VideoDex: Learning Dexterity from Internet Videos

Anonymous Author(s)
Affiliation
Address
email

Abstract: To build general robotic agents that can operate in many environments, it is imperative for the robot to collect experience in the real world. However, this is often not feasible due to safety, time and hardware restrictions. We thus propose leveraging the next best thing as real world experience: internet videos of humans using their hands. Visual priors, such as visual features, are often learned from videos, but we believe that more information from videos can be utilized as a stronger prior. We build a learning algorithm, VideoDex, that leverages *visual*, *action* and *physical* priors from human video data to guide robot behavior. These action and physical priors in the neural network dictate the typical behavior for a particular robot task. We test our approach on a robot arm and dexterous hand based system and show strong results on many different manipulation tasks, outperforming various state-of-the-art methods. For videos and supplemental material visit our website at https://videodex.github.io

Keywords: Dexterous Manipulation, Large Scale Robotics, Imitation Learning

1 Introduction

The long-standing dream of many roboticists is to see robots autonomously perform diverse tasks in diverse environments. To build such robotic agents that can operate anywhere, we need access to a lot of successful robot interaction data in many environments. However, deploying inexperienced, real-world robots to collect experience must require constant supervision which is infeasible. This poses a chicken-and-egg problem for robot learning because to collect experience safely, the robot already needs to be experienced. How do we get around this deadlock?

Fortunately, there is plenty of real-world human interaction videos on the internet. This passive human video data can help bootstrap robot learning by side-stepping the data collection-training loop. This insight of leveraging human videos to aid robotics is not new, and has seen immense attention from the community at large [1, 2, 3]. However, most of the prior work tends to use passive data as mechanism for pretraining just the visual representation [4, 5, 6, 7, 8], much like how deep learning has been used a pretraining tool in related areas of computer vision [9, 10] and natural language processing [11, 12]. Although pretraining visual representations can aid efficiency, most of the inefficiency in robot learning is due to exponentially large action space. For continuous control, this is exponential in the number of actions and timesteps, and the problem gets even worse for high degree-of-freedom robots like dexterous hands (shown in Figure 1).

In this work, we study how to go beyond using passive human videos merely as a source of visual pretraining (i.e. **visual priors**), and leverage the information of how humans move their limbs as a guidance for training how robots should move (i.e. **action priors**). However, human and robots have different embodiment and guiding robot motions using human videos would require: understanding the scene in 3D, figuring out human intent and transferring from human to robot hardware. The first two of these issues can be tackled thanks to success in computer vision. For first, 3D human estimation works decently well in general human videos which we can leverage to gather 3D understanding, and for second, there have been large-scale datasets which breakdown the human intent via crowdsourcing labels [2, 1]. To handle the embodiment transfer, we use human hand to

Submitted to the 6th Conference on Robot Learning (CoRL 2022). Do not distribute.

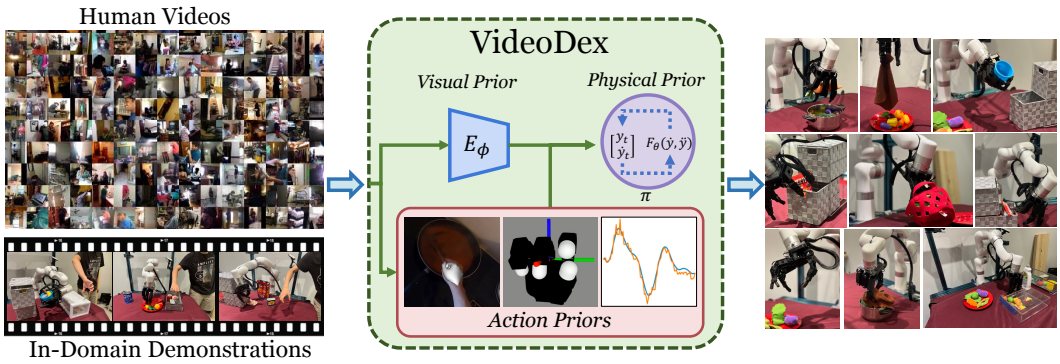

Figure 1: We re-target internet human videos as an action prior, use pretrainined embeddings as a visual prior, and finally use NDPs as a physical prior to complete many different tasks on a robotic hand.

robot hand retargeting as an energy function to pretrain the robot action policy. Our key insight is to combine these visual and action priors from passive data with the physical constraints of how robot should move in the world [13, 14] (i.e., **physical prior**) to obtain dexterous robot policies that can act in the real world. We call this approach, VideoDex. To enhance the performance in the real world, we mix the experience obtained from massive passive data with a few in-domain demonstrations.

In summary, VideoDex is a robot learning algorithm that incorporates visual, action and physical priors into a single policy, learns from passive videos on the internet and then adapts to many different real world tasks with only a few in-domain examples. We find that VideoDex outperforms many state-of-the-art robot learning methods on seven different real-world manipulation tasks in a high dof multi-fingered robotic arm-hand system.

