalizes 3x better than the policy trained from scratch. The training objects are different from the testing ones in their mass, aspect ratio, and relative size with respect to the hand. The performance generally drops whenever the test object is heavy (power drill), too large (cracker box), or too small (marker and scissors) for the allegro hand, which is approximately 1.5X larger than a human hand. Note that in these experiments we train the scratch policy with two billion samples.

Affordances Training an RL policy from scratch for a dexterous hand often leads to grasping poses that are unlike general human affordances. Online RL exploration near a learned human-object interaction prior biases the optimization landscape to favor human-like affordances. As shown in Figure 5, we find that for the *Grasp and Lift* task, our policy grasps objects with more stable and human-like affordances than PPO training from scratch.

Figure 5: Online exploration around the learnt prior from humans leads to grasps with more human-like and stable affordances compared to training PPO from scratch.

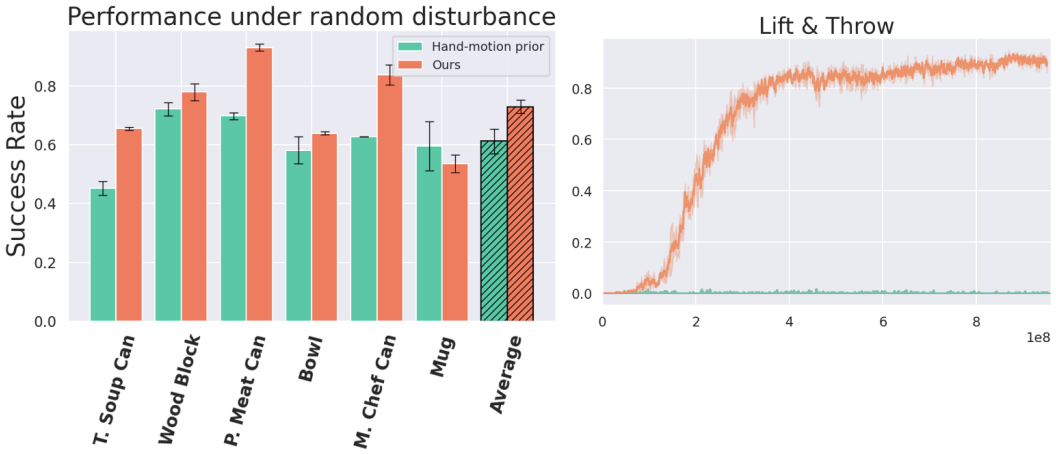

Figure 6: Pre-training only a hand-motion prior leads to decrease in robustness of grasps to force disturbances (left). With our approach, the pre-trained policy learns a prior on object affordances which leads to more robust grasps. In addition, pretraining with object poses leads to a more flexible prior and better finetuning to tasks less aligned with the pre-training data (right).

5.3 Comparison to learning a hand-only motion prior (simulation)

Prior work has shown the benefits of learning a prior on hand motions from videos of human activities [47, 48, 9]. This section aims to understand the benefits of learning a prior on the object and the hand *jointly*. We hypothesize that learning from hand-object interactions gives the base model useful information beyond eigen-grasps (which are captured by a hand-only motion prior), like, for instance, pre- and post-contact trajectories, intuitive physics of the interaction, and human preferences.

We evaluate this hypothesis by training a base policy on our pre-training corpus using masked object observations. This encourages the base policy to primarily learn a *hand motion* prior. As illustrated in Figure 6 (left), we observe that such a pre-trained policy exhibits reduced robustness to grasp disturbances. Furthermore, we find that the hand-only prior is insufficient for learning an effective policy in the Grasp and Throw task (Fig. 6, right). Since this task is underrepresented in the pre-training corpus, the hand motions required are unlikely to be adequately captured by a hand-only prior. In contrast, learning a joint hand-object prior provides the model with a more comprehensive understanding of manipulation, enabling quicker adaptation to this downstream task.

