Dexterous Functional Grasping

Anonymous Author(s)
Affiliation
Address
email

Figure 1: We use a single policy trained in simulation to pickup and grasp objects like hammers, drills, saucepan, staplers and screwdriver in different positions and orientations. An affordance model based on matching DINOv2 features is used to localize the object and move above the relevant region of the object. A blind reactive policy then picks up the object and moves it inside the palm to a firm grasp so that post-grasp motions like drilling, hammering, etc can be executed. Videos at https://dexfunc.github.io/.

Abstract: While there have been significant strides in dexterous manipulation, most of it is limited to benchmark tasks like in-hand reorientation which are of limited utility in the real world. The main benefit of dexterous hands over two-fingered ones is their ability to pickup tools and other objects (including thin ones) and grasp them firmly in order to apply force. However, this task requires both a complex understanding of functional affordances as well as precise low-level control. While prior work obtains affordances from human data this approach doesn't scale to low-level control. Similarly, simulation training cannot give the robot an understanding of real-world semantics. In this paper, we aim to combine the best of both worlds to accomplish functional grasping for in-the-wild objects. We use a modular approach. First, affordances are obtained by matching corresponding regions of different objects and then a low-level policy trained in sim is run to grasp it. We propose a novel application of eigengrasps to reduce the search space of RL using a small amount of human data and find that it leads to more stable and physically realistic motion. We find that eigengrasp action space beats baselines in simulation and outperforms hardcoded grasping in real and matches or outperforms a trained human teleoperator. Videos at https://dexfunc.github.io/.

Keywords: Functional Grasping, Tool Manipulation, Sim2real

Submitted to the 7th Conference on Robot Learning (CoRL 2023). Do not distribute.

1 Introduction

The human hand has played a pivotal role in the development of intelligence – dexterity enabled humans to develop and use tools which in turn necessitated the development of cognitive intelligence. [1, 2, 3, 4, 5] Dexterous manipulation is central to the day-to-day activities performed by humans ranging from tasks like writing, typing, lifting, eating, or tool use to perform end tasks. In contrast, the majority of robot learning research still relies on using two-fingered grippers (usually parallel jaws) or suction cups which makes them restricted in terms of the kind of objects that can be grasped and how they can be grasped. For instance, grasping a hammer using a parallel jaw gripper is not only challenging but also inherently unstable due to the center of mass of the hammer being close to the head, which makes it impossible to use it for the hammering function it is intended for. Although there are lots of recent works in learning control of dexterous hands, they are either limited to simple grasping or the tasks of in-hand reorientation [6, 7, 8, 9, 10, 11] which ignore the functional aspect of picking the object for tool use.

This paper investigates the problem of functional grasping of such complex daily life objects using a low-cost dexterous multi-fingered hand. For instance, consider the sequence of events that take place when one uses a hammer. First, the hammer must be detected and localized in the environment. Next, one must position their hand in a suitable pose perpendicular to the handle such that a suitable grasp pose may be initiated. A hammer may be feasibly grasped from both the hammer or the head and choosing the correct pose (also known as *pre-grasp pose*) requires an understanding of how hammers work. Next, the actual grasping motion is executed which is a high-dimensional closed-loop operation involving first picking up the hammer from the table and then moving it with respect to the hand into a firm power grasp. Power grasp is essential to ensure the stability of the hammer during usage. Once this is done, the arm can then execute the hammering motion while the hand holds it stably (*post-grasp trajectory*). Notably, the act of functional grasping, which is almost a muscle memory for humans, is not just a control problem but lies at the intersection of perception, reasoning, and control. How to do it seamlessly in a robot is the focus of our work.

Inspired by the above example, we approach the problem of functional grasping in three stages: predicting pre-grasp, learning low-level control of grasping, post-grasp trajectory. Out of these stages, visual reasoning is the critical piece of the first and third stage, while the second stage can be performed blind using proprioception as long as the pre-grasp pose is reasonable. To obtain the pre-grasp pose, we use a one-shot affordance model that gives pre-grasp keypoints for different objects in different orientations. However, a more challenging problem is how to learn the low-level control for grasping task itself.

We take a sim2real approach for grasping. Prior approaches to sim2real have shown remarkable success for in-hand reorientation [7, 6] and locomotion [12, 13, 14]. In contrast, grasping typically involves continuous surface contacts and high forces which pose a significant sim2real gap and are nontrivial to engineer reward for. We introduce an action compression scheme to leverage a small amount of human demo data to reduce the action space of the hand from 16 to 9 and constrain it to output physically realistic poses. We evaluate our approach across 7 complex tasks in both the real world and simulation and find that our approach is able to make significant progress towards this major challenge of dexterous functional grasping as illustrated in Figure 1.