2 Related Work

Learning for Dexterity Human dexterity evolved with cognition in complementary fashion [15]. This leads us to study dexterity to understand cognition in robot agents as well. To create dexterity, reinforcement learning (RL) with an engineered reward function can show good results in simulation [16, 17] but requires lots of data, especially in high dimensional dexterous manipulation. This requires simulators [18, 19], which cannot model physics (such as contact forces) properly, making it hard to transfer to the real world. Imitation learning for manipulation can be safer and more sample efficient. Behavior cloning is a common approach to learning [20, 21] and can work in the real world. DIME [22] involves using nearest neighbor matching of the state or image representations of the scene with that of demonstrations to determine actions. Qin et al. [23] proposes a method for pick-and-place and opening door tasks that involves teleoperation and learning policies in simulation, followed by Sim2Real transfer. DexVIP [24] learns human hand-object affordance using curated internet video datasets, and uses these priors as RL initialization.

Learning from Videos and Large-Scale Datasets There have been many efforts to collect and curate datasets from internet human videos, for example FreiHand [25] for hand poses, 100 Days of Hands [26] for hand-object interactions, Something-Something [3] for semantically similar interactions, Human3.6M [27] and the CMU Mocap Database [28] for Human pose estimation, Epic Kitchens [2], ActivityNet datasets [29] or YouCook [30] for action driven datasets. For dextrous manipulation, activity-based datasets contain well labelled atomic tasks that we would like to solve. Recent works have leveraged passive datasets to learn cost functions [31, 32, 33] and build representations for robot learning [6]. R3M [6] trains on the Ego4D dataset using a temporal alignment loss between language labels and video frames. We build on top of previous efforts in this area, where we combine visual representations trained on internet data, with *action* driven representations.

Learning Action from Videos Detecting humans, estimating poses of different body parts or understanding the dynamics and interactions related to human motion is a commonly studied problem. One can model human hands using the MANO [34] parameterization as well as the human body using SMPL, SMPL-X [35, 36] models. There are many efforts in human pose estimation such as

Figure 2: The collection of train objects (left) and test objects (right) used for experimentation.

[37, 38, 39]. We focus on FrankMocap [39] for our project as it is the most robust for online videos and provides good hand-only estimation. Additionally, using these detectors and watching humans can be powerful for controlling computers. Traditionally, teleoperation approaches have employed hand markers with gloves for motion capture [40] or VR settings [41]. Without gloves, Li et. al. [42] used depth images and a paired human-robot dataset for teleoperating the Shadow Hand, and Handa et. al. [43] designed a system that mimics the functional intent of the human operator to perform object manipulation tasks. Recent works such as [33] aim to learn manipulation from watching human videos in the wild. The robot first detects interaction and human poses and then explores around these priors to improve its policy.

3 Background

3.1 Neural Dynamic Policies

Neural Dynamic Policies (NDPs) [13, 14, 44], can produce safe and smooth trajectories for real world robots. Additionally, they can rollout to trajectories of arbitrary lengths, so using them as a network backbone enables the use of varying length internet videos. NDPs can be described with the Dynamic Movement Primitive equation [45, 46, 47, 48]:

$$\ddot{y} = \alpha(\beta(g - y) - \dot{y}) + f_w(x, g), \tag{1}$$

where y is the coordinate frame of the robot, g is the desired goal in the given coordinate frame, f_w is a radial basis forcing function, x is a time variable, and α, β are global constants. NDPs take the state (robot joints and/or image of the scene) and use a neural network to output the goal g and shape parameters w of the forcing function f_w.

3.2 Learning from Watching Humans

Recently, Sivakumar et al. [49] introduced Robotic Telekinesis, a pipeline that teleoperates the Allegro Hand [50] using a single, uncalibrated RGB camera. Leveraging work in monocular human hand and body pose estimation [39], hand and body modelling [34, 35, 36], and internet data of humans, Robotic Telekinesis can re-target human hand and body poses to robot hand and end effector pose in real time. Due to its efficiency, ease of data collection and setup, we leverage Sivakumar et al. [49]'s approach for our demonstration collection setup.

We borrow the human hand to robot hand retargeting method from Robotic Telekinesis [49] that manually defines key vectors between palms and fingertips on both robot and human hands. These are leveraged to build an energy function E_π which minimizes the the distance between human hand poses (β, θ) (in the MANO [34] parameterization) and the robot hand poses q. Human v_i^h and robot v_i^r keyvectors are obtained from the parameterizations. Therefore, the energy function is defined as:

$$E_\pi(\ (\beta_h, \theta_h),\ q\ ) = \sum_{i=1}^{10} ||v_i^h - (c_i \cdot v_i^r)||_2^2 \tag{2}$$

Here, c_i are scale parameters that are manually chosen dependent on the robot. Sivakumar et al. [49] trains an MLP $\mathrm{HR}(.)$ to implicitly minimize the energy function described in 2, conditioned on knowing human poses (β, θ). For more details, we refer the readers to Sivakumar et al. [49]. Throughout the paper, we refer to this re-targeting approach as $HR(.)$.