6 Related Work

Learning Policies from Human Videos. In-the-wild videos hold the promise of solving the data problem in robotics. One of the pioneering efforts in this direction is by Yang et al. [52], where video data was used to generate action plans. Several works followed up on this idea, relying on pre-defined action primitives [53, 54, 55, 56, 57, 58]. However, waving the requirement for pre-defined

primitives is challenging since in-the-wild videos lack motor information. One way to recover motor information from videos is training with a trajectory-matching objective [59, 60, 61, 62], possibly using intermediate representations like object segmentation or optical flow [63, 64, 65]. However, this approach requires collecting task- and environment-specific videos where humans and robots operate in the same workspace. Therefore, the trajectory-matching formulation largely constrains the number of videos that can be used for training. To overcome these constraints, researchers have focused on either learning exclusively visual representations from videos or extracting 3D human poses and mapping them to robot actions. In the following, we cover these works in detail.

Visual Representation Learning for Robotics. Inspired by successes in computer vision [1] and natural language processing [66], the robot learning community has recently focused on pretraining representations on large video datasets like Ego4D [50] and fine-tuning these representations on downstream tasks [4, 7, 3, 5]. However, being the training objective based exclusively on image reconstruction, the representations focus primarily on low-level vision features, *e.g.*, shapes or edges. This gives them limited benefits compared to representations trained on standard vision datasets [6]. Overall, these works focus on visual generalization, *e.g.*, picking up two objects with the same shape but different colors. However, they have not yet demonstrated action generalization, where motor skills are adapted to accomplish novel objectives.

Actions from Videos via 3D. One common approach to extracting action information from videos is using 3D as an intermediate representation. If the embodiment gap is small, human motions can be mapped to robot actions via inverse kinematics. This is particularly effective when the videos are task-specific, *i.e.*, when the robot aims to mimic the human motion [10, 12, 67, 68, 69, 70, 11]. Instead of learning specific skills, other works focus on learning a re-usable sensorimotor prior from videos. However, this prior only captures human actions [47, 48, 9], disregarding the trajectory of the manipulated object. Conversely, our work aims to use 3D hand-object interactions from in-the-wild videos to learn a re-usable prior for object manipulation.

Dexterous Manipulation. Dexterous manipulation has been studied for decades [71, 72, 73, 74, 75]. In recently years, learning-based approaches make significant progress [76, 77]. They can be generally categorized to learning in simulation and then transferring to the real world (Sim-to-Real) [78, 79, 80, 81, 82, 83], and learning in the real-world [39, 84, 85, 86, 87]. Qin et al. [12] uses hand-object trajectories but only use it to collect demonstrations. Xu et al. [88] also does functional grasp generation, but the results are limited in simulation. Most of the aforementioned work learn policies from scratch and does not use any internet data as prior. Our approach studies learning hand-object interaction prior from human videos and is effective both in simulation and real-world.

7 Conclusion and Limitations

This work presents an approach to learning general yet flexible manipulation priors for robot policies from human videos. While our approach demonstrates a way to pre-train on a single object interaction, this can, in practice, be limiting. Indeed, human behavior in a video can potentially be conditioned on information encompassing multiple objects in the current and previous scenes. This leads to a loss of signal that could be extracted from the raw video. We predict that advances in 3-D reconstruction will enable us to use a more complex scene reconstruction and pretraining.

Acknowledgments

This work was supported by the DARPA Machine Common Sense program, the DARPA Transfer from Imprecise and Abstract Models to Autonomous Technologies (TIAMAT) program, and by the ONR MURI award N00014-21-1-2801. This work was also funded by ONR MURI N00014-22-1-2773. We thank Adhithya Iyer for assistance with teleoperation systems, Phillip Wu for setting-up the real robot, and Raven Huang, Jathushan Rajasegaran and Yutong Bai for helpful discussions.

References

[1] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick. Masked autoencoders are scalable vision learners. In *CVPR*, 2022.

[2] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 2019.

[3] Y. J. Ma, S. Sodhani, D. Jayaraman, O. Bastani, V. Kumar, and A. Zhang. Vip: Towards universal visual reward and representation via value-implicit pre-training. In *ICLR*, 2023.

[4] I. Radosavovic, T. Xiao, S. James, P. Abbeel, J. Malik, and T. Darrell. Real-world robot learning with masked visual pre-training. *CoRL*, 2022.

[5] A. Majumdar, K. Yadav, S. Arnaud, J. Ma, C. Chen, S. Silwal, A. Jain, V.-P. Berges, T. Wu, J. Vakil, P. Abbeel, J. Malik, D. Batra, Y. Lin, O. Maksymets, A. Rajeswaran, and F. Meier. Where are we in the search for an artificial visual cortex for embodied intelligence? *NeurIPS*, 2023.

