Learning Prehensile Dexterity by Imitating and Emulating State-only Observations

Yunhai Han[1], Zhenyang Chen[1], Kyle A Williams[1], Harish Ravichandar[1]

***Abstract*—When human acquire physical skills (e.g from experts, we tend to first learn from merely ob expert. But this is often insufficient. We then engage where we try to emulate the expert and ensure that produce similar effects on our environment. Inspi observation, we introduce *Combining IMitation and for Motion Refinement* (CIMER) – a two-stage fra learn dexterous prehensile manipulation skills fron observations. CIMER's first stage involves *imitation*: ously encode the complex interdependent motions o hand and the object in a structured dynamical sy results in a reactive *motion generation* policy that reasonable *motion prior*, but lacks the ability to re contact effects due to the lack of action labels. stage involves *emulation*: learn a *motion refinement* reinforcement that adjusts the robot hand's motion that the learned *object* motion is reenacted. CIMI *task-agnostic* (no task-specific reward design or sh *intervention-free* (no additional teleoperated or labe strations). Detailed experiments with prehensile dext that i) imitation alone is insufficient, but adding emu tically improves performance, ii) CIMER outperfor... methods in terms of sample efficiency and the ability to generate realistic and stable motions, iii) CIMER can either zero-shot generalize or learn to adapt to novel objects from the YCB dataset, even outperforming expert policies trained with action labels in most cases. Source code and videos are available at https://sites.google.com/view/cimer-2024/.**

I. Introduction

Learning dexterous manipulation skills involving multi-finger hands (e.g., relocation and tool use) presents numerous challenges, such as high-dimensional state and action spaces, complex nonlinear dynamics, and contact effects. Unfortunately, both imitation learning (IL) and reinforcement learning (RL) tend to struggle with such complex tasks when employed alone. Specifically, IL suffers from distribution shift [1], [2] and RL from poor sample efficiency [3]. To address these challenges, prior works have shown promise in leveraging demonstrations to expedite reinforcement learning [4]–[9].

However, collecting demonstrations for dexterous manipulation in the form of state-action pairs can be challenging and cumbersome due to complex system design and setup [10], [11]. To alleviate this burden, recent research has focused on learning dexterous manipulation skills from state-only observations [12]–[14]. However, learning without action labels can result in kinematic motions that are oblivious to force and contact. This lack of knowledge is particularly limiting in dexterous *prehensile* manipulation tasks (e.g., grasping and tool use) as they exhibit heightened sensitivity to applied force [12], [15]–[17]. To compensate, existing methods often require additional teleoperated demonstrations [14], [18], human-in-the-loop corrections [19], task-specific rewards [5], [13], [20], [21], or user-defined sub-goal images [7], [22].

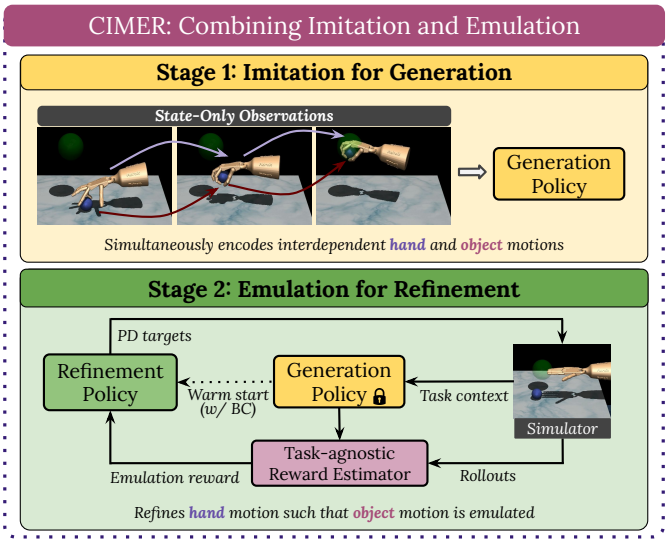

Figure 1: CIMER is a *task-agnostic* and *intervention-free* framework to learn dexterous manipulation skills from state-only observations by learning to first generate interdependent desired motions of the robot hand and the object (*Imitation*), and then refine the generated robot motion in order to reenact the learned object motion (*Emulation*).

In this work, we contribute a novel two-stage framework to learn dexterous prehensile manipulation skills from state-only observations. We call our framework CIMER, short for ***C**ombining **IM**itation and **E**mulation for Motion **R**efinement* (see Fig. 1). The first stage is *Imitation*: CIMER encodes the interdependent desired motions of both the robot hand and the object from the observations into a structured time-invariant dynamical system that acts as a reactive policy and a reasonable *motion prior* (see Sec. III-C). The second stage is *Emulation*: CIMER learns to refine the *robot's* motion prior based on context by optimizing a *task-agnostic* reward that incentivizes the reenactment of the learned *object* motion [23].

Three key insights motivate our design of CIMER. First, dexterous prehensile manipulation involves complex and inherently *interdependent* motions of the robot hand and the object. As such, CIMER's imitation stage captures both their motions simultaneously using a single but structured dynamical system. Second, state-only observations of the expert unambiguously demonstrate how the *object* is supposed to move, and contain valuable (albeit crude) information about the robot hand's motion. As such, CIMER learns to only refines the hand

[1] Georgia Institute of Technology, Atlanta, GA 30332, USA (email: {yhan389, zchen927, kwilliams319, harish.ravichandar}@gdatech.edu).

motion and relies on the learned object motion for reward signals. Third, separating motion generation and refinement amounts to first learning what the robot must do from *state-only* observations, and then learning how to do it via *self-guided* experimentation. Unlike prior methods, CIMER is both *intervention-free* and *task-agnostic* – it neither requires in-context additional demonstrations nor task-specific reward design. See Sec. II for a detailed discussion of related work.

To both experimentally investigate the central issues and to evaluate CIMER, we conducted a series of thorough experiments on three dexterous prehensile manipulation tasks: *Tool Use*, *Object Relocation*, and *Door Opening*.

First, we demonstrate that pure imitation (without emulation) is rarely successful, and necessitates meticulous expert tuning of the low-level controller when it does. But combining imitation and emulation (i.e., CIMER) significantly improves the task success rate, even while using the *same* controller across tasks. We also uncover qualitative insights into how CIMER refines motions and why they succeed.

