Learning Dexterous Manipulation from Exemplar Object Trajectories and Pre-Grasps

Anonymous Author(s)
Affiliation
Address
email

Abstract

Learning diverse dexterous manipulation behaviors with assorted objects remains an open grand challenge. While policy learning methods offer a powerful avenue to attack this problem, they require extensive per-task engineering and algorithmic tuning. This paper seeks to escape these constraints, by developing a **P**re-**G**rasp informed **D**exterous **M**anipulation (**PGDM**) framework that generates diverse dexterous manipulation behaviors, without any task-specific reasoning or hyper-parameter tuning. At the core of PGDM is a well known robotics construct, *pre-grasps* (i.e. the hand-pose preparing for object interaction). This simple primitive is enough to induce efficient exploration strategies for acquiring complex dexterous manipulation behaviors. To exhaustively verify these claims, we introduce **TCDM**, a benchmark of 50 diverse manipulation tasks defined over multiple objects and dexterous manipulators. Tasks for TCDM are defined automatically using *exemplar object trajectories* from various sources (animators, human behaviors, etc.), without any per-task engineering and/or supervision. Our experiments validate that PGDM's exploration strategy, induced by a surprisingly simple ingredient (single pre-grasp pose), matches the performance of prior methods, which require expensive per-task feature/reward engineering, expert supervision, and hyper-parameter tuning. For animated visualizations, trained policies, and project code, please refer to https://sites.google.com/view/pregrasp/.

1 Introduction

Dexterous manipulation tasks – loosely defined as controlling a robot hand to effectively re-arrange its own environment [34] – were often solved by designing controllers to realize a sequence of stable object transitions. These approaches were limited to narrowly-scoped scenarios, as they required experts to carefully reason about hand-object contact behavior on a per-task basis: e.g. in terms of geometry [36, 37, 46], and/or force closures [15, 6, 30, 48]. As a result, the field has trended towards robot learning paradigms [27, 4, 17], which seek to leverage data-driven exploration to *automatically* acquire dexterous behaviors [25, 24, 39]. However, learning algorithms rely on naive search strategies [33] that struggle with dexterous manipulation, due to high-dimensional search spaces (e.g. ShadowHand [23] has 30-DoF) and small error-margins. In practice, this "deep exploration challenge" is addressed using a wealth of task-specific supervision and engineering. While effective, this approach takes us back to square one (heavy reliance on expert knowledge).

This paper seeks to find a general strategy that can overcome the aforementioned exploration challenge with minimal assumptions. Instead of viewing this issue from a purely algorithmic perspective, we demonstrate that "pre-grasp" states – i.e. a classical robotics construct [18] that denotes states where the robot is poised to initiate object interaction – can act as a general supervisory structure to relax exploration challenges in learning dexterous manipulation behaviors. Specifically, pre-grasps act as a

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

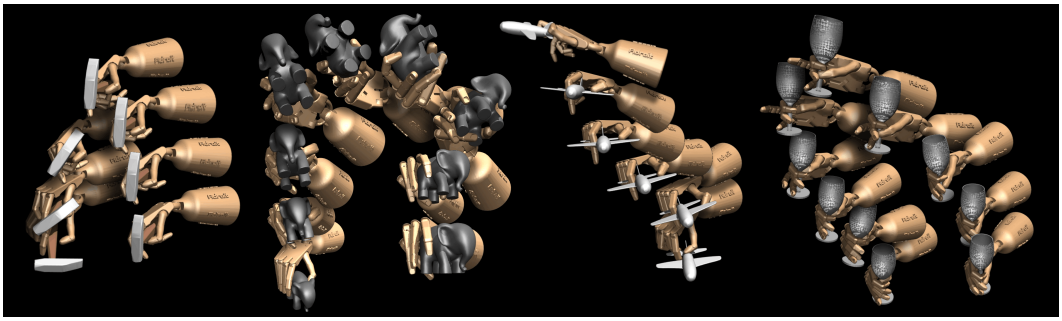


Figure 1: Dexterous behaviors learned by PGDM. Our experiments span diverse objects and tasks.

"pre-condition" that can enable robots to efficiently and safely explore object contacts and intermittent interaction dynamics, without requiring exquisite optimization techniques. In addition, pre-grasps are practical: they are easy to specify (e.g. via human annotation), realize (movement in free space), and unlike grasps (see A.2 for details), do not involve hard to sense object surface or inertial details. Thus, we propose a **P**re-**G**rasp informed **D**exterous **M**anipulation (**PGDM**) framework that embeds pre-grasp poses (as exploration primitives) into existing learning pipelines, to synthesize behaviors without requiring task engineering or hyper-parameter tuning. While the connection between pre-grasps and manipulation has been studied before [19, 20, 5], to the best of our knowledge, we are the first to analyze pre-grasp's effectiveness in learning based paradigms.

To demonstrate PGDM's versatility, it is important to test across as many scenarios as possible. However, existing dexterous manipulation benchmarks are shallow and packed with expert knowledge (e.g. favourable tasks with heavily engineered details – initialization, input, reward, etc.). Thus, we developed a **T**rajectory **C**onditioned **D**exterous **M**anipulation benchmark, **TCDM**. As the name suggests, TCDM tasks are automatically constructed from diverse *exemplar object trajectories* (sourced from human behaviors, animations, etc.), which prevents expert designers from injecting task-specific supervision. Indeed, every part of the task setup (e.g. formulation, reward/termination functions, hyper-parameters, etc.) is kept constant across TCDM, except for the exemplar trajectory itself. TCDM's diverse tasks span: 3 robotic hands, 30+ standardized objects [7, 9], and behaviors ranging from fixed goal-reaching (e.g. relocation [45, 43]) to cyclic, dynamic skills (e.g. hammering, bottle shaking, etc.).