[6] S. Dasari, M. K. Srirama, U. Jain, and A. Gupta. An unbiased look at datasets for visuo-motor pre-training. In *CoRL*, 2023.

[7] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta. R3m: A universal visual representation for robot manipulation. In *CoRL*, 2022.

[8] I. Radosavovic, B. Shi, L. Fu, K. Goldberg, T. Darrell, and J. Malik. Robot learning with sensorimotor pre-training. In *CoRL*, 2023.

[9] K. Shaw, S. Bahl, and D. Pathak. Videodex: Learning dexterity from internet videos. In *CoRL*, 2022.

[10] X. B. Peng, A. Kanazawa, J. Malik, P. Abbeel, and S. Levine. Sfv: Reinforcement learning of physical skills from videos. *Transactions On Graphics*, 2018.

[11] A. Patel, A. Wang, I. Radosavovic, and J. Malik. Learning to imitate object interactions from internet videos. *arXiv:2211.13225*, 2022.

[12] 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.

[13] I. Radosavovic, B. Zhang, B. Shi, J. Rajasegaran, S. Kamat, T. Darrell, K. Sreenath, and J. Malik. Humanoid locomotion as next token prediction. *arXiv:2402.19469*, 2024.

[14] A. Kannan, K. Shaw, S. Bahl, P. Mannam, and D. Pathak. Deft: Dexterous fine-tuning for real-world hand policies. *CoRL*, 2023.

[15] P. Mandikal and K. Grauman. Dexvip: Learning dexterous grasping with human hand pose priors from video. In *Conference on Robot Learning*, pages 651–661, 2022.

[16] J. Wu, G. Pavlakos, G. Gkioxari, and J. Malik. Reconstructing hand-held objects in 3d. *arXiv:2404.06507*, 2024.

[17] H. Choi, N. Chavan-Dafle, J. Yuan, V. Isler, and H. Park. Handnerf: Learning to reconstruct hand-object interaction scene from a single rgb image. *arXiv:2309.07891*, 2023.

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

[19] B. Tekin, F. Bogo, and M. Pollefeys. H+O: Unified egocentric recognition of 3d hand-object poses and interactions. In *CVPR*, 2019.

[20] 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.

[21] Z. Cao, I. Radosavovic, A. Kanazawa, and J. Malik. Reconstructing hand-object interactions in the wild. In *ICCV*, 2021.

[22] G. Pavlakos, D. Shan, I. Radosavovic, A. Kanazawa, D. Fouhey, and J. Malik. Reconstructing hands in 3D with transformers. In *CVPR*, 2024.

[23] X. Zhu, J. Ke, Z. Xu, Z. Sun, B. Bai, J. Lv, Q. Liu, Y. Zeng, Q. Ye, C. Lu, M. Tomizuka, and L. Shao. Diff-lfd: Contact-aware model-based learning from visual demonstration for robotic manipulation via differentiable physics-based simulation and rendering. In *CoRL*, 2023.

[24] A. S. Lakshmipathy, N. Feng, Y. X. Lee, M. Mahler, and N. Pollard. Contact edit: Artist tools for intuitive modeling of hand-object interactions. *Transactions on Graphics*, 2023.

[25] A. S. Lakshmipathy, J. K. Hodgins, and N. S. Pollard. Kinematic motion retargeting for contact-rich anthropomorphic manipulations. *arXiv:2402.04820*, 2024.

[26] Y. Kim, H. Park, S. Bang, and S.-H. Lee. Retargeting human-object interaction to virtual avatars. *Transactions on Visualization and Computer Graphics*, 2016.

[27] A. Sivakumar, K. Shaw, and D. Pathak. Robotic telekinesis: Learning a robotic hand imitator by watching humans on youtube. In *RSS*, 2022.

[28] L. A. Jones and S. J. Lederman. *Human hand function*. Oxford university press, 2006.

[29] M. Chen, A. Radford, R. Child, J. Wu, H. Jun, D. Luan, and I. Sutskever. Generative pretraining from pixels. In *ICML*, 2020.

[30] A. Kumar, A. Singh, F. Ebert, M. Nakamoto, Y. Yang, C. Finn, and S. Levine. Pre-training for robots: Offline rl enables learning new tasks from a handful of trials. In *RSS*, 2023.