Second, we demonstrate that CIMER tends to achieve better sample efficiency compared to existing methods that can learn from state-only observations. Moreover, we highlight CIMER's ability to effectively leverage motion priors (encoded in a dynamical system) to produce realistic and stable motions, in contrast to baselines that tend to exploit the simulator and generate aggressive behaviors. We also show that policies trained using CIMER exhibit better robustness against changes to physical parameters such as mass and damping.

Third, we show that CIMER significantly outperforms the baselines in terms of zero-shot generalization to 17 unseen objects from the YCB dataset [24]. Notably, CIMER surpasses even the expert policy trained with action labels in many instances. When CIMER can't readily generalize to a novel object with significant differences, it can effectively leverage previously learned skills to expedite adaptation.

In summary, we contribute CIMER – a two-stage framework that effectively combines imitation and emulation in order to learn dexterous prehensile manipulation skills from state-only observations. CIMER first learns a reasonable motion prior by imitating observations without relying on action labels or user interventions, and then employs a task-agnostic and self-guided strategy to learn how to refine the *hand's* motion so that it emulates the learned *object* motion.

II. Related Work

In this section, we discuss our contributions within the context of different related bodies of work.

Learning dexterity from observations: Given the abundance of video data on the internet and the advances in motion retargetting (e.g., [5]), recent research has focused on learning dexterous manipulation skills from videos or state-only observations in general. A key challenge in learning from videos or state-only observations is the lack of action labels that speaks to the contact effects between the robot, the object, and the environment. Existing works circumvent this challenge in one of two ways. Some methods provide robots an opportunity to learn from interactions, either in simulation [5], [9], [25], [26] or in the real world [7], [8], [22]. Other methods leverage additional in-domain teleoperated demonstrations from an expert [6], [14], [18], [20], [27]. Our work falls under the first category since it does not rely on any additional demonstrations. Unlike CIMER, many of the prior methods in the first category are limited to replicating a specific hand motion from a video clip, such as object grasping [8], [9], [25], [26], bagel flipping [6] or bottle opening [27]. While the remaining methods are capable of learn a variety of skills and can handle changes to the objects' initial and target configurations, they require at least one of the following: task-specific reward engineering [5], [20], user-provided sub-goal images as reward signals [7], [22], and bespoke refinements tailored to dexterous grasping [12], [28]. In contrast, CIMER can learn a variety of skills requiring neither task-specific reward shaping nor expert interventions.

Learning dexterous teleoperation: Another class of existing methods enable a human operator to seamlessly teleoperate a dexterous robotic hand by leveraging learning to close the embodiment gap [11], [29]–[31]. However, it is important to note that these methods do *not* learn autonomous policies. Instead, they rely on real-time corrections and guidance from the human operator who tends to compensate for the robot's inability to reason about contact effects [32].

Learning low-dim skills from videos: There is a large body of work that focuses on learning polices for low-dimensional manipulators (serial-link robots with parallel jaw grippers) from videos. One group of methods utilize a hierarchical framework to learn task-specific waypoint trajectories from human videos that can then be tracked using standard controllers [33]–[36]. Other methods learn a high-level planner that operates in a latent space and then use it to condition a low-level planner that can produce diverse robot behaviors [37]–[39]. However, unlike the low-dimensional skills learned by these methods, dexterous manipulation with multi-fingered hands inherently involves more complex finger-object interactions and contact effects. Further, similar to their counterparts that learn dexterous skills, many of these methods also rely on additional teleoperated demonstrations.

Learning locomotion from observations: Our approach to emulation is inspired by research in learning locomotion skills by imitating animals [40] or animated characters [41], [42]. These methods first extract the reference motion from video clips, and refine the learned motions to account for foot-ground contact effects and to ensure that the resulting gaits closely track the demonstrated ones. However, merely imitating the observed robot motion is unlikely to succeed in dexterous manipulation since it is object-centric (see Sec. IV-B for empirical validation). In contrast, we emphasize the reenactment of the learned *object* motion.

III. Approach

We begin by formulating the problem of learning dexterous manipulation skills from state-only observations.

A. Problem Formulation

Let $\mathcal{D} = [\{\mathrm{h}_t^{(1)}, \mathrm{o}_t^{(1)}\}_{t=1}^{t=T^{(1)}}, \cdots, \{\mathrm{h}_t^{(N)}, \mathrm{o}_t^{(N)}\}_{t=1}^{t=T^{(N)}}]$ denotes a dataset of N state-only observations of a prehensile

manipulation skill, where $\mathrm{h}_t^{(n)} \in \mathcal{H} \subseteq \mathbb{R}^n$ and $\mathrm{o}_t^{(n)} \in \mathbb{R}^m$ respectively denote the robot hand state and objec (e.g., hand joint positions and object 6D poses) at tin n-th observation. We focus on prehensile manipulatio since they are particularly challenging to learn without labels. (see Sec. IV-B). Our goal is to learn a polic the dataset $\mathcal{D}$ that will enable the robot to autono perform the associated dexterous skill. In addition dataset, we assume access to a simulator. But, to ensure *agnostic* solution, we do not allow task-specific reward or shaping. To remain *intervention-free*, we do not a access to additional labeled demonstrations. As suc problem requires an approach that learns a robust de manipulation skill purely from state-only observatio self-guided interactions with a simulator.

B. Solution Overview

Our framework to address the above problem, Combining IMitation and Emulation for Motion Refinement (CIMER), operates in two stages: i) *Imitation*: learn a **Motion Generation Policy** Φ in the form of a structured dynamical system that encodes the desired reference motions for both the robot and the object from state-only observations, and ii) *Emulation*: learn a **Motion Refinement policy** Ψ that compensates for the lack of action information by refining the desired *robot* motions such that the learned *object* motion is achieved. The overall pseudo-code is given in Appendix D.