2 Method: Dexterous Functional Grasp

In this paper, we aim to combine the best of both human data and large scale simulation training to accomplish dexterous functional grasping in the real world. Given an object to grasp we use an affordance model to predict a plausible *functional* grasp pose for the hand. Then, we train a blind pickup policy to pickup the object and then grasp it tightly so that the arm may execute the post-grasp trajectory. Our method is divided into three phases - the pre-grasp, grasp and post-grasp (see Fig. 2)

In the pre-grasp phase, an affordance model outputs a region of interest of the object and we use the local object geometry around that region to compute a reasonable pre-grasp pose. We train a sim2real

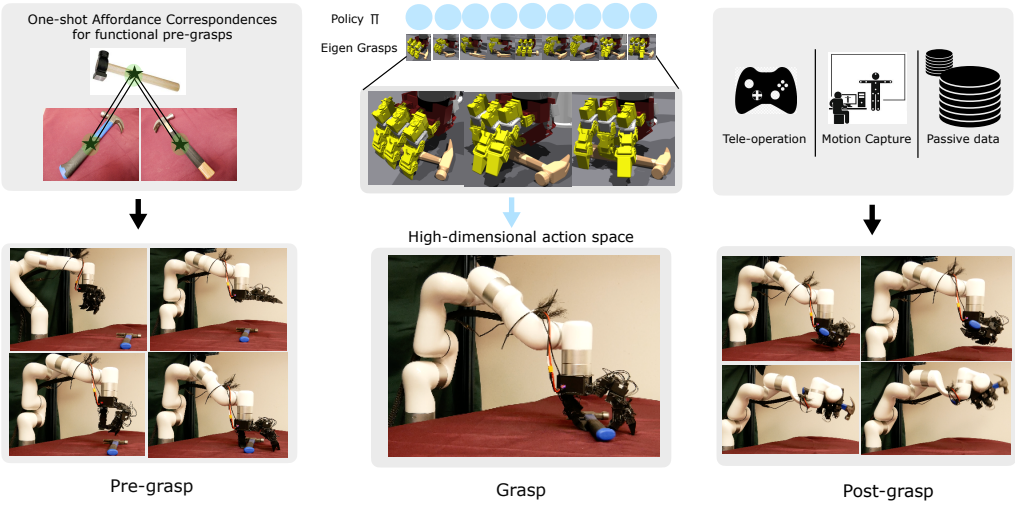

Figure 2: To get the pre-grasp pose we use a one-shot affordance model. After annotating one object we are able to get affordances for other objects in that category via feature matching. Given a new object, the arm is moved to that point and oriented perpendicular to the principal component of the object mask. The sim2real pickup policy is then executed and moves the object into a power grasp. After this, a post grasp trajectory can be safely executed.

policy to execute robust grasps for pickup. However, in contrast to two fingered manipulation or locomotion where simple reward functions suffice, in the complex high-dimensional dexterous case it is easy to fall into local minima or execute poses in simulation that are not realizable in the real world. We therefore use human data to extract a lower-dimensional subspace of the full action space and run RL inside the restricted action space. Empirically, this leads to physically plausible poses that can transfer to real and stabler RL training.

2.1 Pre-grasp pose from affordances

An affordance describes a region of interest on the object that is relevant for the purpose of using it. This usually cannot be inferred from object geometry alone and depends upon the intended proper use of the object. For instance, by just looking at the geometry or by computing grasp metrics we could conclude that grabbing a hammer from the head or handle are both equally valid ways of using it. However, because we have seen other people use it we know that the correct usage is to grab the handle. This problem has been studied in the literature and one approach is to use human data in the form of videos, demos to obtain annotations for affordances. However, these are either not scalable or too noisy to enable zero-shot dexterous grasping.

Another approach is to leverage the fact that affordances across objects usually correspond. For all hammers, no matter the type the hammering, affordance will always be associated with the handle. This implies that feature correspondence can be used in a one-shot fashion to obtain affordances. In particular, we use Hadjivelichkov et al. [15], where for each object category we annotate one image from the internet with its affordance mask. To obtain the affordance mask for a new object instance we simply match DINO-ViT features to find the region which matches the specified mask. Since the mask may bleed across the object boundary we take its intersection with the segment obtained using DETIC [16]. Taking the center of the resulting mask gives us the keypoint $(x_{\text{img}}, y_{\text{img}})$ in image space corresponding to the pre-grasp position. To get the z_{img}, we project to the points $(x_{\text{img}}, y_{\text{img}})$ into the aligned depth image and then transform by camera intrinsics and extrinsics to get the corresponding point in the coordinate frame of the robot $(x_{\text{robot}}, y_{\text{robot}}, z_{\text{robot}})$. To get the correct hand orientation $\mathbf{q}$ we use the object mask obtained from DETIC and take the angle perpendicular to its largest principal component. We first move the hand to a point at fixed offset $(x_{\text{robot}}, y_{\text{robot}}, z_{\text{robot}} + \delta z)$ during pre-grasp. We then move the finger joints to a pre-grasp pose with the joint positions midway between

their joint limits. We found that same pre-grasp pose to work well across objects since our policy learns to adapt to the inaccuracies in the pre-grasp.