[31] O. M. Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, T. Kreiman, C. Xu, J. Luo, Y. L. Tan, L. Y. Chen, P. Sanketi, Q. Vuong, T. Xiao, D. Sadigh, C. Finn, and S. Levine. Octo: An open-source generalist robot policy. In *RSS*, 2024.

[32] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *NeurIPS*, 2017.

[33] WonikRobotics. Allegrohand. https://www.wonikrobotics.com/, 2013.

[34] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State. Isaac gym: High performance gpu-based physics simulation for robot learning. In *NeurIPS Datasets and Benchmarks*, 2021.

[35] Y.-W. Chao, W. Yang, Y. Xiang, P. Molchanov, A. Handa, J. Tremblay, Y. S. Narang, K. V. Wyk, U. Iqbal, S. Birchfield, J. Kautz, and D. Fox. Dexycb: A benchmark for capturing hand grasping of objects. *CVPR*, 2021.

[36] D. Shan, J. Geng, M. Shu, and D. F. Fouhey. Understanding human hands in contact at internet scale. In *CVPR*, 2020.

[37] D. C. Liu and J. Nocedal. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 1989.

[38] S. G. Johnson. The NLopt nonlinear-optimization package. https://github.com/stevengj/nlopt, 2007.

[39] Y. Qin, H. Su, and X. Wang. From one hand to multiple hands: Imitation learning for dexterous manipulation from single-camera teleoperation. *RA-L*, 2022.

[40] L. Chen, K. Lu, A. Rajeswaran, K. Lee, A. Grover, M. Laskin, P. Abbeel, A. Srinivas, and I. Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *NeurIPS*, 2021.

[41] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. In *ICLR*, 2019.

[42] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv:1707.06347*, 2017.

[43] A. Petrenko, A. Allshire, G. State, A. Handa, and V. Makoviychuk. Dexpbt: Scaling up dexterous manipulation for hand-arm systems with population based training. In *RSS*, 2023.

[44] R. Ramrakhya, D. Batra, E. Wijmans, and A. Das. Pirlnav: Pretraining with imitation and rl finetuning for objectnav. In *CVPR*, 2023.

[45] 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. *arXiv:1709.10087*, 2017.

[46] X. B. Peng, Z. Ma, P. Abbeel, S. Levine, and A. Kanazawa. Amp: Adversarial motion priors for stylized physics-based character control. *Transactions on Graphics*, 2021.

[47] H. Bharadhwaj, A. Gupta, S. Tulsiani, and V. Kumar. Zero-shot robot manipulation from passive human videos. *arXiv:2302.02011*, 2023.

[48] 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. *NeurIPS*, 2024.

[49] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[50] K. Grauman, A. Westbury, E. Byrne, Z. Chavis, A. Furnari, R. Girdhar, J. Hamburger, H. Jiang, M. Liu, X. Liu, M. Martin, T. Nagarajan, I. Radosavovic, S. K. Ramakrishnan, F. Ryan, J. Sharma, M. Wray, M. Xu, E. Z. Xu, C. Zhao, S. Bansal, D. Batra, V. Cartillier, S. Crane, T. Do, M. Doulaty, A. Erapalli, C. Feichtenhofer, A. Fragomeni, Q. Fu, A. Gebreselasie, C. Gonzalez, J. Hillis, X. Huang, Y. Huang, W. Jia, W. Khoo, J. Kolar, S. Kottur, A. Kumar, F. Landini, C. Li, Y. Li, Z. Li, K. Mangalam, R. Modhugu, J. Munro, T. Murrell, T. Nishiyasu, W. Price, P. R. Puentes, M. Ramazanova, L. Sari, K. Somasundaram, A. Southerland, Y. Sugano, R. Tao, M. Vo, Y. Wang, X. Wu, T. Yagi, Z. Zhao, Y. Zhu, P. Arbelaez, D. Crandall, D. Damen, G. M. Farinella, C. Fuegen, B. Ghanem, V. K. Ithapu, C. V. Jawahar, H. Joo, K. Kitani, H. Li, R. Newcombe, A. Oliva, H. S. Park, J. M. Rehg, Y. Sato, J. Shi, M. Z. Shou, A. Torralba, L. Torresani, M. Yan, and J. Malik. Ego4d: Around the world in 3,000 hours of egocentric video. In *CVPR*, pages 18995–19012, 2022.