This work develops PGDM, a simple exploration framework for dexterous manipulation, and validates it across the diverse tasks in TCDM. Our contributions include: **(1)** identifying pre-grasps as a key ingredient to guide successful exploration in dexterous manipulation, and embedding them into existing behavior synthesis pipelines. **(2)** Next, our PGDM framework achieves SOTA results on a diverse suite of dexterous manipulation tasks, while using significantly less supervision (e.g. single frame vs full hand trajectory) than representative baseline methods. **(3)** In addition, we find which pre-grasp properties (e.g. proximity, finger pose) are important for successful behavior learning. **(4)** Finally, we commit to open-sourcing the TCDM benchmark, PGDM's code-base, and all experimental artifacts (e.g. trained policies) for the community's benefit.

2 Related Work

Prior robot learning approaches achieved impressive results on various dexterous tasks [11, 16, 13, 45, 43, 32, 39, 2, 3]. But while learning approaches strive to be automatic, prior work requires a wealth of expertise for successful deployment. We now classify these approaches (and others), by the supervision strategies required to make them work in practice.

Task-engineering A popular solution is to carefully design environments, tasks, and learning curriculums that structure the robot's exploration. This can be accomplished by: extensive reward shaping [39, 3] to reduce noise in optimization; action space constraints [45] to prevent degenerate solutions; decomposing tasks into sub-skills [22, 42]; changing environment physics so the policy can develop its skill over the course of training [11, 3]; and cleverly initializing the policy so it can learn to pass through challenging bottlenecks in the state space [41, 12]. These strategies require *weeks* of expert trial and error to make a single-task solution, and often rely on unrealistic assumptions

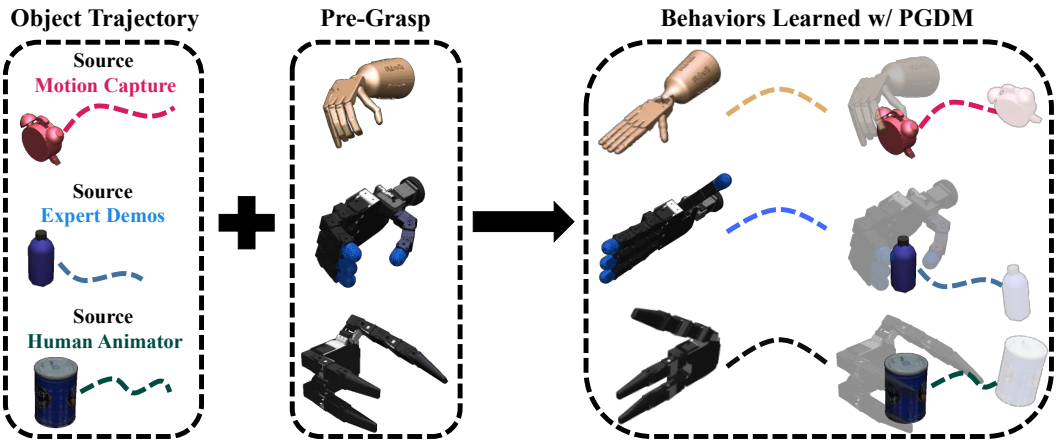


Figure 2: Our method uses simple ingredients to learn dexterous behaviors in diverse scenarios. The PGDM framework leverages *pre-grasp* states as general exploration primitives, to solve tasks automatically defined using *object trajectories*, without any per-task tuning. Both of these ingredients can be easily mined (e.g. from MoCap data) or hand-annotated (e.g. by human animators).

(e.g. changing gravity, reset robot in mid-air, etc.). In contrast, PGDM avoids these issues by using a simple pre-grasp based exploration primitive to accelerate learning, and needs no task knowledge.

Expert Data Another common strategy is to initialize exploration strategies with expert data – ranging from trajectories collected by human demonstrators [45, 14, 43] to affordances [32] mined from human contact data [7, 52]. However, this data rarely generalizes between settings (i.e. trajectories are robot and task specific), and collecting it is expensive, since it requires special purpose experimental setups [14, 7]. More fundamentally, it is unclear if/when adding additional data can yield performance benefits. Our investigation addresses these issues, by outlining how simple data sources (pre-grasps and object trajectories) can accelerate policy learning, while being easy to acquire [47, 56, 10, 29].

3 Methods

How can a single method learn a diverse range of dexterous manipulation behaviors? We argue that a simple, data-driven solution with minimal hyper-parameters offers the best chance. In this spirit, Sec. 3.1 presents a general task formulation, which parameterizes diverse dexterous behaviors using *exemplar object trajectories*, and Sec. 3.2 introduces PGDM, a framework for accelerating policy learning using pre-grasp states. An overview of our approach is shown in Fig. 2.