2.2 Sim2real for dexterous grasping

Once the robot is in a plausible pre-grasp pose it must execute the grasp action which involves using the fingers to grip the object and then moving it into a stable grasp pose. This requires high frequency closed-loop control. Further, this is typically a locally reactive behavior which can be accomplished using proprioception alone. Indeed, once we move our hand close to the object we wish to grasp we can usually pick it up even if we close our eyes. However, the challenge is that learning high frequency closed loop behavior typically requires a lot of interaction data which is missing from human videos and infeasible to scale via demos. In the past, sim2real has had remarkable successes in locomotion and in-hand dexterous manipulation in learning robust and reactive policies and we propose to use this method here.

Dexterous manipulation however presents a unique challenge because of its high-dimensional nature. It is easy for the hand to enter physically inconsistent poses or experience self collisions. Further, RL in high dimensional action spaces is unstable or sample efficient. We propose to leverage a small amount of human data to restrict the action space to physically realistic poses.

Eigengrasp action space A small number of human demos are often used to guide RL towards reasonable solutions like offline RL [17], DAPG [18]. However, the main problem with these is that they fail to learn optimal behavior from highly suboptimal demos. Further, the coverage of the demo data may be very poor which can artificially restrict the exploration space of the RL algorithm. We propose a simple alternative to these approaches which works from a few demos and can discover optimal behaviors even from suboptimal data. Our insight is that we have a very weak constraint on the behavior of the RL policy. We only care that the individual hand poses are realistic and not so much about the exact sequence in which they occur. We can therefore restrict the action space such that only realistic hand poses are possible.

In particular, suppose we are given a mocap dataset $\mathcal{D} = \{\tau_1, \ldots, \tau_n\}$ where $\tau_i = (\mathbf{x}_1, \ldots, \mathbf{x}_k)$ and $\mathbf{x}_i \in \mathbb{R}^{16}$ is a set of joints angles of the 16 dof hand. We perform PCA on the set of all hand poses to get 9 eigenvectors $\mathbf{e}_1, \ldots, \mathbf{e}_m$ where $m = 9$. These vectors are called eigengrasps [19] and have been classically used in grasp synthesis approaches. Here, we instead use it as a compressed action space for RL. Our policy predicts m-dimensional actions $\pi(\mathbf{o}_t) = \mathbf{a}_t \in \mathbb{R}^m$. The raw joint angles are then computed as a linear combination of eigenvectors $(\mathbf{a_t})_1\mathbf{e}_1 + \ldots + (\mathbf{a_t})_k\mathbf{e}_k$. This transformation reduces the action dimension of the RL problem and decreases sample complexity in addition to enforcing realism. It also exploits the property that the convex combination of any two realistic hand poses is also likely to be realistic. Thus, doing PCA (as opposed to training a generative model) allows the policy to output hand poses that were not seen in the dataset. Empirically, we find that this stabilizes training and minimizes variation between different random seeds.

Rewards We train our policy to lift objects off the ground and them firmly grasp them in their hand. We find that a simple reward function that is a combination of two terms $r_{\text{threshold}}$ and $r_{\text{hand-obj}}$ is enough. The first, is a binary signal incentivizing the policy to pickup the object

$$r_{\text{threshold}}(t) = \mathbb{I}\left[(\mathbf{r}_{\text{obj}}(t))_z \geq 0.04\text{cm}\right]$$

and the second is a sum of exponentials and an L2 distance to incentivize the object to be close to the palm of the hand

$$r_{\text{hand-obj}}(t) = \sum_{i=1}^{3} \exp\left(-\frac{\|\mathbf{r}_{\text{obj}} - \mathbf{r}_{\text{hand}}\|}{d_i}\right) - 4\|\mathbf{r}_{\text{obj}} - \mathbf{r}_{\text{hand}}\|$$

where $d_1 = 10$cm, $d_2 = 5$cm and $d_3 = 1$cm. The overall reward function is $r(t) = r_{\text{hand-obj}}(t) + 0.1 \cdot r_{\text{threshold}}(t) + 1$. Due to the eigengrasp parameterization we do not need any additional reward shaping terms.