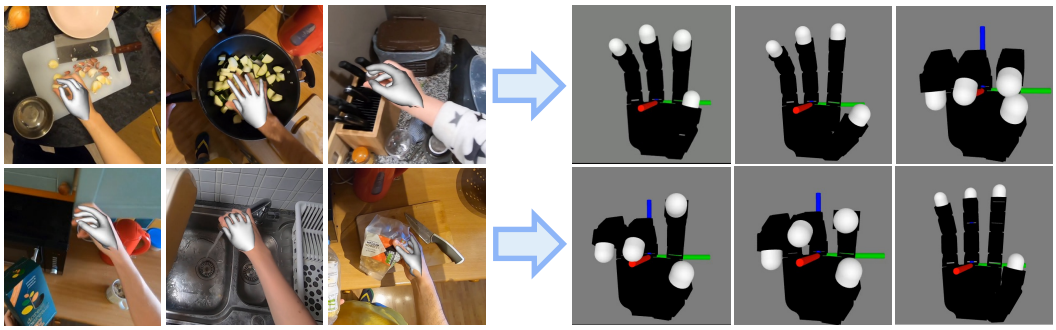

Figure 3: We re-target human hand detections from the 3D MANO model [34] embodiment to the 16 Degree of freedom Robotic Allegro Hand[50] embodiment. Videos at https://videodex.github.io

4 VideoDex: Learning Dexterity from Youtube

We learn general purpose manipulation by utilizing large scale human hand action data. We leverage not only visual priors of the scene, but also leverage important aspects of the human hand's motion, intent and interaction. By pretraining policies with this hand data, we learn *action* priors of the human hand encoded in the network as an action representation. However, it's notoriously difficult to leverage these noisy human video detections. Therefore, we must also employ a policy with *physical* priors to learn smooth and robust policies that do not overfit to noise. Another difficulty is that human arms and hands and robot arms and hands have a very different shape and embodiment. We circumvent this issue by *re-targeting* human data to the robot's point of view. We explain each of these insights used to leverage *action* priors in the sections below.

4.1 Visual Priors from Internet-Scale Data

Many previous works [6, 7] have tackled visual priors and representations for robot learning. These works often use a low dimensional embedding of raw sensory data, such as generative models like in Nair et al. [8]. Recently, large efforts have been made to build universal visual representations which are pretrained networks for downstream learning on general purpose robotics tasks. These often encode some form of semantic visual priors into the pretrained network to help the policy in learning from human videos. We find that instead of training our own visual prior, the encoder from Nair et al. [6] is a useful initialization to our policy. Nair et al. [6] is trained on a visual-language alignment as well as a temporal consistency loss. Our network takes human video frames and processes them using the ResNet18 [51] encoder, E_ϕ from R3M [6] released by the authors. The output of this network is our visual representation for downstream learning.

4.2 Action Priors from Internet-Scale Data

While visual pretraining aids in semantic understanding, human data contains a lot more information about how to interact with the world. VideoDex uses action information to pretrain an action prior, a network initialization that encodes information about the typical actions for a particular task. However, training robot policies on human actions is difficult, as there is a large embodiment gap between human and robot as described in Handa et al. [43] and Sivakumar et al. [49] Thus, we must re-target the motion of the human to the robot embodiment to use it in training. Human hand joints are mapped to the robot fingers, and the global pose of the wrist is mapped to the global pose of the robot arm. This problem is solved using three main components. First, we detect human hands in videos. Second, we project hand poses H to robot finger joints H_r. Finally we convert hand pose P to robot pose P_r. H_r and P_r define the trajectory of the human in the robot's frame, from which we can extract actions to pretrain our network with the action prior. See 4 for a summary of the stages.

Action and Hand Detections First, we must detect the right actions the human is completing. To expedite development, we use the action annotations from the EpicKitchens dataset [2] but an action detection network can easily be used. We clip videos to be of each action the human is completing. Now, we must detect the hand. VideoDex first computes a crop c around the operator's hand using OpenPose [52] and the result is passed to a pose estimator from FrankMocap [39] to obtain hand

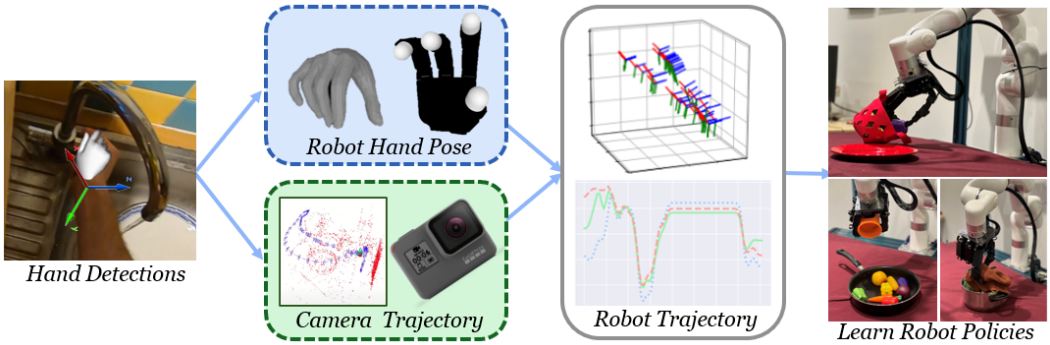

Figure 4: To use internet videos as an action prior for training policies, we must re-target them to the robot embodiment to use them. The detected human fingers are converted to the robot fingers using an energy function. The wrist is re-targeted using the detections and camera trajectory and transformed to the robot arm.

shape (β) and pose parameters (θ) of the 3D MANO model [34]. These parameters are passed through a low pass filter and used in the subsequent sections to re-target the wrist and hand pose.