3.1 Task Formulation

Let's begin by formalizing the definitions for robotic tasks and environments. We adopt the finite Markov Decision Process (MDP) formulation [51]. At each time-step the agent observes states ($s_t \in \mathcal{S}$) and goals ($g_t \in \mathcal{S}$), and executes an action ($a_t \in \mathcal{A}$). The next state evolves according to stochastic dynamics ($s_{t+1} \sim P(\cdot|s_t, a_t)$). The agent collects trajectory rollouts within the MDP ($\tau = [s_0, a_0, s_1, \ldots, s_n]$), starting from an initial state $s_0 \sim P(s_0)$. Desired behavior is specified by adding *time-varying* goal variables ($G = [g_1, \ldots, g_T]$) that are used to condition both the reward function $R(s_t, a_t, g_t)$ (optimized by agent) and the termination condition $T(s_t, g_t)$ (early-stops failed episodes). To preserve the Markov property, the current time-step t must be appended to s_t, since $g_t = G(t)$. Note that this is a super-set of the more standard static-goal conditioned MDP: it allows us to specify time-varying behaviors, and we can recover the standard formulation by setting $g_t = g \; \forall \; t$. Given a discount factor γ, the learning objective is to find a policy $a_t \sim \pi(\cdot|s_t)$ that maximizes: $\textbf{max}_\pi \;\; J(\pi) = \;\; \mathbb{E}_{\tau\sim\pi}[\Sigma_{t=0}^{\infty} \gamma^t R(s_t, a_t, g_t)]$.

Parameterizing Task MDPs We now describe how to create task MDPs from exemplar object trajectories – i.e. $X = [x_1, \ldots, x_T]$, where each $x_i = [x_i^{(p)}, x_i^{(o)}]$ is an object pose (position and orientation). Object trajectories are used as goal variables (i.e. $G = X$), which in turn parameterize a

pre-defined reward function R and termination condition T. Specifically: **(1)** goal variables are set to match the desired object pose at each time-step $g_t = x_t$; **(2)** the reward function encourages matching the exemplar trajectory – $R(\hat{x}_t, x_t) := \lambda_1 exp\{-\alpha||x_t^{(p)} - \hat{x}_t^{(p)}||_2 - \beta|\angle(x_t^{(o)}, \hat{x}_t^{(o)})|\} + \lambda_2 \mathbb{1}\{lifted\}$, where $\hat{x}_t = \hat{x}(s_t)$ is the real object pose in state s_t, $\angle$ is the Quaternion angle between the two orientations, and $\mathbb{1}\{lifted\} = x_t^{(z)} > \zeta$ **and** $\hat{x}_t^{(z)} > \zeta$ encourages stable object lifting; **(3)** episodes are terminated when the object is too far from the goal $T(x_t, \hat{x}_t) := ||x_t - \hat{x}_t||_2 > \omega$. All hyper-parameters for these function are reported in App. B.1. This formulation encourages the robot to produce behaviors that match the given template object trajectory. In practice, it allows us to specify diverse tasks – including dynamic, cyclic behaviors that eluded past work (e.g. hammering) – simply by supplying an appropriate object trajectory. One can even recover standard static-goal conditioned behaviors (e.g. lifting), by setting a fixed goal pose for the whole trajectory. Note that all this is possible without any per-task engineering (e.g. knob-turning bonus [45], etc.).

3.2 PGDM: Accelerating Exploration w/ Pre-Grasps

Our attention turns to creating a general exploration primitive that can accelerate learning across a wide range of tasks. Note that all dexterous tasks begin with the hand gaining proximity to the target object, before transitioning into general manipulation. Thus, it's natural to decompose dexterous tasks into a "reaching stage" and a "manipulation stage," and use different strategies to solve each. But in the first stage, what state should the robot reach for? We argue that pre-grasp states (i.e. hand pose directly preceding contact) provide the answer. Pre-grasps favourably position the robot relative to a target object, so that it can quickly learn the intermittent contacts behaviors required for dexterous manipulation. For example, the pre-grasps shown in Fig. 2 position the robot hand near the target object and *around* functional parts (e.g. fingers wrapped around handle), which allows the robot to easily gain control. As an added bonus pre-grasps require minimal assumptions: they can be cheaply annotated by human labellers or mined from human behavior data [56, 10], and can be easily reached by robots (e.g. w/ free-space planner [21]). The key insight of PGDM is to exploit these favorable properties, by moving robot hands to the pre-grasp state before beginning the learning process. From the learning agent's (i.e. π) perspective, this is equivalent to modifying P_0 to maximally reduce exploration complexity, while still making minimal assumptions in practice. For additional pre-grasp examples and a more extensive definition, please refer to App. A.

4 Experimental Setup

The following sections describe how our task formulation is used to create TCDM (see Sec. 4.1), alongside our implementation of PGDM (see Sec. 4.2). We stress that these decisions were made for the sake of consistent experiments, and are not inherent to our framework.

4.1 Introducing TCDM

Our task formulation (see Sec. 3.1) acts as a recipe for converting exemplar object trajectories into dexterous manipulation tasks. We use this to define a set of 50 tasks, which span: 34 different objects; 3 distinct robotic hand platforms; and unique object trajectories mined from three sources – motion capture data-sets, human animated trajectories, and expert policy behaviors (see Figs. 1, 2). The pre-grasps for each task come from one of four sources: (1) human MoCap recordings [52] transferred to robot via IK, (2) expert pre-grasps extracted from Tele-Op data [45], (3) manually labeled pre-grasps, and (4) learned pre-grasps generated by an object mesh conditioned grasp predictor [52]. Further details on the task creation process and a full table of all tasks are presented in App. B.2.

To make an investigation of this scale reproducible, the tasks are simulated (using MuJoCo [53]) and compiled into a benchmark, named TCDM-50. Note that a subset of 30 tasks (named TCDM-30) contain additional supervision, in the form of expert hand trajectories and grasping data. While not useful for our formulation, this data is required for the baselines. Thus, some of our experiments are run on the abridged TCDM-30 for fair comparison. More details are provided in App. B.3.