C. Motion Generation via Imitation

We view the complex interdependent desired motions of the robot and the object as solutions to a learnable underlying nonlinear behavioral dynamical system [2], [43], [44]. Once learned, these dynamical systems can be integrated to predict the next desired robot state from initial conditions or the current state [2]. In recent work, we developed a framework named KODex [45] that learns such underlying dynamics in a highly computationally-efficient manner using Koopman operator theory. Data-driven Koopman-based approaches help effectively encode highly nonlinear dynamical systems in a linear system using lifted global linearization [46], [47]. Building on this, CIMER employs a Koopman-based method theory to learn the Motion Generation Policy Φ from the dataset $\mathcal{D}$. Note that KODex requires access to action labels, but CIMER does not. Formally, we formulate the Φ as a *linear* dynamical system in the *lifted* space: $\phi(\mathrm{h}_{t+1}, \mathrm{o}_{t+1}) = \mathbf{K}\,\phi(\mathrm{h}_t, \mathrm{o}_t)$, where $\phi : \mathbb{R}^{n+m} \rightarrow \mathbb{R}^p$ (with $p > (n+m)$) is a user-defined function that "lifts" the robot hand and object states to a higher-dimensional space in which the underlying dynamics appear linear, and $\mathbf{K}$ is the Koopman matrix [46], [47]. We use a *task-agnostic* second-order polynomial lifting function for all tasks in all our experiments. A key benefit of using a Koopman-based approach is that we can *analytically* compute $\mathbf{K}$ (and thereby our motion generation policy Φ) from $\mathcal{D}$ using least squares [47]. To extract the original states without a decoder, we augment the original states to the lifted states before learning the Koopman operator. See Appendix E for details. We then use the learned dynamical system as an autonomous policy to predict desired hand and object motions for any initial condition.

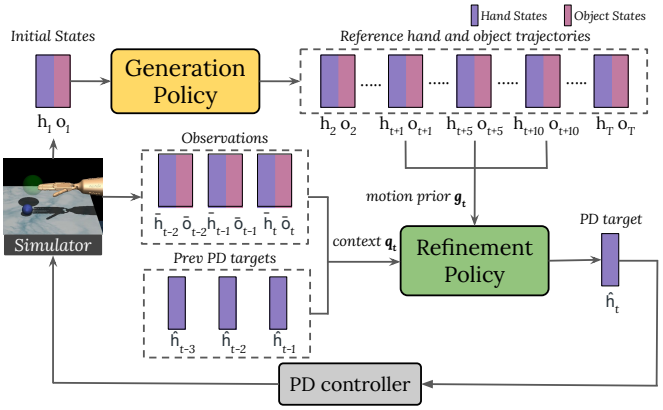

Figure 2: CIMER generates a motion prior based on initial conditions and refines it based on context to generate PD targets for the hand.

D. Motion Refinement via Emulation

Merely tracking the hand trajectory generated by the motion generation policy Φ does not guarantee task success in dexterous prehensile manipulation tasks due to the small margin for error and the lack of reasoning about contact effects (see Sec. IV-B for empirical validation). As such, CIMER refines the robot's *hand* motion so as to ensure emulation of the learned *object* motion. Specifically, CIMER learns Ψ to refine finger and palm trajectories (as opposed to all the DoFs) generated by Φ such that the robot reenacts the object motion generated by Φ. Interacting with the object helps compensate for the lack of action labels and account for contact effects. Once learned, CIMER employs a *task-agnostic* PD controller to track the refined hand motions.

We formulate emulation as an RL problem and maximize rewards computed from policies learned via imitation. Our task-agnostic reward function captures the essence of emulation: reenacting the desired object motion rather than only the expert's motion [23]. To reduce training variance, we use Generalized Advantage Estimator (GAE) [48] and PPO [49].

State & Action Space: We parameterize Ψ as an MLP network that predicts the current action: $\hat{\mathrm{h}}_t = \Psi(\mathrm{q}_t, \mathrm{g}_t)$. Here, $\hat{\mathrm{h}}(t) \in \mathcal{H}$ contains the refined PD targets of the robot hand at time t, $\mathrm{q}_t = (\bar{\mathrm{h}}_{t-2:t}, \bar{\mathrm{o}}_{t-2:t}, \hat{\mathrm{h}}_{t-3:t-1})$ is the *context* containing the history of hand and object states and refined PD targets, and $\mathrm{g}_t = (\mathrm{h}_{t+1,t+5,t+10}, \mathrm{o}_{t+1,t+5,t+10})$ provides the *motion prior* (predicted by the motion generation policy Φ) that needs to be refined. Since the necessary refinements are likely to be local and fine-grained, we restrict Ψ to only refine the finger and palm trajectories. See Fig. 2 for an illustration.

Rewards: Unlike prior works that rely on task-specific reward shaping, we use tracking errors of both the robot hand and the object as reward signals. Formally, we define the reward function as $r_t = r_t^h + r_t^o + r_t^b$, where $r_t^h = \exp\left(-k^h||\bar{\mathrm{h}}_{t+1} - \mathrm{h}_{t+1}||^2\right)$, $r_t^o = \exp\left(-k^o||\bar{\mathrm{o}}_{t+1} - \mathrm{o}_{t+1}||^2\right)$, r_t^b is a bonus reward that is triggered when the object tracking error is smaller than a threshold ϵ_o. Here, $\bar{\mathrm{h}}_{t+1}, \mathrm{h}_{t+1}$, and $\bar{\mathrm{o}}_{t+1}, \mathrm{o}_{t+1}$ denote the hand and object states from both the robot and reference

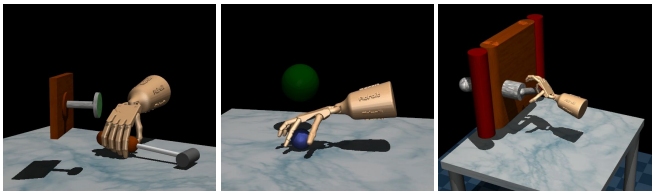

Figure 3: We evaluate on three dexter
Tool Use (left), Object Relocation (cent

motion, respectively. Note that r_t^h
the reference hand motion h_{t+1}
robot hand to emulate the referen
include the training parameters in

Warm Start: Since the motion ge
a reasonable motion prior, we exp
policy Ψ to only make local adj
contact effects. As such, we use t
Φ to warm start Ψ by minimizing
counter covariate shift, we inject
icy's input and use the same output
[52]. This warm start represents
imitation influences the emulation
the reward function and the policy

IV. Experimental

Our experiments pose the follow
tion necessary? (Sec. IV-B), ii) Is
cient than baselines? (Sec. IV-C),
robust and realistic motions? (Sec.
generalize and adapt to novel objects (Sec. IV-D).