Re-targeting Wrist Pose In this section, we show how to compute the transformation that describes the wrist pose in the robot frame denoted as M^{Wrist}_{Robot}. First, to calculate $M^{Wrist}_{C_t}$, where C_t is the camera frame at timestep t we leverage the Perspective-n-point algorithm [53]. This takes 2D keypoint outputs (u_i, v_i) by the hand detection model and 3D keypoints from the hand model (x_i, y_i, z_i) and computes $M^{Wrist}_{C_t}$. To accurately obtain camera intrinsics for PnP, COLMAP is used [54]. We call this human wrist 3D trajectory τ_H.

In internet videos, the position of the camera is not always fixed because it is attached to the human's head. Thus, VideoDex deals with moving camera poses as well. Specifically, we compute the transformation between the camera pose in the first frame C_1 and all other frames in the trajectory, C_t. We call this transform $M^{C_t}_{C_1}$. To this end, we run monocular SLAM, specifically ORBSLAM3 [55]. We provide details for this procedure in the Appendix.

Computing poses in the first frame of the camera is important but this is still not in robot frame because the robot is always upright. Many off the shelf cameras with image stabilization save their acceleration data in the metadata and this can be used to compute a transformation between C_1 and the upright world frame, $M^{C_1}_{World}$. For example camera's acceleration data will return $(0, 0, 9.8)m/s$ when upright. We therefore extract the true orientation using the following equations:

$$\text{pitch} = \tan^{-1}(x_{Acc}/\sqrt{y^2_{Acc} + z^2_{Acc}}) \tag{3}$$

$$\text{roll} = \tan^{-1}(y_{Acc}/\sqrt{x^2_{Acc} + z^2_{Acc}}) \tag{4}$$

The pitch and the roll are used to compute $M^{C_1}_{World}$. We only need to extract this for C_1 as we know $M^{C_t}_{C_1}$. Since the robot has workspace limits, and we would also like to center the starting pose of the robot, we heuristically compute T^{World}_{Robot} which rescales the human trajectory in the world frame τ^{wrist}_W into the robot trajectory τ^{wrist}_R. The final function to obtain M^{Wrist}_{Robot} can be described as:

$$M^{Wrist}_{Robot} = T^{World}_{Robot} \cdot M^{C_1}_{World} \cdot M^{C_t}_{C_1} \cdot M^{wrist}_{C_t} \tag{5}$$

Re-targeting Hand Pose Human hands are also in a different *embodiment* compared to that of robot hands, like our 16 DOF Allegro Hand. The goal of this section is to convert joint angles of the human fingers H and project them to the joint angles H_r in the robot's morphology. Similarly, to Sivakumar et al. [49], we use $HR(.)$ to map hand poses to robot hand poses. Given human detected pose x_h, we obtain $x_r = HR(x_h)$ using a similar re-targeting network to Sivakumar et al. [49], and get human hand trajectories: τ^{hand}_R in the robot's embodiment. We use τ_R to denote the combined and and wrist trajectories: τ^{hand}_R, τ^{wrist}_R. See 3 for a visualization.

Trajectories are collected for each human action and are used to train action priors for the corresponding task on the robot. For instance, small pick trajectories are collected from snippets of internet

videos using the action annotations or action detections. These trajectories are retargeted and then used to pretrain the pick action prior on the network.

4.3 Learning with Human Videos

We must design a policy π that learns first from the re-targeted human trajectories (the action prior) and then from real robot trajectories collected in teleoperation. Naively, training a neural network policy on τ_R will lead to overfitting to noisy hand detections. To circumvent this, we first use visual priors from the visual ResNet-based [51] encoder provided by Nair et al. [6], E_ϕ. Then, we introduce a *physical prior* to the network backbone, the physically-aware Neural Dynamic Policies [13, 14].

We construct π with the following setup. We firstly process the scene image I with the visual encoder E_ϕ. Then the extracted features $E_\phi(I)$ are used to condition an NDP for the wrist and hand separately, f_{wrist} and f_{hand}. Concretely, each NDP operates by processing the input features with a small MLP which outputs w, g that are trajectory shape and goal parameters. The forward integrator of the NDP outputs a trajectory for the hand and the wrist, $\hat{\tau}_R$. We use the following loss function:

$$\mathcal{L} = \sum_k \text{Loss}_{L1}(\tau_R - [f_{\text{hand}}(E_\phi(I_k)), f_{\text{wrist}}(E_\phi(I_k))])$$

Algorithm 1 Procedure for VideoDex

Require: Human videos $V^H_{1:K}$ (length T), policy π_θ, demonstrations $\mathcal{D}_{1:N}$. Human detection f_{human} [39].
  for $k = 1...K$ **do**
    for $t = 1...T$ **do**
      Pose parameters $\theta_t, \beta_t = f_{\text{human}}(I_t)$
      Get wrist pose w_t from 3, 4 and 5,
      Hand pose $h_t = HR(\theta_t, \beta_t)$
    end for
    Store all h_t, w_t into robot trajectory τ^k_R