Success Metrics: Before continuing, let's discuss quantitative metrics for judging performance on TCDM tasks. Put simply, a "good" policy is one that stably controls the object and matches the exemplar trajectory. We define (using the constants from Sec. 3.1) three simple metrics that

capture both these properties. The *COM Error* metric – $E(\hat{X}) = \frac{1}{T}\Sigma_{t=0}^{T}||x_t^{(p)} - \hat{x}_t^{(p)}||_2$ – calculates Euclidean error (in meters) between the object's COM position, and the desired position from the exemplar trajectory. Similarly, the *Ori Error* metric – $E(\hat{X}) = \frac{1}{T}\Sigma_{t=0}^{T}\angle(x_t^{(o)} - \hat{x}_t^{(o)})$ – is defined as the angle (in radians) between the achieved object orientation and desired orientation. In addition, the *success* metric – $S(\hat{X}) = \frac{1}{T}\Sigma_{t=0}^{T}\mathbb{1}\{||x_t^{(p)} - \hat{x}_t^{(p)}||_2 < \epsilon\}$ reports the fraction of time-steps where COM error is below a $\epsilon = 1$cm threshold. In practice, we've noticed that that humans easily perceive roll-outs that score from $60 - 80\%$ as "successful."

4.2 Implementing PGDM

Finally, let's discuss our implementation for the two stage task decomposition proposed by PGDM (see Sec. 3.2). Given a task's initial state distribution $P(s_0)$ (e.g. hand at reset position and object on table), we load an appropriate pre-grasp state s_{pg}, and solve for a policy (using a scene-agnostic trajectory optimizer [31]) that moves the robot to s_{pg}. At this point, our system transitions to learning an agent within the MDP (i.e. maximize J) as normal. Specifically, we utilize the PPO [50] algorithm, to learn the dexterous behavior. Note that (except for pre-grasp) the entire system remains fixed across different tasks. All relevant hyper-parameters and pseudo-code are presented in App. B.4.

5 Experiments

These experiments seek to validate both our trajectory centric task formulation (i.e. TCDM) and our pre-grasp based exploration primitive (i.e. PGDM). Specifically, we pose the following questions: **(Q1)** Can our methodology learn a broad and diverse range of dexterous manipulation skills? **(Q2)** Are we able to match the performance of baselines methods that leverage task specific reasoning (demonstrations, curriculum, etc.)? **(Q3)** What attributes of pre-grasps make them useful exploration primitive? **(Q4)** And finally, how accurately do our simulated results match real world behavior?

5.1 Learning Behaviors w/ Pre-Grasps and PGDM Tasks

To verify our methods' viability, we deploy the PGDM policy learning scheme on the *entire* TCDM benchmark. Recall, no task specific tuning is allowed for *any* component in our setup. Since RL algorithms often display significant run-to-run variance, we run this experiment using 3 random seeds. Error and success metrics at the end of training (broken down by pre-grasp source and averaged across tasks) are shown in Table 1. The behavior policies learned with PGDM achieve a tracking error of 5.23e-3, success rate of 74.5%, and low run-to-run variance, despite the breadth and diversity of TCDM tasks. Note that PGDM can learn effective policies using any of our 4 pre-grasp sources (MoCap, Expert Tele-Op, Human Labeled, and Learned). In particular, the successful experiments w/ learned pre-grasps suggest further avenues for scaling our results. Even though PGDM uses no hand supervision outside of pre-grasps, we note that the final policies often produce smooth motions and realistic finger behavior. This defies conventional wisdom in the field that suggests human supervision is critical for "normal" behaviors [45, 32, 43]. That being said, multiple imperfections (e.g. large forces) remain in our policies, which leaves room for further improvement. Readers are encouraged to view the supplementary video[1], to understand the learned qualitative behaviors. For additional visualizations, learning curves, and a more thorough breakdown of individual tasks please refer to App. C.

	All Trials	MoCap	Tele-Op	Labeled	Learned
Success	74.5%	75.0%	84.5%	69.2%	90.4%
COM Error (m)	5.23e-3	4.45e-3	1.12e-3	7.76e-3	1.55e-3
Ori Error (rad)	0.33	0.32	0.059	0.42	0.18
# of Tasks	50	37	3	7	3

Table 1: Error and success metrics (averaged across all tasks w/ 3 seeds per-task) at the end of RL training (50M samples), and broken down by pre-grasp source. Note how PGDM achieves high performance using diverse pre-grasp sources.

5.2 Baselines: Is Additional Supervision Needed for Exploration?

Our prior experiment demonstrated that PGDM can solve a wide range of manipulation tasks. We now seek to understand if PGDM can compete with baselines, which rely on significantly more expert