Policy Architecture We use a recurrent policy as that maps observations $\mathbf{o}_t \in \mathbb{R}^{16}$ to actions $\mathbf{a}_t \in \mathbb{R}^9$. A stateful policy is able to adapt to changes in environment dynamics better than a feedforward one. This allows our robot to adapt to slight errors in the pre-grasp pose from the affordance model. The policy observes the 7 dimensional target pose (position, quaternion) of the end-effector and the 16 joint angle positions of the hand.

Training environment We want our policy to be robust to different surface properties and geometries and grasp them firmly. We therefore domain randomize the physical properties of the object, robot and simulation environment. We procedurally generate a set of hammers in simulation with randomized physical parameters. The hand is initialized in a rough pre-grasp pose with hand joint angles zeroed out. This corresponds to a neutral relaxed pose for the hand. The end-effector pose is initialized to be close to the real world pose obtained from the affordance model. The arm is kept close to the ground for 1s to allow the grasp to execute and then spun around in a circle. Episodes are terminated if the hand object distance exceeds 20cm. This spinning motion produces tight grasps and we see emergent behavior where the hand adjusts its grasp in response to changes in orientation. We also randomize physical properties of the simulation and add gaussian noise to observations and actions to simulate actuator noise.

2.3 Post-grasp trajectory

Once the object or tool is firmly grasped, since it is mounted on a 6-dof arm it can be moved arbitrarily in space to accomplish tasks such as screwing, hammering, drilling, etc. During training and evaluation we use either motion capture trajectories or define a set of keypoints and interpolate between them, but in principle these could be obtained from other sources such as internet video or third person imitation.

3 Experimental Setup

We demonstrate the performance of our method on a variety of objects, both similar and very dissimilar to the training objects like stapler, drill (light and heavy), saucepan, hammer (light and heavy). In our real world experiments, we aim to understand the reliability and efficiency of our method relative to an expert *teleop oracle* (20 hours) and a *hardcoded* grasping primitive. The former acts as an upper bound on the performance of the hardware while the latter is designed to show that large scale sim training yields a more robust policy than a grasping hardcoded.

In simulation, we test the effectiveness of our restricted action space and policy architecture. First, we compare against an *unconstrained* baseline that operates in the full 16 dimensional action space. Second, we compare against a policy that operates in the latent space of a *VAE* trained on the mocap dataset. Unlike our method, since a VAE is a generative model it can only output hand poses seen in the dataset and cannot extrapolate to new ones. Finally, we compare to a *feedforward* version of our method where the RNN policy is replaced by a feedforward one. This is designed to test whether recurrence helps in adaptation to domain randomization.

3.1 Hardware

We use the xarm6 with our own custom hand pictured in Fig. 3. The arm has 6 actuated joints, while the hand has 16 joints, four on each digit (three fingers and one thumb). An overhead calibrated D435 camera facing downward is used to obtain masks and affordance regions. The hand consists of Dynamixel servos mounted in a special kinematic structure designed to maximize dexterity. We use an overhead D435 camera to obtain pre-grasp end-effector poses. Both the arm and the hand run at 30Hz. To teleoperate the hand and collect human demos for eigengrasps we use a Manus VR glove with SteamVR lighthouses which gives fingertip and hand positions which are then retargeted to our hand as in Figure 4.

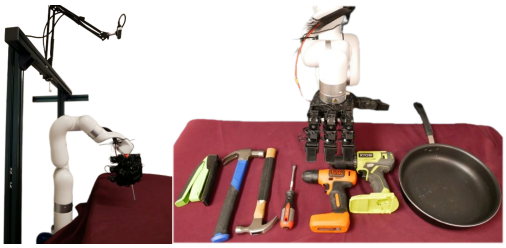

Figure 3: (Right) we grasp a variety of household objects and tools. (left) our hardware setup consists of an xarm6 mounted on our custom hand with a down-facing D435.

Figure 4: (left) the Manus VR glove we use to teleoperate our hand (right) the hand in the retargeted pose.

	Average Reward			Success Rate		
	Hammer	Drill	Screwdriver	Hammer	Drill	Screwdriver
Unconstrained	213.40 ± 169.37	102.12 ± 36.12	121.28 ± 96.05	0.60 ± 0.55	0.09 ± 0.11	0.46 ± 0.45
VAE	140.60 ± 109.24	83.34 ± 43.32	117.25 ± 76.26	0.30 ± 0.44	0.08 ± 0.18	0.25 ± 0.41
Feed-forward	232.80 ± 175.59	104.61 ± 44.84	153.19 ± 105.83	0.60 ± 0.54	0.21 ± 0.19	0.56 ± 0.52
Ours	$\mathbf{327.40 \pm 11.61}$	$\mathbf{129.03 \pm 22.58}$	$\mathbf{211.13 \pm 11.14}$	$\mathbf{1.00 \pm 0.00}$	$\mathbf{0.23 \pm 0.16}$	$\mathbf{0.95 \pm 0.10}$

Table 1: We measure the average reward and success rate of the trained policy in simulation. For each method we train a policy to hold the object close to the palm while arm spins. A success is counted when the arm does not drop the object at anytime. We see that our method outperforms the baselines and has significantly less variation between the runs. This is likely because the restricted action space makes the exploration problem easier and the physically plausible poses help keep the motion smooth. Each policy was trained randomized hammer but still generalizes to other different objects.