    $\hat{\tau}^k_R = \pi_\theta(I^k_1, h^k_1, w^k_1)$
    Optimize $\mathcal{L}_\theta = ||\tau^k_R - \hat{\tau}^k_R||_1$
  end for
  Store policy weights θ_h to initialize π_θ
  while not converged **do**
    for $n = 1...N$ **do**
      $\tau_n, I^n_{1:T} = \mathcal{D}_n$
      $\hat{\tau}_n = \pi_\theta(I^n_1, h^n_1, w^n_1)$
      Optimize $\mathcal{L}_\theta = ||\tau_n - \hat{\tau}_n||_1$
    end for
  end while

Training Methodology: We collect between 500-3000 video clips of humans completing the same task as the robot will from the Epic Kitchens dataset [2]. For example, in pick, there are close to 3000 video clips of humans picking items. These are retargeted to the robot domain and used to pretrain the network with the human action prior of the pick task. Then, the final policy π is trained on a few teleoperated demonstrations of pick on the real robot. The full training takes about 10 hours on a single 2080Ti GPU. More training details can be found in the appendix and in Algorithm 1. Our network consists of the R3M [6] initialized ResNet-18 [51]. We process these features with a 3 layer MLP with a hidden layer size of 512, which are then processed by 2 NDP [13] networks.

5 Experimental Setup

We perform thorough real world experiments on manipulation tasks, specifically many tasks that require dexterity. See our webpage for videos of these tasks. We aim to answer the following questions. (1) Is VideoDex able to perform general purpose manipulation? (2) How much does the action prior of VideoDex help? (3) How much does the physical prior of VideoDex help? (4) What important design choices are there (visual priors, physical priors or training setup)?

Task Setup To test our approach we design 7 different manipulation tasks. We pretrain 7 action priors on retargeted Epic Kitchens data, one for each task on the robot. Then, we collect about 120 demonstrations per hour for each of these tasks on our setup to use in training the policy. In `pick`, the goal is to pickup an object and we train on 8 objects. In `rotate`, the agent has to grasp and rotate the object in place. We train on 12 objects. In `cover` and `uncover`, the goal is to cover or uncover a pan/plate with a soft object such as a dish cloth or sheet. We train with 4 different deformable objects and 4 different pan/plates. `Push` involves flicking/poking an object with the fingers. We train on 12 different objects. In `place`, the robot has to pick up an object and place it into a plate, pan or pot. There are 8 training objects and 6 different placement objects (plates/pans/pots). In `open` we open three different train drawers. Our testing procedure consists of unseen locations and objects. Details on the tasks are in the appendix.

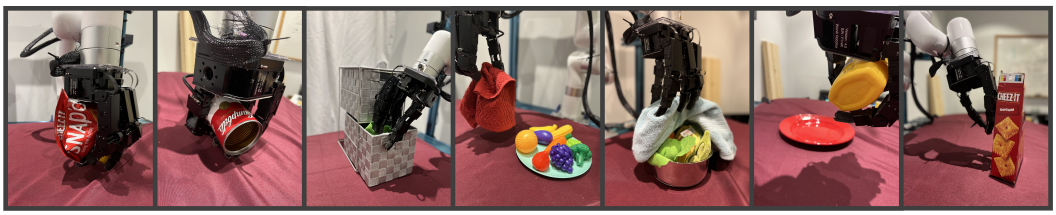

Figure 5: Tasks used in experiments. From left to right: pick, rotate, open, cover, uncover, place and push. See https://videodex.github.io for videos of these tasks.

	Pick		Rotate		Open		Cover		Uncover		Place		Push	
	train	test	train	test	train	test	train	test	train	test	train	test	train	test
BC-NDP [14]	0.64	0.38	**0.94**	0.56	**0.90**	0.60	**0.78**	0.58	0.88	0.82	0.70	0.35	1.00	0.71
BC-Open[44]	0.50	0.44	0.72	0.38	0.80	0.40	0.44	0.58	**1.00**	**0.91**	0.40	0.25	1.00	0.93
BC-RNN [44]	0.56	0.31	0.78	0.50	**0.90**	0.50	0.56	0.42	0.88	0.75	0.70	0.50	1.00	**1.00**
VideoDex	**0.81**	**0.75**	0.89	**0.69**	**0.90**	**0.80**	**0.78**	**0.67**	**1.00**	0.90	**0.90**	**0.70**	1.00	**1.00**

Table 1: We present the results of train objects and test objects for Videodex and baselines described above.