[1] https://sites.google.com/view/pregrasp/

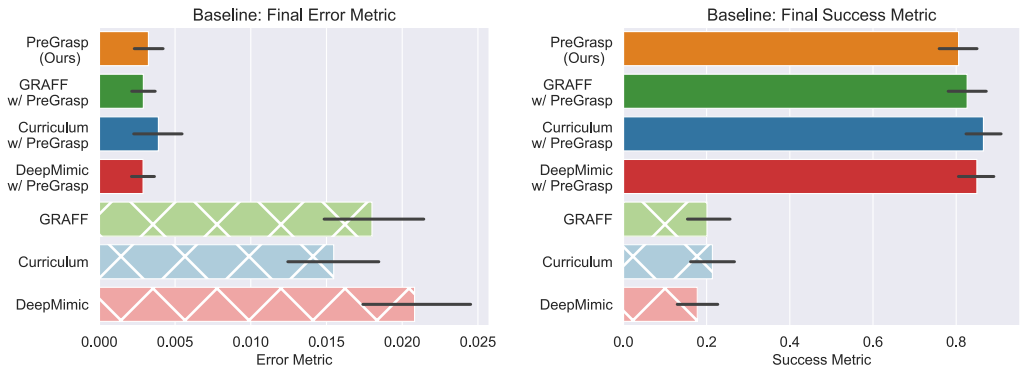


Figure 3: Average Success and Error metrics at the end of training for both the PGDM-only method and 6 baselines (3 methods, w/ and w/out PGDM). Note how methods using PGDM strongly outperform those that don't, and how adding additional supervision does not improve performance.

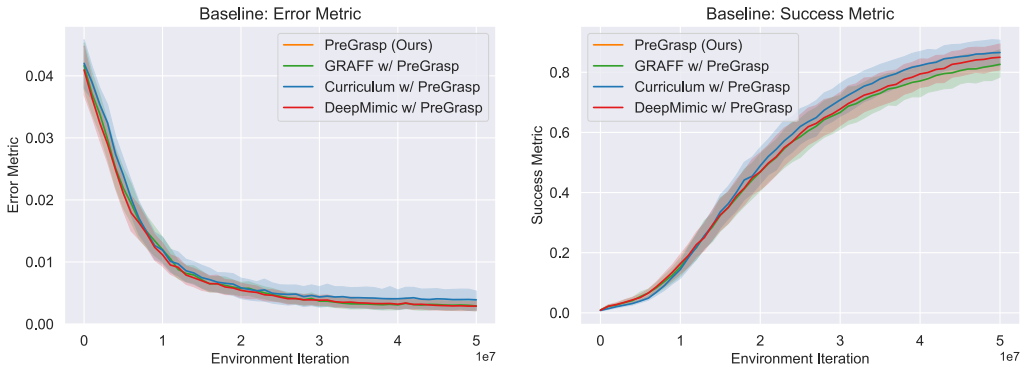
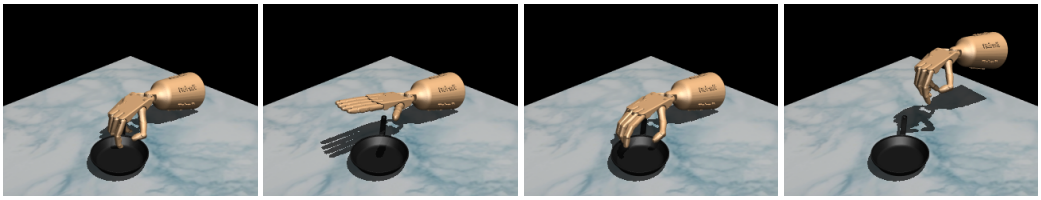


Figure 4: Learning curves (of Success/Error metric) comparing PGDM-only against baselines initialized w/ PGDM. Note how adding supervision to pre-grasps does not improve performance at any stage of learning.

supervision and tuning for stable exploration. Specifically, we consider three baselines (listed below) that broadly exemplify prior work in this area:

- **DeepMimic** [41]: DeepMimic requires full hand and object trajectory supervision: it optimizes the robot to imitate expert fingertip poses in addition to the object trajectory. Thus, this baseline receives the maximum possible expert supervision at every time-step. DeepMimic is easily implemented by adding rewards and termination conditions for the fingertips to our existing task formulation.
- **GRAFF*** [32]: GRAFF encourages the robot to make functional contacts with the objects using "object affordances" – i.e. parts of the object where a human expert would grasp to accomplish a task. In practice, it rewards the robot for making contact at ground truth grasping points. While the original paper operated on visual observations, we re-implement it with simulator state information for fair comparison.
- **Task Curriculum** [22]: This baseline uses an expert designed curriculum to accelerate policy learning, in the hope that learning easy tasks will accelerate learning harder tasks later on. First, the robot must learn how to stably pick (i.e. lift) objects. To learn the rest of the task, our full tracking objective (e.g. λ_1 from Sec. 3.1) is linearly activated over the course of 4M timesteps (i.e. average time to learn lifting).

A major benefit of using PGDM is that it shortens the task exploration horizon, since it positions the robot near the object. To control for this factor, we implement each baseline twice – with and without PGDM. Additional implementation details are presented in App. D. All six baselines (3



(a) Pre-Grasp (b) Open Hand (c) Mean Hand (d) Distance Ablation

Figure 5: Our ablations are implemented by sub-optimally adjusting the pre-grasp pose. We show visualizations of this process with the "Frying Pan" object.

	Pre-Grasps	**Ablate Pose**		**Ablate Distance**	
		Open	Mean	15 cm	25 cm
Success Metric	**75.5%**	23.1%	59.5%	28.6%	19.7%
Error Metric	**4.82e-3**	1.55e-2	5.90e-3	1.17e-2	1.92e-2

Table 2: PGDM is run on TCDM-30, using default and ablated pre-grasps. Error and success metrics at the end of training, are presented above. Destroying key pre-grasp attributes (i.e. pose and proximity) significantly harms performance. The best ablation (Mean Hand), represents the least deviation from the default pre-grasp.

methods, w/ and w/out PGDM) are evaluated against a PGDM-only method on TCDM-30[2] tasks. Their performance at 50M steps are presented in Fig. 3. We observe that the baseline methods (which require dense supervision beyond pre-grasps) provide no appreciable performance boost (even when using PGDM), and are completely ineffective w/out PGDM. This is true even during the early stages of training: the PGDM-only method remains competitive against all baseline w/ PGDM implementations at every optimization step (see Fig. 4).