[51] Y. Jiang, C. Wang, R. Zhang, J. Wu, and L. Fei-Fei. Transic: Sim-to-real policy transfer by learning from online correction. *arXiv:2405.10315*, 2024.

[52] Y. Yang, Y. Li, C. Fermuller, and Y. Aloimonos. Robot learning manipulation action plans by" watching" unconstrained videos from the world wide web. In *AAAI*, 2015.

[53] A. Nguyen, D. Kanoulas, L. Muratore, D. G. Caldwell, and N. G. Tsagarakis. Translating videos to commands for robotic manipulation with deep recurrent neural networks. In *ICRA*, 2018.

[54] J. Lee and M. S. Ryoo. Learning robot activities from first-person human videos using convolutional future regression. In *CVPR*, 2017.

[55] V. Arapi, C. Della Santina, D. Bacciu, M. Bianchi, and A. Bicchi. Deepdynamichand: a deep neural architecture for labeling hand manipulation strategies in video sources exploiting temporal information. *Frontiers in neurorobotics*, 2018.

[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] S. Bahl, R. Mendonca, L. Chen, U. Jain, and D. Pathak. Affordances from human videos as a versatile representation for robotics. In *CVPR*, 2023.

[58] H. Bharadhwaj, R. Mottaghi, A. Gupta, and S. Tulsiani. Track2act: Predicting point tracks from internet videos enables diverse zero-shot robot manipulation. *arXiv:2405.01527*, 2024.

[59] S. Bahl, A. Gupta, and D. Pathak. Human-to-robot imitation in the wild. In *RSS*, 2022.

[60] C. Finn, I. Goodfellow, and S. Levine. Unsupervised learning for physical interaction through video prediction. *NeurIPS*, 2016.

[61] T. Yu, C. Finn, A. Xie, S. Dasari, T. Zhang, P. Abbeel, and S. Levine. One-shot imitation from observing humans via domain-adaptive meta-learning. In *RSS*, 2018.

[62] J. Jin, L. Petrich, M. Dehghan, Z. Zhang, and M. Jagersand. Robot eye-hand coordination learning by watching human demonstrations: a task function approximation approach. In *ICRA*, 2019.

[63] H. Bharadhwaj, A. Gupta, V. Kumar, and S. Tulsiani. Towards generalizable zero-shot manipulation via translating human interaction plans. *arXiv:2312.00775*, 2023.

[64] C. Wen, X. Lin, J. So, K. Chen, Q. Dou, Y. Gao, and P. Abbeel. Any-point trajectory modeling for policy learning. In *RSS*, 2023.

[65] S. Kumar, J. Zamora, N. Hansen, R. Jangir, and X. Wang. Graph inverse reinforcement learning from diverse videos. In *CoRL*, 2022.

[66] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *ACL*, 2019.

[67] J. Ye, J. Wang, B. Huang, Y. Qin, and X. Wang. Learning continuous grasping function with a dexterous hand from human demonstrations. *RA-L*, 2023.

[68] 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.

[69] Z. Q. Chen, K. Van Wyk, Y.-W. Chao, W. Yang, A. Mousavian, A. Gupta, and D. Fox. Learning robust real-world dexterous grasping policies via implicit shape augmentation. In *CoRL*, 2022.

[70] Z. Q. Chen, K. Van Wyk, Y.-W. Chao, W. Yang, A. Mousavian, A. Gupta, and D. Fox. Dextransfer: Real world multi-fingered dexterous grasping with minimal human demonstrations. *arXiv:2209.14284*, 2022.

[71] R. Fearing. Implementing a force strategy for object re-orientation. In *ICRA*, 1986.

[72] L. Han and J. C. Trinkle. Dextrous manipulation by rolling and finger gaiting. In *ICRA*, 1998.

[73] A. M. Okamura, N. Smaby, and M. R. Cutkosky. An overview of dexterous manipulation. In *ICRA*, 2000.

[74] M. T. Ciocarlie and P. K. Allen. Hand posture subspaces for dexterous robotic grasping. *IJRR*, 2009.

[75] A. S. Morgan, K. Hang, B. Wen, K. Bekris, and A. M. Dollar. Complex in-hand manipulation via compliance-enabled finger gaiting and multi-modal planning. *RA-L*, 2022.

[76] OpenAI, M. Andrychowicz, B. Baker, M. Chociej, R. Józefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, J. Schneider, S. Sidor, J. Tobin, P. Welinder, L. Weng, and W. Zaremba. Learning dexterous in-hand manipulation. *IJRR*, 2019.