Baselines and Ablations Firstly, we compare the need for initialization with the action priors obtained from θ_h (the result of training the policy on human videos and trajectories). We call this baseline BC-NDP. It uses the same exact physical prior and visual network initialization, without the initialization from θ_h. Secondly, we compare against two standard open-loop behavior cloning approaches introduced in recent benchmarks [44]. BC-open uses a 2 layer MLP instead of the NDP network. BC-RNN, uses an RNN to preprocess the visual features and then a two-stream, 2 layer MLP for wrist and hand trajectories. We try an offline RL ablation CQL [56], where we use the demonstrations as a sparse reward. We train a behavior cloning policy with the action prior from human videos without the physical prior of the NDP. We call this VideoDex-BC-Open. We ablate the type of visual representation/prior use by trying an initialization using the VGG16 network [57] (VideoDex-VGG) and the MVP network [7] [58] (VideoDex-MVP) based representation trained for robot learning. We ablate the need for a two stream policy, instead training a single NDP for both hand and wrist. (VideoDex-Single) To see if VideoDex works with fewer demonstrations (around 50 demonstrations, 5-7 per variant only), we train a policy called VideoDex-Cons.

6 Results

We analyze the results of our experiments and the guiding questions discussed in Section 5. We present the results of our findings as a 0-1 success rate in Table 1 and the result of the ablations we ran on the place task in Table 2.

Effect of Action Priors We firstly compare VideoDex against methods that do not employ a action prior trained on human data, as explained in Section 5. For almost all of the tasks VideoDex either outperforms baselines or has a similar performance, especially for held out objects/instances. We believe that one of the key aspects of VideoDex generalizing to test objects is the action prior pretraining on human videos.

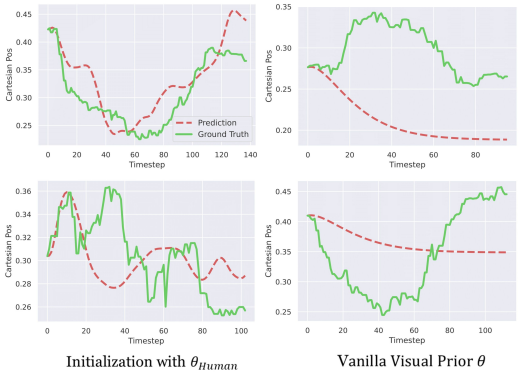

Figure 6: Networks initialized using action priors on internet data without further training are closer to ground truth robot trajectories than networks only initialized using visual priors.

In fact, this can be seen in Figure 6. Without ever training on the robot demonstrations, the trajectories initialized using the action prior pretrained network θ_h (left) are much closer to the ground truth trajectories of a network that is initialized using only a visual prior such as the encoder from Nair et al. [6] (right). From the results, we see that VideoDex-BC-Open with action priors (Table 2)

outperforms BC-Open. Having a physical prior added (BC-NDP) tends to help, but it is not the case for every task. We suspect that some tasks require smoother behavior than others.

Additionally, in Table 2 our offline RL baseline, CQL [56] does not perform as well as the rest of the approaches, even under-performing the Behavior Cloning setup. Qualitatively, we see a much less smooth and less safe execution with this method, thus we only perform it on one task (place). Note that we use the same visual prior for this as well.

	Train	Test
Baselines:		
BC-NDP [14]	0.70	0.35
BC-Open [44]	0.40	0.25
BC-RNN [44]	0.70	0.50
CQL [56]	0.40	0.20
No Physical Prior:		
VideoDex-BC-Open	0.50	0.50
VideoDex-Single	0.50	0.30
Visual Prior Ablation:		
VideoDex-VGG	0.20	0.20
VideoDex-MVP	0.40	0.20
Constrained Data:		
VideoDex-Constrained-5	0.80	0.60
VideoDex-Constrained-10	0.50	0.30
VideoDex **(ours)**	**0.90**	**0.70**

Table 2: We present the results of the ablations discussed in Section 5. These are all performed on the place task.

Effect of Physical Priors and Architectural Choices We compare different types of physical priors in Table 1 and in Table 2. In general (BC-NDP) tends to outperform baselines without a physical prior, except for BC-RNN in a couple of tasks. BC-RNN performs less aggressive behavior, which allowed for it to efficiently grasp more objects. In Table 2 its shown that an important physical prior is to treat the wrist and the hand in a more disentangled manner, as the performance for VideoDex-Single tends to drop compared to BC-NDP and VideoDex-BC-Open (Behavior Cloning with our action prior pretraining). The two stream architecture aids in learning, as it allows the policy to disentangle the actions of the wrist and the hand. This is important as the same grasp might be used for picking objects in many different locations, and similarly, it is possible to localize many objects and perform completely different types of interactions.

Generalization with Less Data We limit VideoDex to a maximum of 5 and 10 teleoperated demonstrations per variant (we have 12-15 variants in our setup). As shown in Table 1, even with 5 instances per variant, we still see a 30% success rate for unseen objects. Empirically, the policies generally go to the right area, but are not able to grasp objects properly. VideoDex works with less robot experience, showing that action priors allow not only higher performance, but higher sample efficiency as well.

Effect of Visual Priors We compared using our approach with MVP (VideoDex-MVP) [7] and VGG (VideoDex-VGG) [57] and their performance was below VideoDex using [6]. This is likely because both encoders are much larger than the ResNet18 [51] we use and require a lot more training time than feasible on human videos. However, VideoDex-MVP still performs better than VideoDex-VGG, which indicates that using a visual prior trained on human data does in fact help, as Xiao et al. [7] trained the representation in self-supervised fashion on videos and use the embeddings to perform robotics tasks in simulation. We see in Table 1, that while visual priors are important, action priors are in fact more impactful.