These experiments offer strong evidence that pre-grasps act as crucial supervision for dexterous manipulation, since the baselines could only remain competitive when implemented w/ PGDM. Simply put, behavior synthesis frameworks that leverage pre-grasps can more easily acquire diverse dexterous manipulation behaviors, thus making further supervision far less valuable. Indeed, this observation is also reflected (though unacknowledged) in past learning work [11, 16, 3] – we find that removing pre-grasps from their setups causes them to collapse entirely (see App. A.3).

5.3 Ablations: What Makes a Pre-Grasp Useful?

Our investigation has established that pre-grasps are a key source of supervision that enable scaling to the diverse tasks in TCDM. We now run an ablation study to understand what pre-grasp properties (e.g. hand pose, proximity to object, etc.) make them useful during learning.

The following ablation classes are considered (visualized in Fig. 5):

- **Ablate Pose:** This ablation tests if finger pose information (i.e. finger joint positions) is required for PGDM to work. Specifically, we replace the pre-grasp finger pose with both an "Open Hand" (see Fig. 5b) and a "Mean Hand" (see Fig. 5c), calculated by averaging all the pre-grasps used in our investigation.
- **Ablate Distance:** The previous ablation does not address the importance of the object proximity. To test this factor, we shift the wrist away from the pre-grasp (towards default robot's reset pose) by two fixed offsets (see Fig. 5d), while keeping the finger pose fixed.

These ablations[3] (see Table. 2) reveal that object proximity matters significantly – moving the hand away from the object dramatically reduces PGDM's performance. Finger pose information is critical as well. The "Open Hand" experiment demonstrates how removing the pre-grasp's finger pose causes a drastic decrease in performance. Furthermore, the "Mean Hand" experiments show that some of the fine-grained aspects of the pre-grasp pose matter, since replacing it with a generic hand pose resulted in a 20% performance hit. However, the Mean Hand does perform significantly better than the Open

[2] TCDM-30 is used, since the baselines need added supervision.

[3] Also run on TCDM-30 for consistency w/ baselines.

Hand setting (59.5% vs 23.1% respectively), which indicates that pre-grasps need not be perfect for control. This error tolerance suggests that one could use predicted pre-grasp states in policy learning.

5.4 Real World Validation

Since PGDM does not use extensive supervision to shape learning, it is possible that the learnt policies will not be viable on real hardware (e.g. actions are too aggressive). Our final experiment seeks to dispel this fear, by executing actions from a trained (using PGDM) policy (in open-loop fashion) on an actual D'Manus robot. Specifically, a simple "Cracker-box Lifting" task is defined using PGDM, alongside a matching real-world environment replica (see Fig. 6, details in App. E). We find that simulated actions can be replayed on the robot – i.e. the robot grasps and lifts the cracker-box using the learned behavior. This provides initial evidence that our simulated results could fully transfer to hardware. However, a more thorough real world investigation of our ideas (i.e. robot policy deployment for all tasks) is outside the scope of this paper.

Cracker-Box Lifting Robotic Setup

Figure 6: Picture of our real world task setup. The D'Manus robotic hand is controlled to lift the "Cheez-Itz" cracker-box from YCB [9].

6 Discussion

This paper demonstrated that simple ingredients can enable learning dexterous behaviors in diverse scenarios. Specifically, we use exemplar object trajectories as generic task specifiers, and pre-grasps as supervisory signals for exploration. Our primary contributions are (1) the PGDM framework, which functions as a simple exploration prior for dexterous policy learning, and (2) the TCDM benchmark which fills an important gap – the lack of diverse dexterous manipulation benchmarks (50 tasks, 20+ objects, 3 robots). Our system was able to achieve diverse control results with *no* per-task expert engineering, while using the minimal possible supervision (i.e. a single pre-grasp frame). Indeed, our learned behaviors match the performance of baselines that make significantly more assumptions. Finally, we characterize the pre-grasp properties required for stable exploration, as well as demonstrate that our learned behaviors are physically plausible.

7 Limitations and Future Work

While our investigation was expansive, there are multiple vectors of improvement that should be addressed in future work. First, our investigation was primarily conducted in simulation, which is quite common in this space due to the lack of affordable dexterous hands (ShadowHand is $100K+). However, a few affordable solutions are in development [13, 2], which we are starting to investigate. We hope to eventually deploy a fully trained PGDM policy in the real world using a combination of: domain randomization [3]; real world training [2, 39]; and/or adaptation [35]. In addition, the pre-grasps used in this investigation were curated, but this approach would not work "in-the-wild" where objects are innumerable. Inspired by recent work in the vision community [56, 10, 29, 38], we plan to replace curated pre-grasps with pre-grasps predicted from visual inputs using minimal assumptions (e.g. unknown object mesh). Next, while our trajectory centric task formulation can encode a wide range of behaviors, extensions are needed to further increase task diversity. For example, additional constraints will be needed for tasks with precise force requirements (e.g. hammering a nail with 15N force). In addition, we only consider single-object tasks without distractor objects or clutter. Handling these situations will require changes to our task formulation, and a more flexible (i.e. clutter-aware [26]) reaching policy in PGDM. Finally, we hope to swap our policies from state to visual observations, in order to handle a wider range of (e.g. deformable) objects and enable inter-task generalization.