3.2 Implementation Details

We use IsaacGym as a simulator with IsaacGymEnvs for the environments and rl_games as the reinforcement learning library. The policy contains a layer-normed GRU with 256 as the hidden state followed by an MLP with hidden states 512, 256, 128. The policy is trained using PPO with backpropagation through time truncated at 32 timesteps. We run 8192 environments in parallel and train for 400 epochs.

4 Results and Analysis

4.1 Simulation Results

We train each baseline and our method for 400 epochs over 5 seeds. We find that ours beats all other methods primarily because it is stable with respect to the seed whereas the other baselines fluctuate widely in performance across seeds resulting in a high standard deviation and lower average overall performance. Note that our method also perfectly solves the training task for all seeds. This is likely due to a combination of two factors (a) the restricted action space nearly halves the action dimension (from 16 to 9), since the search space scales exponentially with action dimension this cuts down the space significantly and it is more likely that the algorithm discovers optimal behavior regardless of seed, and (b) since each hand pose is realistic

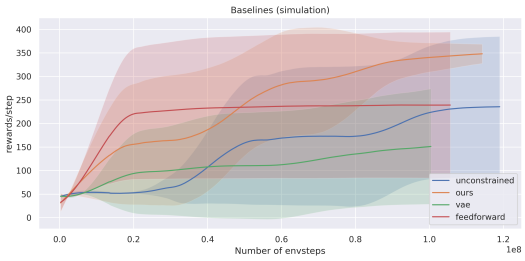

Figure 5: Training curves for baselines in simulation. Each baseline is run over 5 seeds. We see that ours outperforms the other baselines and also is more stable with respect to the seed. This is because of the lower dimensional action space.

and doesn't have self-collisions it leads to smoother and more predictable dynamics in simulation allowing the policy to learn better.

The RNN policy is also better and more stable than the feedforward variant as reported in Table 1. This is because (a) an RNN can use the hidden state to adapt to domain randomization (b) since the hand hardware does not output joint velocities, the feedforward policy has no idea of how fast the fingers are moving which can hinder performance. The RNN on the other hand is able to implicitly capture velocity of joints in the hidden state and this helps it to learn better.

	Success Rate ↑		
	Teleop Oracle	Hardcoded	Ours
Hammer (heavy)	0.5	0.0	0.8
Hammer (light)	0.6	0.3	0.9
Sauce pan	0.9	0.3	0.9
Drill (heavy)	0.9	0.2	0.5
Drill (light)	0.9	0.3	0.8
Stapler	0.9	0.3	1.0
Screwdriver	0.5	0.0	0.7

Table 2: We show functional grasping for a varied set of objects. We compare to a hardcoded pinch grasp and a trained teleoperator with a VR glove. The hardcoded baseline fails since the fingers push the object behind. Our method is able to beat the teleop oracle on challenging objects such as screwdriver, stapler and hammer.

4.2 Real World Results

We choose a variety of objects to compare against – hammer (light and heavy), saucepan, drill (light and heavy), stapler and screwdriver. Of these, hammer and saucepan are quite similar to the training distribution because of the handle geometry while the drill, stapler and screwdriver have substantially different geometry. The heavy drill is especially challenging because of its narrow grip and unbalanced weight distribution. We run 10 trials per object per baseline in the real world (see Table 2). For all objects except the saucepan we execute a post-grasp trajectory where the object is picked up and waved around to test the strength of the grasp. For the saucepan we simply pick it up since waving it around is a safety hazard. During each trial, the orientation is randomized in the range $[-\pi, \pi]$ and position is randomized in the entire workspace $1\text{m} \times 0.5\text{m}$, the affordance model is run and the hand is moved to the pre-grasp pose. Qualitative results can be found at `https://dexfunc.github.io/`.

We obtain the hardcoded baseline by interpolating between the fully open and fully closed eigengrasp over 1s. This leads to the hand quickly snapping shut before the arm rises up. We find that this baseline performs poorly and gets zero success rate on many objects, especially thin ones. This is because in order to successfully grasp the object the thumb must retract closer to the palm. However, the timing of this is crucial, if the thumb retracts too early then the object flies back away from the hand. This is the most common failure case of this baseline that we observe. The hardcoded grasp succeeds for tall objects like an upright stapler or if the object happens to be in a favorable pose at the time of grasping.