7 Discussion and Limitations

Robot learning promises to be able to generalize to a broad range of scenarios. However, this promise is often limited by the amount of experience the robot has, which can be difficult to collect. Our key insight is that the internet contains a massive corpus of rich and diverse human hand videos. We re-target this internet-scale data to be used as a action prior on the robot. In the future, we believe that internet-scale videos of humans will be crucial in completing robotics tasks.

Although we see strong results on the held-out objects, VideoDex has several limitations and scope for future work. First, we rely on off the shelf human hand detection modules that very often have erroneous 6D pose detections, especially when the hand is interacting with objects. Second, the actions priors rely on the arm trajectory as well as the hand trajectory retrageting which must be recomputed for each different set of robot parameters and embodiments. Finally, our method of behavior cloning in the real world is currently open loop, so it cannot react to changes in the environment. This is because training RL in the real world is difficult due to hardware limitations. We leave this to future work, to train policies that can react to changes in the real-world.

References

[1] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18995–19012, 2022.

[2] D. Damen, H. Doughty, G. M. Farinella, S. Fidler, A. Furnari, E. Kazakos, D. Moltisanti, J. Munro, T. Perrett, W. Price, and M. Wray. Scaling egocentric vision: The epic-kitchens dataset. In *European Conference on Computer Vision (ECCV)*, 2018.

[3] R. Goyal, S. Ebrahimi Kahou, V. Michalski, J. Materzynska, S. Westphal, H. Kim, V. Haenel, I. Fruend, P. Yianilos, M. Mueller-Freitag, F. Hoppe, C. Thurau, I. Bax, and R. Memisevic. The "something something" video database for learning and evaluating visual common sense. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[4] L. Pinto, D. Gandhi, Y. Han, Y.-L. Park, and A. Gupta. The curious robot: Learning visual representations via physical interactions. In *ECCV*, 2016.

[5] P. Sermanet, C. Lynch, Y. Chebotar, J. Hsu, E. Jang, S. Schaal, and S. Levine. Time-contrastive networks: Self-supervised learning from video. In *ICRA*, 2018.

[6] S. Nair, A. Rajeswaran, V. Kumar, C. Finn, and A. Gupta. R3m: A universal visual representation for robot manipulation. *arXiv preprint arXiv:2203.12601*, 2022.

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

[8] A. V. Nair, V. Pong, M. Dalal, S. Bahl, S. Lin, and S. Levine. Visual reinforcement learning with imagined goals. In *NeurIPS*, pages 9191–9200, 2018.

[9] K. He, G. Gkioxari, P. Dollar, and R. Girshick. Mask r-cnn. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[10] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton. A simple framework for contrastive learning of visual representations. In H. D. III and A. Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 1597–1607. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/chen20j.html.

[11] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. 2020.

[12] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[13] S. Bahl, M. Mukadam, A. Gupta, and D. Pathak. Neural dynamic policies for end-to-end sensorimotor learning. In *NeurIPS*, 2020.

[14] S. Bahl, A. Gupta, and D. Pathak. Hierarchical neural dynamic policies. *RSS*, 2021.

[15] R. R. Ma and A. M. Dollar. On dexterity and dexterous manipulation. In *2011 15th International Conference on Advanced Robotics (ICAR)*, pages 1–7, 2011. doi:10.1109/ICAR.2011.6088576.

[16] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, et al. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. *arXiv preprint arXiv:1806.10293*, 2018.

[17] S. Levine, C. Finn, T. Darrell, and P. Abbeel. End-to-end training of deep visuomotor policies. *JMLR*, 2016.

[18] E. Todorov, T. Erez, and Y. Tassa. MuJoCo: A physics engine for model-based control. In *IROS*, 2012.

[19] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, et al. Isaac gym: High performance gpu-based physics simulation for robot learning. *arXiv preprint arXiv:2108.10470*, 2021.

[20] D. A. Pomerleau. Alvinn: An autonomous land vehicle in a neural network. In D. Touretzky, editor, *Advances in Neural Information Processing Systems*, volume 1. Morgan-Kaufmann, 1988. URL https://proceedings.neurips.cc/paper/1988/file/812b4ba287f5ee0bc9d43bbf5bbe87fb-Paper.pdf.

[21] M. Bojarski, D. Del Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba. End to end learning for self-driving cars, 2016. URL https://arxiv.org/abs/1604.07316.

[22] S. P. Arunachalam, S. Silwal, B. Evans, and L. Pinto. Dexterous imitation made easy: A learning-based framework for efficient dexterous manipulation, 2022. URL https://arxiv.org/abs/2203.13251.

[23] Y. Qin, H. Su, and X. Wang. From one hand to multiple hands: Imitation learning for dexterous manipulation from single-camera teleoperation, 2022. URL https://arxiv.org/abs/2204.12490.

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

[25] C. Zimmermann, D. Ceylan, J. Yang, B. Russell, M. Argus, and T. Brox. Freihand: A dataset for markerless capture of hand pose and shape from single rgb images. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 813–822, 2019.