References

[1] V-HACD, howpublished = https://github.com/kmammou/v-hacd, note = Accessed: 2022-01-28, year=2012, author=Mamou, Khaled and Ratcliff, John W.

[2] Michael Ahn, Henry Zhu, Kristian Hartikainen, Hugo Ponte, Abhishek Gupta, Sergey Levine, and Vikash Kumar. Robel: Robotics benchmarks for learning with low-cost robots. In *Conference on Robot Learning*, pages 1300–1313. PMLR, 2020.

[3] Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, et al. Solving rubik's cube with a robot hand. *arXiv preprint arXiv:1910.07113*, 2019.

[4] Kai Arulkumaran, Marc Peter Deisenroth, Miles Brundage, and Anil Anthony Bharath. Deep reinforcement learning: A brief survey. *IEEE Signal Processing Magazine*, 34(6):26–38, 2017.

[5] Inhyuk Baek, Kyoosik Shin, Hyunjun Kim, Seunghoon Hwang, Eric Demeester, and Min-Sung Kang. Pre-grasp manipulation planning to secure space for power grasping. *Ieee Access*, 9:157715–157726, 2021.

[6] Antonio Bicchi. On the closure properties of robotic grasping. *The International Journal of Robotics Research*, 14(4):319–334, 1995.

[7] Samarth Brahmbhatt, Cusuh Ham, Charles C Kemp, and James 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.

[8] Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym, 2016.

[9] Berk Calli, Arjun Singh, Aaron Walsman, Siddhartha Srinivasa, Pieter Abbeel, and Aaron M Dollar. The ycb object and model set: Towards common benchmarks for manipulation research. In *2015 international conference on advanced robotics (ICAR)*, pages 510–517. IEEE, 2015.

[10] Zhe Cao, Ilija Radosavovic, Angjoo Kanazawa, and Jitendra Malik. Reconstructing hand-object interactions in the wild. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12417–12426, 2021.

[11] Tao Chen, Jie Xu, and Pulkit Agrawal. A system for general in-hand object re-orientation. In *Conference on Robot Learning*, pages 297–307. PMLR, 2022.

[12] Carlos Florensa, David Held, Markus Wulfmeier, Michael Zhang, and Pieter Abbeel. Reverse curriculum generation for reinforcement learning. In *Conference on robot learning*, pages 482–495. PMLR, 2017.

[13] Abhishek Gupta, Justin Yu, Tony Z Zhao, Vikash Kumar, Aaron Rovinsky, Kelvin Xu, Thomas Devlin, and Sergey Levine. Reset-free reinforcement learning via multi-task learning: Learning dexterous manipulation behaviors without human intervention. In *International Conference on Robotics and Automation (ICRA)*, 2021.

[14] Ankur Handa, Karl Van Wyk, Wei Yang, Jacky Liang, Yu-Wei Chao, Qian Wan, Stan Birchfield, Nathan Ratliff, and Dieter 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.

[15] W Stamps Howard and Vijay Kumar. On the stability of grasped objects. *IEEE transactions on robotics and automation*, 12(6):904–917, 1996.

[16] Wenlong Huang, Igor Mordatch, Pieter Abbeel, and Deepak Pathak. Generalization in dexterous manipulation via geometry-aware multi-task learning. *arXiv preprint arXiv:2111.03062*, 2021.

[17] Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, and Chrisina Jayne. Imitation learning: A survey of learning methods. *ACM Computing Surveys (CSUR)*, 50(2):1–35, 2017.

[18] Sing Bing Kang and Katsushi Ikeuchi. Determination of motion breakpoints in a task sequence from human hand motion. In *Proceedings of the 1994 IEEE International Conference on Robotics and Automation*, pages 551–556. IEEE, 1994.

[19] Daniel Kappler, Lillian Chang, Markus Przybylski, Nancy Pollard, Tamim Asfour, and Rüdiger Dillmann. Representation of pre-grasp strategies for object manipulation. In *2010 10th IEEE-RAS International Conference on Humanoid Robots*, pages 617–624. IEEE, 2010.

[20] Daniel Kappler, Lillian Y Chang, Nancy S Pollard, Tamim Asfour, and Rüdiger Dillmann. Templates for pre-grasp sliding interactions. *Robotics and Autonomous Systems*, 60(3):411–423, 2012.

[21] Sertac Karaman, Matthew R Walter, Alejandro Perez, Emilio Frazzoli, and Seth Teller. Anytime motion planning using the rrt. In *2011 IEEE International Conference on Robotics and Automation*, pages 1478–1483. IEEE, 2011.

[22] Andrej Karpathy and Michiel Van De Panne. Curriculum learning for motor skills. In *Canadian Conference on Artificial Intelligence*, pages 325–330. Springer, 2012.

[23] Vikash Kumar. *Manipulators and Manipulation in high dimensional spaces*. PhD thesis, University of Washington, Seattle, 2016.

[24] Vikash Kumar, Abhishek Gupta, Emanuel Todorov, and Sergey Levine. Learning dexterous manipulation policies from experience and imitation. *arXiv preprint arXiv:1611.05095*, 2016.

[25] Vikash Kumar, Emanuel Todorov, and Sergey Levine. Optimal control with learned local models: Application to dexterous manipulation. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 378–383. IEEE, 2016.