[77] C. Yu and P. Wang. Dexterous manipulation for multi-fingered robotic hands with reinforcement learning: a review. *Frontiers in Neurorobotics*, 2022.

[78] OpenAI, I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, J. Schneider, N. Tezak, J. Tworek, P. Welinder, L. Weng, Q. Yuan, W. Zaremba, and L. Zhang. Solving rubik's cube with a robot hand. *arXiv:1910.07113*, 2019.

[79] A. Handa, A. Allshire, V. Makoviychuk, A. Petrenko, R. Singh, J. Liu, D. Makoviichuk, K. Van Wyk, A. Zhurkevich, B. Sundaralingam, Y. Narang, J.-F. Lafleche, D. Fox, and G. State. Dextreme: Transfer of agile in-hand manipulation from simulation to reality. In *ICRA*, 2023.

[80] 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.

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

[82] 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.

[83] A. Agarwal, S. Uppal, K. Shaw, and D. Pathak. Dexterous functional grasping. In *CoRL*, 2023.

[84] 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.

[85] I. Guzey, B. Evans, S. Chintala, and L. Pinto. Dexterity from touch: Self-supervised pre-training of tactile representations with robotic play. In *CoRL*, 2023.

[86] I. Guzey, Y. Dai, B. Evans, S. Chintala, and L. Pinto. See to touch: Learning tactile dexterity through visual incentives. In *ICRA*, 2024.

[87] 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.

[88] 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.

[89] T. Xiao, I. Radosavovic, T. Darrell, and J. Malik. Masked visual pre-training for motor control. *arXiv:2203.06173*, 2022.

[90] B. Calli, A. Singh, J. Bruce, A. Walsman, K. Konolige, S. Srinivasa, P. Abbeel, and A. M. Dollar. Yale-cmu-berkeley dataset for robotic manipulation research. *IJRR*, 2017.

[91] P. Wu, Y. Shentu, Z. Yi, X. Lin, and P. Abbeel. Gello: A general, low-cost, and intuitive teleoperation framework for robot manipulators. *arXiv:2309.13037*, 2023.

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

[93] 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.

[94] C. Chi, Z. Xu, C. Pan, E. Cousineau, B. Burchfiel, S. Feng, R. Tedrake, and S. Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. In *Proceedings of Robotics: Science and Systems (RSS)*, 2024.

8 Supplementary Material

8.1 3D Hand-Object Interaction from Videos

Our setup closely follows the pre-trained MCC-HO model [16] to lift RGB videos to 3D. However, this approach was designed to work on a single frame, while we are interested in extracting trajectories from videos. Specifically, MCC-HO input is the image patch containing the hand and manipulated object. Acquiring these patches requires identifying which hand and object is part of the manipulation sequence from in-the-wild videos. To make the problem tractable, we use the fact that the 100 Days of Hands dataset (100DOH) [36] includes 100K labeled frames (*e.g.*, hand and object bounding boxes) randomly sampled from Internet videos.

We use such sparse labels as follows: Given a video with a labeled frame at timestamp t_0, we download a 10-second video clip centered at t_0 using the original video frame rate. Then, we propagate the combined hand-object bounding box from t_0 to $t < t_0$ and $t > t_0$ iteratively until the interacting hand is no longer detected by HaMeR [22]. At each frame t, the labeled hand-object bounding box is translated so the hand bounding-box center is aligned with the HaMeR hand bounding-box center. Subsequently, the images are cropped/resized using these hand-object bounding boxes and passed to MCC-HO for network inference. This gives a sequence of 3D hand-object poses. We do not temporally smooth the sequences further (*e.g.*, via high-pass filtering) since this happens (to a certain extent) as a by-product of the robot's inertia during the simulator-in-the-loop motion re-targeting.

8.2 Simulation Experiments Setup

We reuse environment definitions from previous works ([34, 89]) with minimal changes. We do not make any changes to the structure of the reward function. Below, we provide a brief description for each task:

1. *Grasp and Lift* We adapt this environment definition from IsaacGymEnvs[34] for this task. We changed the robot to an Xarm7 with an Allegro Hand as the end-effector attached vertically. For increasing realism, we enable gravity in the environment and increase the number of convex decompositions in VHACD. For the objects, we use 6 canonical objects from the YCB[90] benchmark. Specifically, we use the *Tomato Soup Can*, *Wood Block*, *Potted Meat Can*, *Bowl*, *Bowl*, *Master Chef Can* and *Mug*. The choice of the objects is done keeping in mind the limitations of the robot embodiment such as the large size of the palm and thick fingers.