A. Experimental Design

Evaluation Platform: We used the widely-used ADROIT Hand [4] – a 30-DoF simulated system (24-DoF hand + 6-DoF floating base) built with MuJoCo Simulator [53].

Data and Expert Policy: We utilized expert DAPG policies [4] trained using action labels to generate 200 state-only observations (both hand and object states) for each task.

Tasks: We consider three widely-used prehensile tasks originally proposed in [4] (see Fig. 3 and Appendix A).

- *Tool Use*: Pick up the hammer to drive the nail into the board placed at a randomized height.
- *Object Relocation*: Move an object to a randomized target location (green sphere).
- *Door Opening*: Given a randomized door position, unlock the latch and pull the door open.

To ensure fairness and reproducibility, we adopt the same success criteria originally proposed in [4].

Metrics: We quantify performance in terms of *Task success rate* (see [4], [13], [45] for criteria) and sample efficiency, reported across five random seeds unless specified otherwise.

B. Need for Emulation

We first demonstrate why it insufficient to imitate state-only observations without emulation. To this end, we compare CIMER against three baselines that solely rely on imitation:

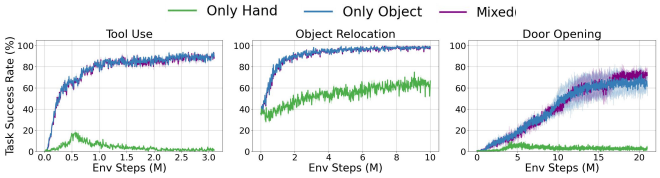

Figure 4: Emulating the observed object motion is significantly more

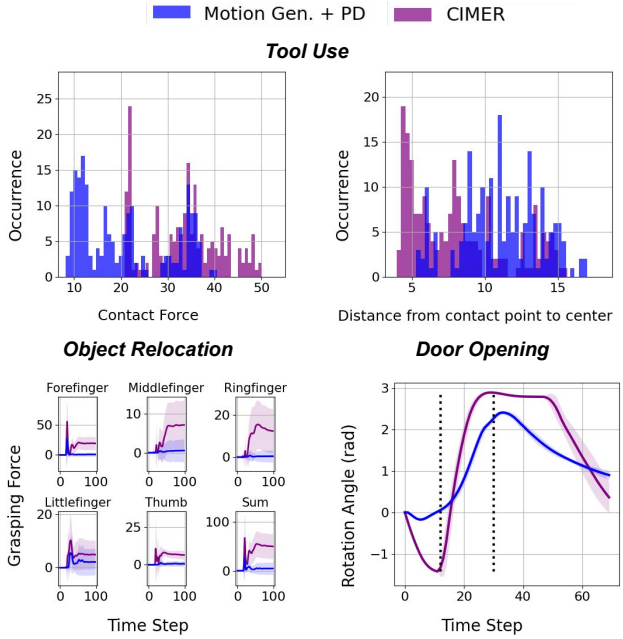

Figure 5: Intuitive refinements emerge from CIMER's emulation. *Top*: Hand ensures hammer hits closer to nail's center, and applies larger driving force; *Bottom Left*: Fingers exert more force to ensure firmer grasps and stable transport; *Bottom Right*: Hand rotates faster to boost momentum while turning door handle (enclosed by dotted lines).

a). *Expert Obs. [4] + PD*: Reference hand motion generated by the expert policy trained with action labels, b). *Motion Gen. + PD*: Reference hand motion generated by Φ, and c). *DexTransfer [12] + PD*: Reference hand motion generated by a policy trained on an augmented dataset (generated by simulating perturbed observations and augmenting new trajectories deemed successful by task-specific success criteria). We used the same PD controller with identical gains for all tasks and methods, and report success rates over 200 initial conditions.

Table I: Tracking state trajectories from an expert or an imitation policy is insufficient for prehensile manipulation. Combing imitation and emulation (CIMER) dramatically improves task performance.

Task Success Rate \ Task / Policy	Tool	Relocation	Door
Expert [4] Obs. + PD	28.0%	34.5%	10.5%
Motion Gen. + PD	32.0%	35.5%	10.5%
DexTransfer [12] + PD	46.6(±3.2)%	24.2(±0.7)%	9.3(±0.8)%
CIMER + PD	**92.7**(±**1.7**)%	**96.0**(±**0.5**)%	**76.3**(±**13.4**)%

As seen in Table I, CIMER significantly outperforms all three baselines. These results support our claim that merely tracking an observed expert trajectory is insufficient for prehensile dexterous manipulation since contact effects and grasp forces are ignored when merely imitating state-only observations. Though DexTransfer uses the simulator to evaluate