[26] D. Shan, J. Geng, M. Shu, and D. F. Fouhey. Understanding human hands in contact at internet scale. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9869–9878, 2020.

[27] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu. Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE transactions on pattern analysis and machine intelligence*, 36(7):1325–1339, 2013.

[28] Cmu graphics lab motion capture database. http://mocap.cs.cmu.edu/.

[29] B. G. Fabian Caba Heilbron, Victor Escorcia and J. C. Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *CVPR*, pages 961–970, 2015.

[30] P. Das, C. Xu, R. F. Doell, and J. J. Corso. A thousand frames in just a few words: Lingual description of videos through latent topics and sparse object stitching. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2634–2641, 2013.

[31] L. Shao, T. Migimatsu, Q. Zhang, K. Yang, and J. Bohg. Concept2robot: Learning manipulation concepts from instructions and human demonstrations. *The International Journal of Robotics Research*, 40(12-14), 2021.

[32] A. S. Chen, S. Nair, and C. Finn. Learning generalizable robotic reward functions from" in-the-wild" human videos. *arXiv preprint arXiv:2103.16817*, 2021.

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

[34] J. Romero, D. Tzionas, and M. J. Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 36(6), Nov. 2017.

[35] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black. Smpl: A skinned multi-person linear model. *ACM transactions on graphics (TOG)*, 34(6):1–16, 2015.

[36] G. Pavlakos, V. Choutas, N. Ghorbani, T. Bolkart, A. A. A. Osman, D. Tzionas, and M. J. Black. Expressive body capture: 3D hands, face, and body from a single image. In *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 10975–10985, 2019.

[37] J. Wang, F. Mueller, F. Bernard, S. Sorli, O. Sotnychenko, N. Qian, M. A. Otaduy, D. Casas, and C. Theobalt. Rgb2hands: real-time tracking of 3d hand interactions from monocular rgb video. *ACM Transactions on Graphics (TOG)*, 39(6):1–16, 2020.

[38] A. Kanazawa, M. J. Black, D. W. Jacobs, and J. Malik. End-to-end recovery of human shape and pose. *CoRR*, abs/1712.06584, 2017. URL http://arxiv.org/abs/1712.06584.

[39] Y. Rong, T. Shiratori, and H. Joo. Frankmocap: A monocular 3d whole-body pose estimation system via regression and integration. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV) Workshops*, pages 1749–1759, October 2021.

[40] S. Han, B. Liu, R. Wang, Y. Ye, C. D. Twigg, and K. Kin. Online optical marker-based hand tracking with deep labels. *ACM Transactions on Graphics (TOG)*, 37(4):1–10, 2018.

[41] V. Kumar and E. Todorov. Mujoco haptix: A virtual reality system for hand manipulation. In *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pages 657–663, 2015. doi:10.1109/HUMANOIDS.2015.7363441.

[42] S. Li, X. Ma, H. Liang, M. Görner, P. Ruppel, B. Fang, F. Sun, and J. Zhang. Vision-based teleoperation of shadow dexterous hand using end-to-end deep neural network. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 416–422. IEEE, 2019.

[43] 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 *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9164–9170, 2020. doi:10.1109/ICRA40945.2020.9197124.

[44] S. Dasari, J. Wang, J. Hong, S. Bahl, Y. Lin, A. S. Wang, A. Thankaraj, K. S. Chahal, B. Calli, S. Gupta, et al. Rb2: Robotic manipulation benchmarking with a twist. In *NeurIPS Datasets and Benchmarks Track (Round 2)*, 2021.

[45] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal. Dynamical movement primitives: Learning attractor models for motor behaviors. *Neural Computation*, 2013.

[46] M. Prada, A. Remazeilles, A. Koene, and S. Endo. Dynamic movement primitives for human-robot interaction: Comparison with human behavioral observation. In *International Conference on Intelligent Robots and Systems*, 2013.

[47] S. Schaal. Dynamic movement primitives-a framework for motor control in humans and humanoid robotics. In *Adaptive motion of animals and machines*. Springer, 2006.

[48] P. Pastor, H. Hoffmann, T. Asfour, and S. Schaal. Learning and generalization of motor skills by learning from demonstration. In *ICRA*, 2009.

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

[50] Allegro hand. https://www.wonikrobotics.com/research-robot-hand.

[51] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015. URL http://arxiv.org/abs/1512.03385.

[52] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh. Openpose: realtime multi-person 2d pose estimation using part affinity fields. *IEEE transactions on pattern analysis and machine intelligence*, 43(1):172–186, 2019.

[53] M. A. Fischler and R. C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, jun 1981. ISSN 0001-0782. doi:10.1145/358669.358692. URL https://doi.org/10.1145/358669.358692.

[54] J. L. Schönberger, E. Zheng, M. Pollefeys, and J.-M. Frahm. Pixelwise View Selection for Unstructured Multi-View Stereo. In *European Conference on Computer Vision (ECCV)*, 2016.

[55] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós. Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. *IEEE Transactions on Robotics*, 37(6):1874–1890, 2021.

[56] A. Kumar, A. Zhou, G. Tucker, and S. Levine. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems*, 33:1179–1191, 2020.

[57] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[58] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022.