The teleop oracle baseline was carried out with a Manus VR glove with the joints mapped one to one to the robot hand (ignoring the human pinky). This was teleoperated by a trained user (20 hours of experience). This was intended to serve as an upper bound of hardware capability. We find our method matches or slightly lags behind the oracle for drill (light) and saucepan. Surprisingly, for stapler, screwdriver and both hammers it even exceeds the oracle baseline. This is because these objects are heavy and sit close to the ground and require very swift and forceful motion which is also very precise in order to be successfully picked out. This is very hard to execute reliably for a human being, whereas our policy is able to do it well. We also find that our method is able to complete the task in a shorter time for the same reason.

5 Related Work

In-hand dexterous manipulation: Dexterity in humans is the ability to manipulate objects within their hand's workspace [20, 21, 22]. Accordingly, in-hand reorientation has remained a standard, yet challenging task in robotics to imitate a human's dexterity. In recent years, there has been a surge of interest in this field and sim2real approaches have shown some success at reorienting objects [7, 23, 9, 24, 8] and also manipulating them [6, 25]. Other works bypass sim and directly learn

in-hand manipulation through trial and error in the real world [11, 10]. Some other works use human demos to guide RL [18] and others directly use demos to learn policies [26].

Dexterous grasping: While in-hand reorientation is an important task most of the use cases of a dexterous hand involve grasping objects in different poses. Because of the large degrees of freedom, grasp synthesis is significantly more challenging. The classical approach is to pose this as an optimization problem [19, 27, 28]. This approach is still used today with the form or force closure objective. [29, 30, 31] Some methods use the contact between the object and the hand as a way to learn proper grasping. [32, 33, 34, 35] Oftentimes methods learn a VAE on these generated poses to learn a function that maps from object to grasp pose. [30, 36] Recent works leverage differentiable simulation to synthesize stable grasp poses [37]. Other works don't decouple this problem into a grasp synthesis phase and learn it end-to-end in simulation [38] or use demonstrations [39, 40, 26] or teleoperate the robot [41, 42].

Functional Grasping: While simulation can be a powerful tool to optimize grasp metrics, functional affordances are usually human data since there may be more than one physically valid grasp pose but only one functionally valid one that allows one to use the object properly. Some approaches rely on clean annotations or motion capture datasets [43, 44, 45, 46] for hand object contact [47, 48, 49, 50, 51]. Some papers learn affordances from human images or video [52, 53] directly or through retargeting. These can however be noisy since they rely on hand pose detectors such as [54, 55] which are often noisy and difficult to learn from directly [39]. Some recent work in this area begin to target functional grasping using large scale datasets as a prior [56, 57, 58].

6 Limitations and Conclusion

We show that combining semantic information from models trained on internet data with the robustness of low-level control trained in simulation can yield functional grasps for a large range of objects. We show that using eigengrasps to restrict the action space of RL leads to policies that transfer better and are physically realistic. This leads to policies that are better to deploy in the real world on robot hand hardware.

We find that the main failure case of our policy in the real world is due to the fact that the pre-grasp from the affordance model are not always correct. In particular, if the pre-grasp is such that the knuckle of the thumb joint lies over the object then the grasp fails since the hand cannot get the thumb around the object. One way to address this limitation is to equip the robot with local field of view around the wrist such that it can finetune its grasp even if the affordance model is incorrect.

Another limitation is that our method currently does not leverage hand joint pose information from the affordance model for the pre-grasp. While we found this to not be necessary in the set of objects we have, it might be useful in the case of more fine-grained manipulation such as picking up very thin objects like coins or credits cards.

References

[1] K. Libertus, A. S. Joh, and A. W. Needham. Motor training at 3 months affects object exploration 12 months later. *Developmental Science*, 19(6):1058–1066, 2016.

[2] E. J. Gibson. Exploratory behavior in the development of perceiving, acting, and the acquiring of knowledge. *Annual review of psychology*, 39(1):1–42, 1988.

[3] K. E. Adolph and S. E. Berger. *Motor Development*, chapter 4. John Wiley & Sons, Ltd, 2007. ISBN 9780470147658. doi:https://doi.org/10.1002/9780470147658.chpsy0204. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470147658.chpsy0204.

[4] T. Bruce. *Learning through play, for babies, toddlers and young children*. Hachette UK, 2012.

[5] R. A. Cortes, A. E. Green, R. F. Barr, and R. M. Ryan. Fine motor skills during early childhood predict visuospatial deductive reasoning in adolescence. *Developmental Psychology*, 2022.

[6] O. M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020.