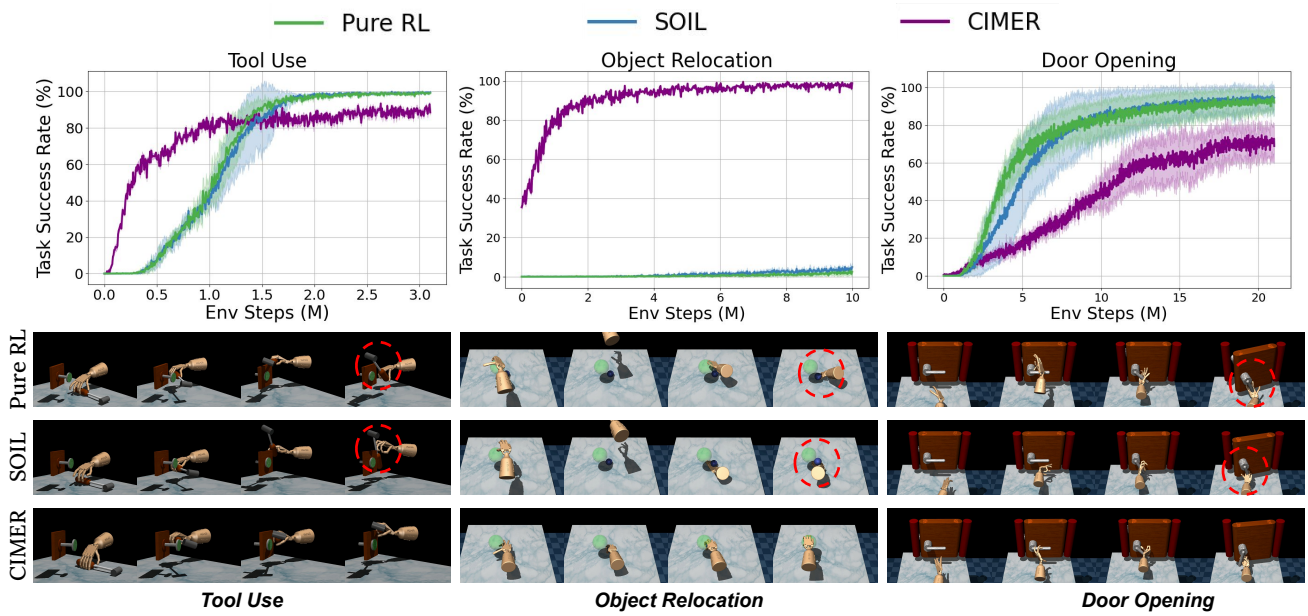

Figure 6: *Top*: CIMER is considerably more sample efficient than the baselines in two tasks (*Tool Use* and *Object Relocation* tasks). The *Door Opening* task is an exception likely due to larger refinements being necessary to firmly grasp and swing the door open. *Bottom*: Qualitative analyses reveals that both the baselines exploit the simulator and generate unrealistic and aggressive behaviors in all three tasks (see dashed red circles indicating loss of grasp, bouncing objects, and hand-object penetration). In contrast, CIMER generates realistic motions as its refinement is anchored by the motion priors from its imitation phase.

the effectiveness of perturbed trajectories, it only learns from successful trajectories and disregards ones that fail. In contrast, CIMER uses emulation to "practice" what it learned during imitation, refining the hand motion to reproduce the observed *object* motion. We also observed that refining only the robot finger and palm motions may not always be sufficient. Specifically, we found that doing so improves task success to only 46.5($\pm$3.4)% in the *Door Opening* task. However, the success rate jumps to 76.3($\pm$13.4)% when CIMER also refines the motion of the 6-DoF hand base. This more comprehensive refinement is likely helping the hand generate larger momentum to successfully turn the door handle.

A key aspect of our approach to emulation is the increased focus on the object. To validate this focus, we evaluated the relative importance of hand- and object-tracking rewards. Results in Fig. 4 highlight the crucial role of object tracking rewards. Importantly, using hand tracking rewards alone results in little to no performance gains.

To better understand how emulation improves performance, we qualitatively analyzed the modifications made by the motion refinement policy to compensate for the lack of action labels in each task (see Fig. 5). *Tool Use*: The hammer hits the nail closer to its center point and with greater force (measured by a touch sensor on the nail head), helping drive the nail completely into the board. *Object Relocation*: the grasping force applied from each fingertip on the object is noticeably increased, producing a more secure grasp during the initial contact and transportation towards the target. *Door Opening*: the hand initially rotates in one direction, then swiftly changes direction to generate greater momentum for turning the handle (the dashed lines denote the handle turning phase). These observations suggest that emulation indeed nudges the robot to apply the necessary forces and adjusts the hand's motion so that the object is manipulated as desired.

C. Sample Efficiency and Realism

Though no existing methods can accommodate all the constraints and challenges of our problem setting (see Sec. III-A), we compare CIMER to the following baselines to evaluate its relative benefits in terms of sample efficiency and realism:

- *Pure RL*: We trained TRPO [54] as a pure RL baseline without offline data, but with task-specific reward shaping [4] to evaluate the benefits of leveraging observations.
- *SOIL* [13]: We also evaluated CIMER against this SOTA state-only imitation method for dexterous manipulation, which also leverages both expert observations and interactions with a simulator. Unlike CIMER, SOIL also requires task-specific reward engineering [4].

For both baselines, we utilized the same model architectures and training parameters as recommended in their original works. We report all results over five random seeds.

As shown in Fig. 6 (top), CIMER exhibits significantly better sample efficiency than SOIL and Pure RL for both *Tool Use* and *Object Relocation* tasks. In particular, while the baselines struggle to make any progress on the *Object Relocation* task, CIMER achieved 100% task success rate. This is likely the baselines end up learning a fragile strategy of bouncing the ball against the table instead of grasping it (see Fig. 6 (bottom)). In contrast, baselines were able to achieve higher success rates in the *Door Opening* task with fewer samples compared to CIMER. This is partly due to the door opening task likely requiring more significant motion refinements to generate enough momentum that will swing the door open. Further, unlike the baselines, CIMER does not benefit from task-specific reward design.

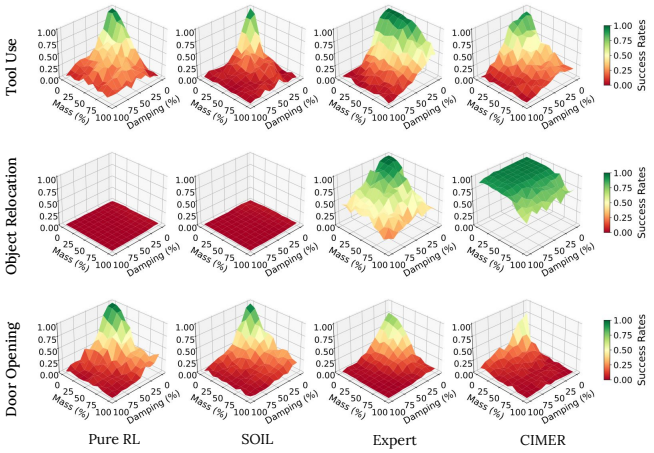

Figure 7: The evaluation of the robustness of each method to systematic changes in mass and damping coefficients.