2. *Lift and Throw* We adapt this environment definition for *Lift and Throw*. We change the robot, the objects and the physical parameters same as in *Grasp and Lift*.

3. *Open Cabinet* We adapt this environment definition from PixMC[89] for *Open Cabinet*. We change the robot as in above tasks and keep other parameters the same. The input to the policy in this task is the point cloud for the handle of the cabinet.

8.3 Model Architecture

Our policy is a GPT-2 style transformer with causal attention. The transformer has 4 heads, 4 layers, and a hidden dimension of 192. For real-world experiments, proprioception and the depth images are embedded to the hidden size using a linear projection and a 4-layer CNN, respectively. In simulation, point clouds are embedded to the hidden size using a pointnet with 2 hidden layers of size 64. The input to the transformer are proprioception and depth/pointcloud embeddings for previous 16 timesteps. Additive learnable positional embeddings are used for both the proprioception and depth embeddings.

8.4 Real-World Experiments Setup

Data Collection We build a custom teleoperation setup for data collection by combining two existing systems. We control the xArm with a Gello [91] and the hand motion with VR using OpenTeach [92]. We experimented with controlling the whole system via VR but found obtaining precise and smooth motion challenging. While our solution could achieve such motions, it comes with the disadvantage of requiring two people to collect demonstrations. We randomize the initial pose of the robot between demonstrations by adding a uniform noise of magnitude 0.3 rad to all joints from a fixed starting location.

Inference While sensors operate at different frequencies, we get the latest available measurements from each sensor at constant intervals to achieve a whole inference loop of 20Hz. The predicted action, *i.e.*absolute joint positions, are given to a low-level P controller that operates at 120Hz. Such controller directly sends commands to the xArm API to convert joint position into joint torques.

Tasks We study three tasks of increasing complexity. We recommend the reader to checkout videos on our supplementary website to visualize the task setup. For all tasks, we report a success if the policy can complete the task in less than 30 seconds. In the following, an explanation of each task:

- *Grasp and Drop* The robot needs to pick up a soft block and drop it in a bowl. The box and the bowl are both at the same location at training and evaluation. Note that for this task, playing a demonstration open loop has almost 100% success rate.
- *Grasp and Pour* The robot needs to pick up a bottle and rotate it to point its cap towards the bowl. There are two bottles of similar shape but very different material, one 3D printed and another of plastic (filled with water), that we use for data collection and evaluation. The bowl and bottle position are constant at training and evaluation time. For this task, replaying a demonstration open-loop is successful, but more sensitive to the bottle location. Tiny variations in position will make open-loop fail. However, the trained policies are robust to such small variations.
- *Grasp and Lift* In this task, the robot should grasp one object and lifting it a minimum of 15 cm above the table. There are four very different objects we work on, with different material and shape. In this task, replaying a trajectory does not work since the object to be grasped is not known in advanced. The objects and their location are same at training and test time.

Training Details

- **Ours** We finetune the model with a batch size of 128 using the AdamW optimizer with learning rate set to 1e-4 and weight decay set to 1e-2. For each task, we finetune for 9000 gradient steps and pick the best checkpoint from those collected every 1000 steps. For comparing with our base model trained from scratch, we train the policies trained from scratch with the same configuration as above.
- **Diffusion Policy** For training the diffusion policy baseline, we keep the batch size, optimizer and other parameters same as in [93]. We train diffusion policy for 60,000 gradient steps for each task. We find the policy trained for 15,000 gradients steps to be performing similar to later checkpoints for all tasks.
- **Imagenet Baselines** For the imagenet baselines, we train for 40,000 gradient steps and pick the best checkpoint every 5,000 step with the same optimizer as ours.
- **Visual representation learning baselines** We compare with R3M, MVP and VIP from this line of approaches. For each baseline, we replace the image encoder in our architecture with a frozen instance of one of these visual encoders. For training with RGB images, we apply data augmentation in the same way as in [94]. We train each model for 40000 gradient steps and choose the best checkpoint every 5,000 steps.