[7] T. Chen, M. Tippur, S. Wu, V. Kumar, E. Adelson, and P. Agrawal. Visual dexterity: In-hand dexterous manipulation from depth. *arXiv preprint arXiv:2211.11744*, 2022.

[8] A. Handa, A. Allshire, V. Makoviychuk, A. Petrenko, R. Singh, J. Liu, D. Makoviichuk, K. Van Wyk, A. Zhurkevich, B. Sundaralingam, et al. Dextreme: Transfer of agile in-hand manipulation from simulation to reality. *arXiv preprint arXiv:2210.13702*, 2022.

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

[10] A. Nair, A. Gupta, M. Dalal, and S. Levine. Awac: Accelerating online reinforcement learning with offline datasets. *arXiv preprint arXiv:2006.09359*, 2020.

[11] A. Nagabandi, K. Konolige, S. Levine, and V. Kumar. Deep dynamics models for learning dexterous manipulation. In *Conference on Robot Learning*, pages 1101–1112. PMLR, 2020.

[12] A. Agarwal, A. Kumar, J. Malik, and D. Pathak. Legged locomotion in challenging terrains using egocentric vision. *CoRL*, 2022.

[13] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, and M. Hutter. Learning robust perceptive locomotion for quadrupedal robots in the wild. *Science Robotics*, 7(62):eabk2822, 2022.

[14] G. B. Margolis and P. Agrawal. Walk these ways: Tuning robot control for generalization with multiplicity of behavior. In K. Liu, D. Kulic, and J. Ichnowski, editors, *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pages 22–31. PMLR, 14–18 Dec 2023. URL https://proceedings.mlr.press/v205/margolis23a.html.

[15] D. Hadjivelichkov, S. Zwane, M. P. Deisenroth, L. de Agapito, and D. Kanoulas. One-shot transfer of affordance regions? affcorrs! In *Conference on Robot Learning*, 2022.

[16] X. Zhou, R. Girdhar, A. Joulin, P. Krähenbühl, and I. Misra. Detecting twenty-thousand classes using image-level supervision. In *ECCV*, 2022.

[17] A. Kumar, A. Zhou, G. Tucker, and S. Levine. Conservative q-learning for offline reinforcement learning, 2020.

[18] 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 preprint arXiv:1709.10087*, 2017.

[19] M. Ciocarlie, C. Goldfeder, and P. Allen. Dexterous grasping via eigengrasps: A low-dimensional approach to a high-complexity problem. In *Robotics: Science and systems manipulation workshop-sensing and adapting to the real world*, 2007.

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

[21] N. Kamakura, M. Matsuo, H. Ishii, F. Mitsuboshi, and Y. Miura. Patterns of static prehension in normal hands. *The American journal of occupational therapy*, 34(7):437–445, 1980.

[22] C. L. MacKenzie and T. Iberall. *The grasping hand*. Elsevier, 1994.

[23] H. Qi, A. Kumar, R. Calandra, Y. Ma, and J. Malik. In-Hand Object Rotation via Rapid Motor Adaptation. In *Conference on Robot Learning (CoRL)*, 2022.

[24] I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

[25] Y. Qin, B. Huang, Z.-H. Yin, H. Su, and X. Wang. Dexpoint: Generalizable point cloud reinforcement learning for sim-to-real dexterous manipulation. In *Conference on Robot Learning*, 2022.

[26] S. P. Arunachalam, S. Silwal, B. Evans, and L. Pinto. Dexterous imitation made easy: A learning-based framework for efficient dexterous manipulation. *arXiv preprint arXiv:2203.13251*, 2022.

[27] A. Miller and P. Allen. Graspit! a versatile simulator for robotic grasping. *IEEE Robotics & Automation Magazine*, 11(4):110–122, 2004. doi:10.1109/MRA.2004.1371616.

[28] D. Berenson and S. S. Srinivasa. Grasp synthesis in cluttered environments for dexterous hands. In *Humanoids 2008-8th IEEE-RAS International Conference on Humanoid Robots*, pages 189–196. IEEE, 2008.

[29] R. Wang, J. Zhang, J. Chen, Y. Xu, P. Li, T. Liu, and H. Wang. Dexgraspnet: A large-scale robotic dexterous grasp dataset for general objects based on simulation. *arXiv preprint arXiv:2210.02697*, 2022.

[30] P. Li, T. Liu, Y. Li, Y. Geng, Y. Zhu, Y. Yang, and S. Huang. Gendexgrasp: Generalizable dexterous grasping. *arXiv preprint arXiv:2210.00722*, 2022.

[31] K. M. Lynch and F. C. Park. *Modern robotics*. Cambridge University Press, 2017.