A closer examination revealed something important: the baselines tend to exploit the simulator and generate unrealistic or aggresive motions across all tasks. Exemplar rollouts in Fig. 6 (bottom) show that the baselines tend to drive the hand to "throw" the hammer, hit and bounce the ball, and penetrate the door handle, while CIMER generates more realistic and stable motions. This is because CIMER's motion generation policy employs a structured dynamical system that provides a realistic and effective motion prior for the motion refinement policy to adjust. These results also suggest that CIMER is better suited for real-world deployment and will not likely require significant efforts in ensuring safety and efficacy.

To quantitatively evaluate realism and the suitability for real-world implementation, we evaluated the robustness of each method to systematic changes in all mass and damping coefficients (see Appendix C for details). The results reveal that CIMER results in comparable or better robustness, even outperforming the expert policy trained using action labels on the *Relocation* task. Though Pure RL and SOIL seem to be more robust than CIMER on the *Door* task, note that qualitative analysis revealed that both Pure RL and SOIL tend to exploit the simulator and generate unrealistic and agrressive motions (see Fig. 6 (bottom)).

D. Generalization and Adaptation to Novel Objects

In addition to the objects considered thus far, we evaluated CIMER and the baselines on their ability to generalize to 17 novel objects (3 synthetic objects and 14 objects from the YCB dataset [24]) in the *Object Relocation* task (see Fig. 8). Note that the Ball is the only default object used for training the expert policy and for collecting state-only demonstrations.

1) Zero-shot Generalization: We report the zero-shot generalization performance of each method and the expert across five random seeds in Fig. 9. Given that both the baselines struggled to learn to relocate the default Ball object, it is no surprise that they consistently fail to generalize to novel objects. While the expert achieves a 100% success rate on the default object (ball) with the benefit for action labels, CIMER notably outperforms the expert on the majority of the novel objects. This impressive ability to readily generalize to novel objects could be attributed to CIMER's hierarchical structure and suggests that CIMER's motion prior and refinement strategy are somewhat robust to changes in the object's geometric and kinematic properties. Further, CIMER demonstrates the most consistent performance across different training seeds, as indicated by shorter error bars.

2) Skill Transfer to New Objects: We also evaluated if transferring the skills learned on the default object (Ball) to novel objects would improve sample efficiency (see Appendix G for experiments on learning from scratch). Freezing the motion generation policy, we fine-tuned CIMER's motion refinement policy on six novel objects (each of which resulted in 50% or lower success rate during zero-shot generalization). We compared against a similarly-finetuned expert policy [4], and two additional policies trained from scratch (one using CIMER and the other using the expert method). As shown in Fig. 10, transferring motion refinement skills considerably improves sample efficiency in four of the six objects. On these four objects, CIMER with skill transfer outperforms the baselines trained from scratch as well as the fine-tuned expert. In contrast, skill transfer hurts sample efficiency when dealing with other two objects. This is likely because skill transfer is counter-productive when target objects are drastically different from the source object, requiring a significantly different grasping strategy. In such cases, CIMER has to "unlearn" the previous skill before adapting to the novel object.

V. Limitations and Future Work

Our work represents the first task-agnostic and intervention-free approach to learning dexterous prehensile manipulation skills from state-only observations. But it still leaves behind several avenues for further improvement. First, given that human videos are ubiquitous online, one could integrate recent advances in motion retargetting [5] with CIMER to directly learning from videos. Second, we do not consider the manipulation of deformable and fragile objects. It is yet unclear how to represent such objects in way that makes incentivizing emulation easier and tractable. Third, some objects and contexts require significantly different strategies (e.g., grasping a pencil to write vs. a ball to throw). It might be possible to enable such drastic adaptations by refining all available degrees of freedom of the hand and accounting for affordances. Fourth, reasoning about local contact events using tactile sensors [55]–[57] might help improve refinement and tackle contact-rich tasks (e.g., peg-in-hole). Last, while our robustness experiments show

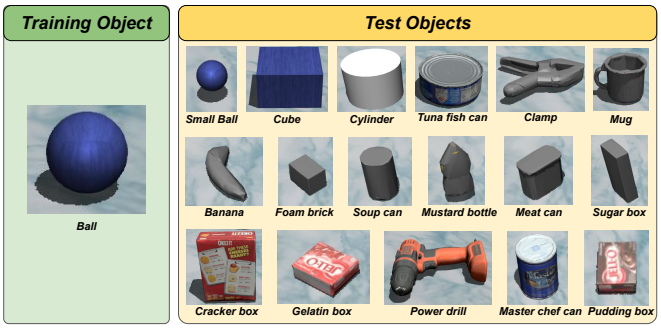

Figure 8: We trained candidate policies on one object, and evaluated their ability to generalize and adapt to 17 novel objects.

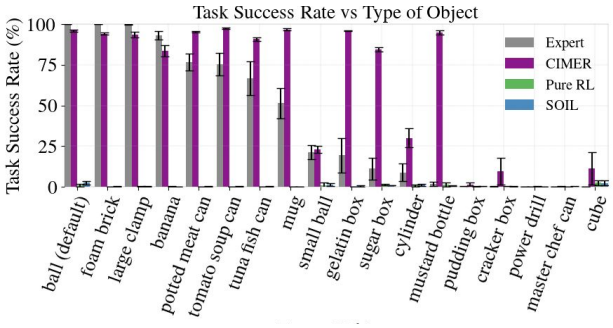

Figure 9: CIMER provides impressive zero-shot generalization to novel objects, even outperforming the expert policy in most cases.