[26] Michael Laskey, Jonathan Lee, Caleb Chuck, David Gealy, Wesley Hsieh, Florian T Pokorny, Anca D Dragan, and Ken Goldberg. Robot grasping in clutter: Using a hierarchy of supervisors for learning from demonstrations. In *2016 IEEE international conference on automation science and engineering (CASE)*, pages 827–834. IEEE, 2016.

[27] Timothy P Lillicrap, Jonathan J Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *International Conference on Learning Representations*, 2016.

[28] Yixin Lin, Austin S. Wang, Giovanni Sutanto, Akshara Rai, and Franziska Meier. Polymetis. https://facebookresearch.github.io/fairo/polymetis/, 2021.

[29] Shaowei Liu, Hanwen Jiang, Jiarui Xu, Sifei Liu, and Xiaolong Wang. Semi-supervised 3d hand-object poses estimation with interactions in time. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14687–14697, 2021.

[30] Yun-Hui Liu, Miu-Ling Lam, and Dan Ding. A complete and efficient algorithm for searching 3-d form-closure grasps in the discrete domain. *IEEE Transactions on Robotics*, 20(5):805–816, 2004.

[31] Kendall Lowrey, Aravind Rajeswaran, Sham Kakade, Emanuel Todorov, and Igor Mordatch. Plan Online, Learn Offline: Efficient Learning and Exploration via Model-Based Control. In *ICLR*, 2019.

[32] Priyanka Mandikal and Kristen Grauman. Dexterous robotic grasping with object-centric visual affordances. In *International Conference on Robotics and Automation (ICRA)*, 2021.

[33] Horia Mania, Aurelia Guy, and Benjamin Recht. Simple random search of static linear policies is competitive for reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

[34] Matthew T Mason. Toward robotic manipulation. *Annual Review of Control, Robotics, and Autonomous Systems*, 1:1–28, 2018.

[35] Takahiro Miki, Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning robust perceptive locomotion for quadrupedal robots in the wild. *Science Robotics*, 7(62):eabk2822, 2022.

[36] Andrew T Miller and Peter K Allen. Graspit! a versatile simulator for robotic grasping. *IEEE Robotics & Automation Magazine*, 11(4):110–122, 2004.

[37] Andrew T Miller, Steffen Knoop, Henrik I Christensen, and Peter K Allen. Automatic grasp planning using shape primitives. In *2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422)*, volume 2, pages 1824–1829. IEEE, 2003.

[38] Adithyavairavan Murali, Weiyu Liu, Kenneth Marino, Sonia Chernova, and Abhinav Gupta. Same object, different grasps: Data and semantic knowledge for task-oriented grasping. In *Conference on Robot Learning*, 2020.

[39] Anusha Nagabandi, Kurt Konolige, Sergey Levine, and Vikash Kumar. Deep dynamics models for learning dexterous manipulation. In *Conference on Robot Learning*, pages 1101–1112. PMLR, 2020.

[40] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32:8026–8037, 2019.

[41] Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel van de Panne. Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions on Graphics (TOG)*, 37(4):1–14, 2018.

[42] Rémy Portelas, Cédric Colas, Lilian Weng, Katja Hofmann, and Pierre-Yves Oudeyer. Automatic curriculum learning for deep rl: A short survey. *arXiv preprint arXiv:2003.04664*, 2020.

[43] Yuzhe Qin, Yueh-Hua Wu, Shaowei Liu, Hanwen Jiang, Ruihan Yang, Yang Fu, and Xiaolong Wang. Dexmv: Imitation learning for dexterous manipulation from human videos. *arXiv preprint arXiv:2108.05877*, 2021.

[44] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 22(268):1–8, 2021.

[45] Aravind Rajeswaran, Vikash Kumar, Abhishek Gupta, Giulia Vezzani, John Schulman, Emanuel Todorov, and Sergey Levine. Learning Complex Dexterous Manipulation with Deep Reinforcement Learning and Demonstrations. In *Proceedings of Robotics: Science and Systems (RSS)*, 2018.

[46] Máximo A Roa and Raúl Suárez. Geometrical approach for grasp synthesis on discretized 3d objects. In *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3283–3288. IEEE, 2007.

[47] Yu Rong, Takaaki Shiratori, and Hanbyul Joo. Frankmocap: A monocular 3d whole-body pose estimation system via regression and integration. In *IEEE International Conference on Computer Vision Workshops*, 2021.

[48] J-P Saut, Constant Remond, Véronique Perdereau, and Michel Drouin. Online computation of grasping force in multi-fingered hands. In *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1223–1228. IEEE, 2005.

[49] John Schulman, Yan Duan, Jonathan Ho, Alex Lee, Ibrahim Awwal, Henry Bradlow, Jia Pan, Sachin Patil, Ken Goldberg, and Pieter Abbeel. Motion planning with sequential convex optimization and convex collision checking. *The International Journal of Robotics Research*, 33(9):1251–1270, 2014.

[50] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[51] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction*. MIT press, 2018.

[52] Omid Taheri, Nima Ghorbani, Michael J Black, and Dimitrios Tzionas. Grab: A dataset of whole-body human grasping of objects. In *European Conference on Computer Vision*, pages 581–600. Springer, 2020.

[53] Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033. IEEE, 2012.

[54] Saran Tunyasuvunakool, Alistair Muldal, Yotam Doron, Siqi Liu, Steven Bohez, Josh Merel, Tom Erez, Timothy Lillicrap, Nicolas Heess, and Yuval Tassa. dm_control: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020.

[55] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

[56] Yufei Ye, Shubham Tulsiani, and Abhinav Gupta. Shelf-supervised mesh prediction in the wild. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8843–8852, 2021.