[32] P. Grady, C. Tang, C. D. Twigg, M. Vo, S. Brahmbhatt, and C. C. Kemp. ContactOpt: Optimizing contact to improve grasps. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021.

[33] P. Mandikal and K. Grauman. Learning dexterous grasping with object-centric visual affordances. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6169–6176, 2021. doi:10.1109/ICRA48506.2021.9561802.

[34] S. Brahmbhatt, C. Ham, C. C. Kemp, and J. Hays. Contactdb: Analyzing and predicting grasp contact via thermal imaging. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[35] S. Brahmbhatt, A. Handa, J. Hays, and D. Fox. Contactgrasp: Functional multi-finger grasp synthesis from contact. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2386–2393, 2019. doi:10.1109/IROS40897.2019.8967960.

[36] Y. Xu, W. Wan, J. Zhang, H. Liu, Z. Shan, H. Shen, R. Wang, H. Geng, Y. Weng, J. Chen, T. Liu, L. Yi, and H. Wang. Unidexgrasp: Universal robotic dexterous grasping via learning diverse proposal generation and goal-conditioned policy. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4737–4746, June 2023.

[37] D. Turpin, L. Wang, E. Heiden, Y.-C. Chen, M. Macklin, S. Tsogkas, S. Dickinson, and A. Garg. Grasp'd: Differentiable contact-rich grasp synthesis for multi-fingered hands. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part VI*, pages 201–221. Springer, 2022.

[38] Y. Qin, B. Huang, Z.-H. Yin, H. Su, and X. Wang. Generalizable point cloud reinforcement learning for sim-to-real dexterous manipulation. In *Deep Reinforcement Learning Workshop NeurIPS 2022*.

[39] K. Shaw, S. Bahl, and D. Pathak. VideoDex: Learning Dexterity from Internet Videos. In *Conference on Robot Learning (CoRL)*, 2022.

[40] 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 *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXIX*, pages 570–587. Springer, 2022.

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

[42] 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. IEEE, 2020.

[43] Z. Fan, O. Taheri, D. Tzionas, M. Kocabas, M. Kaufmann, M. J. Black, and O. Hilliges. ARCTIC: A dataset for dexterous bimanual hand-object manipulation. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[44] R. Goyal, S. Ebrahimi Kahou, V. Michalski, J. Materzynska, S. Westphal, H. Kim, V. Haenel, I. Fruend, P. Yianilos, M. Mueller-Freitag, et al. The" something something" video database for learning and evaluating visual common sense. In *Proceedings of the IEEE international conference on computer vision*, pages 5842–5850, 2017.

[45] 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.

[46] O. Taheri, N. Ghorbani, M. J. Black, and D. Tzionas. GRAB: A dataset of whole-body human grasping of objects. In *European Conference on Computer Vision (ECCV)*, 2020. URL https://grab.is.tue.mpg.de.

[47] S. Brahmbhatt, C. Ham, C. C. Kemp, and J. Hays. Contactdb: Analyzing and predicting grasp contact via thermal imaging. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8709–8719, 2019.

[48] Y. Liu, Y. Liu, C. Jiang, K. Lyu, W. Wan, H. Shen, B. Liang, Z. Fu, H. Wang, and L. Yi. Hoi4d: A 4d egocentric dataset for category-level human-object interaction, 2022.

[49] O. Taheri, N. Ghorbani, M. J. Black, and D. Tzionas. Grab: A dataset of whole-body human grasping of objects. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part IV 16*, pages 581–600. Springer, 2020.

[50] S. Dasari, A. Gupta, and V. Kumar. Learning dexterous manipulation from exemplar object trajectories and pre-grasps. In *IEEE International Conference on Robotics and Automation 2023*, 2023.

[51] A. Patel, A. Wang, I. Radosavovic, and J. Malik. Learning to imitate object interactions from internet videos, 2022.

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

[53] Y. Ye, X. Li, A. Gupta, S. D. Mello, S. Birchfield, J. Song, S. Tulsiani, and S. Liu. Affordance diffusion: Synthesizing hand-object interactions. In *CVPR*, 2023.

[54] 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*, pages 1749–1759, 2021.

[55] A. Mittal, A. Zisserman, and P. H. Torr. Hand detection using multiple proposals. In *Bmvc*, volume 2, page 5, 2011.

[56] 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. *arXiv preprint arXiv:2210.13638*, 2022.

[57] J. Ye, J. Wang, B. Huang, Y. Qin, and X. Wang. Learning continuous grasping function with a dexterous hand from human demonstrations. *IEEE Robotics and Automation Letters*, 8(5): 2882–2889, 2023.

[58] S. Brahmbhatt, A. Handa, J. Hays, and D. Fox. Contactgrasp: Functional multi-finger grasp synthesis from contact. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2386–2393. IEEE, 2019.