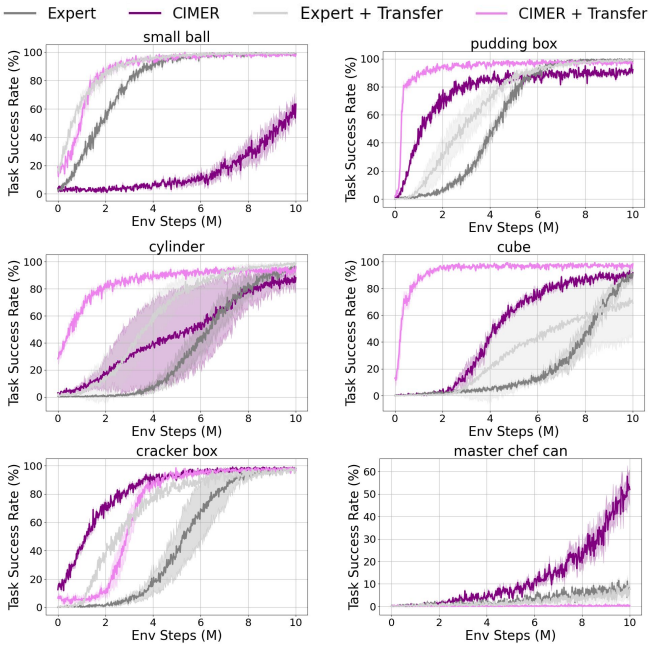

Figure 10: Skill transfer significantly boosts CIMER's sample efficiency (top two rows), unless the target object is considerably different from the source (bottom row). Solid lines show mean trends and shaded areas show $\pm$ standard deviation over five random seeds.

Leveraging the fact that it generates PD targets, we plan to deploy CIMER on hardware and validate its benefits.

References

[1] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *International conference on artificial intelligence and statistics*, 2011.

[2] H. Ravichandar, A. S. Polydoros, S. Chernova, and A. Billard, "Recent advances in robot learning from demonstration," *Annual review of control, robotics, and autonomous systems*, vol. 3, pp. 297–330, 2020.

[3] S. E. Li, "Deep reinforcement learning," in *Reinforcement learning for sequential decision and optimal control*, pp. 365–402, Springer, 2023.

[4] A. R. et al., "Learning Complex Dexterous Manipulation with Deep Reinforcement Learning and Demonstrations," in *Proceedings of Robotics: Science and Systems (RSS)*, 2018.

[5] 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 *European Conference on Computer Vision*, 2022.

[6] S. Haldar, J. Pari, A. Rai, and L. Pinto, "Teach a robot to fish: Versatile imitation from one minute of demonstrations," in *Proceedings of Robotics: Science and Systems (RSS)*, 2023.

[7] K. Xu, Z. Hu, R. Doshi, A. Rovinsky, V. Kumar, A. Gupta, and S. Levine, "Dexterous manipulation from images: Autonomous real-world rl via substep guidance," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

[8] A. Kannan, K. Shaw, S. Bahl, P. Mannam, and D. Pathak, "Deft: Dexterous fine-tuning for real-world hand policies," in *Conference on Robot Learning*, 2023.

[9] P. Mandikal and K. Grauman, "Learning dexterous grasping with object-centric visual affordances," in *IEEE international conference on robotics and automation (ICRA)*, 2021.

[10] 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)*, pp. 657–663, IEEE, 2015.

[11] 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 *IEEE International Conference on Robotics and Automation (ICRA)*, 2020.

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

[13] I. Radosavovic, X. Wang, L. Pinto, and J. Malik, "State-only imitation learning for dexterous manipulation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021.

[14] K. Shaw, S. Bahl, and D. Pathak, "Videodex: Learning dexterity from internet videos," in *Conference on Robot Learning*, pp. 654–665, 2023.

[15] A. Bicchi, "On the closure properties of robotic grasping," *The International Journal of Robotics Research*, vol. 14, no. 4, pp. 319–334, 1995.

[16] Y. Zheng and W.-H. Qian, "Coping with the grasping uncertainties in force-closure analysis," *The international journal of robotics research*, vol. 24, no. 4, pp. 311–327, 2005.

[17] X. Zhu and J. Wang, "Synthesis of force-closure grasps on 3-d objects based on the q distance," *IEEE Transactions on robotics and Automation*, vol. 19, no. 4, pp. 669–679, 2003.

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

[19] C. Wang, H. Shi, W. Wang, R. Zhang, L. Fei-Fei, and C. K. Liu, "Dexcap: Scalable and portable mocap data collection system for dexterous manipulation," *arXiv preprint arXiv:2403.07788*, 2024.

[20] Y. Qin, H. Su, and X. Wang, "From one hand to multiple hands: Imitation learning for dexterous manipulation from single-camera teleoperation," *IEEE Robotics and Automation Letters*, vol. 7, no. 4.

[21] Q. She, R. Hu, J. Xu, M. Liu, K. Xu, and H. Huang, "Learning high-dof reaching-and-grasping via dynamic representation of gripper-object interaction," *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, vol. 41, no. 4, pp. 97:1–97:14, 2022.

[22] Z. Hu, A. Rovinsky, J. Luo, V. Kumar, A. Gupta, and S. Levine, "Reboot: Reuse data for bootstrapping efficient real-world dexterous manipulation," in *7th Annual Conference on Robot Learning*, 2023.

[23] S. Schaal, "Is imitation learning the route to humanoid robots?," *Trends in cognitive sciences*, 1999.

[24] B. Calli, A. Walsman, A. Singh, S. Srinivasa, P. Abbeel, and A. M. Dollar, "Benchmarking in manipulation research: Using the yale-cmu-berkeley object and model set," *Robotics & Automation Magazine*, 2015.

[25] 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 (ICRA)*, 2023.

[26] A. Agarwal, S. Uppal, K. Shaw, and D. Pathak, "Dexterous functional grasping," in *7th Annual Conference on Robot Learning*, 2023.

[27] S. P. Arunachalam, I. Güzey, S. Chintala, and L. Pinto, "Holo-dex: Teaching dexterity with immersive mixed reality," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2023.

[28] Q. Chen, K. V. Wyk, Y.-W. Chao, W. Yang, A. Mousavian, A. Gupta, and D. Fox, "Learning robust real-world dexterous grasping policies via implicit shape augmentation," in *Conference on Robot Learning*, 2023.

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

[30] 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)*, pp. 416–422, IEEE, 2019.

[31] A. Sivakumar, K. Shaw, and D. Pathak, "Robotic telekinesis: Learning a robotic hand imitator by watching humans on youtube," in *Robotics: Science and Systems (RSS)*, 2022.

[32] R. Li, H. Wang, and Z. Liu, "Survey on mapping human hand motion to robotic hands for teleoperation," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 32, no. 5, pp. 2647–2665, 2022.

[33] Y. Liu, A. Gupta, P. Abbeel, and S. Levine, "Imitation from observation: Learning to imitate behaviors from raw video via context translation," in *International Conference on Robotics and Automation (ICRA)*, 2018.

[34] P. Sharma, D. Pathak, and A. Gupta, "Third-person visual imitation learning via decoupled hierarchical controller," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[35] H. Xiong, Q. Li, Y.-C. Chen, H. Bharadhwaj, S. Sinha, and A. Garg, "Learning by watching: Physical imitation of manipulation skills from human videos," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7827–7834, IEEE, 2021.

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

[37] C. Lynch, M. Khansari, T. Xiao, V. Kumar, J. Tompson, S. Levine, and P. Sermanet, "Learning latent plans from play," in *Conference on robot learning*, 2020.

[38] C. Wang, L. Fan, J. Sun, R. Zhang, L. Fei-Fei, D. Xu, Y. Zhu, and A. Anandkumar, "Mimicplay: Long-horizon imitation learning by watching human play," in *Conference on Robot Learning*, 2023.

[39] M. Xu, Z. Xu, C. Chi, M. Veloso, and S. Song, "Xskill: Cross embodiment skill discovery," in *Conference on Robot Learning*, 2023.

[40] X. B. Peng, E. Coumans, T. Zhang, T.-W. Lee, J. Tan, and S. Levine, "Learning agile robotic locomotion skills by imitating animals," in *Robotics: Science and Systems (RSS)*, 2020.

[41] X. B. Peng, P. Abbeel, S. Levine, and M. Van de Panne, "Deepmimic: Example-guided deep reinforcement learning of physics-based character skills," *ACM Transactions On Graphics*, vol. 37, no. 4, pp. 1–14, 2018.

[42] H. Zhang, Y. Yuan, V. Makoviychuk, Y. Guo, S. Fidler, X. B. Peng, and K. Fatahalian, "Learning physically simulated tennis skills from broadcast videos," *ACM Trans. Graph.*

[43] M. Xie, A. Handa, S. Tyree, D. Fox, H. Ravichandar, N. D. Ratliff, and K. V. Wyk, "Neural geometric fabrics: Efficiently learning high-dimensional policies from demonstration," in *Proceedings of The 6th Conference on Robot Learning*, 2023.

[44] S. Bahl, M. Mukadam, A. Gupta, and D. Pathak, "Neural dynamic policies for end-to-end sensorimotor learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 5058–5069, 2020.

[45] Y. Han, M. Xie, Y. Zhao, and H. Ravichandar, "On the utility of koopman operator theory in learning dexterous manipulation skills," in *Conference on Robot Learning*, 2023.

[46] M. O. Williams, I. G. Kevrekidis, and C. W. Rowley, "A data–driven approximation of the koopman operator: Extending dynamic mode decomposition," *Journal of Nonlinear Science*, vol. 25, pp. 1307–1346.

[47] A. Mauroy, Y. Susuki, and I. Mezić, *Koopman operator in systems and control*. Springer, 2020.

[48] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-dimensional continuous control using generalized advantage estimation," in *International Conference on Learning Representations*, 2016.

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

[50] A. Nair, B. McGrew, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Overcoming exploration in reinforcement learning with demonstrations," in *International Conference on Robotics and Automation*, 2018.

[51] L. Ke, J. Wang, T. Bhattacharjee, B. Boots, and S. Srinivasa, "Grasping with chopsticks: Combating covariate shift in model-free imitation learning for fine manipulation," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6185–6191, IEEE, 2021.

[52] P. Florence, L. Manuelli, and R. Tedrake, "Self-supervised correspondence in visuomotor policy learning," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 492–499, 2019.

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

[54] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *ICML*, 2015.

[55] W. Yuan, S. Dong, and E. H. Adelson, "Gelsight: High-resolution robot tactile sensors for estimating geometry and force," *Sensors*, 2017.

[56] Y. Han, K. Yu, R. Batra, N. Boyd, C. Mehta, T. Zhao, Y. She, S. Hutchinson, and Y. Zhao, "Learning generalizable vision-tactile robotic grasping strategy for deformable objects via transformer," *IEEE/ASME Transactions on Mechatronics*, 2024.

[57] K. Yu, Y. Han, M. Zhu, and Y. Zhao, "Mimictouch: Learning human's control strategy with multi-modal tactile feedback," *arXiv preprint arXiv:2310.16917*, 2023.

[58] R. Cheng, A. Verma, G. Orosz, S. Chaudhuri, Y. Yue, and J. Burdick, "Control regularization for reduced variance reinforcement learning," in *International Conference on Machine Learning (ICML)*.

APPENDIX A
STATE SPACE DESIGNS

Our simulation ran at 500 Hz and the state design for each task is as follows:

Tool use: The floating hand base can only rotate along the x and y axes, resulting in $\mathrm{h}_t \in \mathcal{H} \subseteq \mathbb{R}^{26}$. Unlike other tasks, where the objects of interest are directly manipulated by the hand, this task requires indirect manipulation of the nail. As such, in addition to the hammer position, orientation, and corresponding velocities $\mathrm{p}_t^{\text{tool}}, \mathrm{o}_t^{\text{tool}}, \dot{\mathrm{p}}_t^{\text{tool}}, \dot{\mathrm{o}}_t^{\text{tool}}$ ($\mathbb{R}^3$), we define the nail goal position p^{nail} ($\mathbb{R}^3$). Finally, we have $\mathrm{o}_t = [\mathrm{p}_t^{\text{tool}}, \mathrm{o}_t^{\text{tool}}, \dot{\mathrm{p}}_t^{\text{tool}}, \dot{\mathrm{o}}_t^{\text{tool}}, \mathrm{p}^{\text{nail}}] \in \mathcal{O} \subseteq \mathbb{R}^{15}$. We use the $\mathrm{p}_t^{\text{tool}}$ and $\mathrm{o}_t^{\text{tool}}$ as the object tracking states.

Object relocation: The ADROIT hand is fully actuated, so we have $\mathrm{h}_t \in \mathcal{H} \subseteq \mathbb{R}^{30}$ (24-DoF hand + 6-DoF floating wrist base). Regarding the object states, we define $\mathrm{p}^{\text{target}}$ and $\mathrm{p}_t^{\text{ball}}$ as the target and current positions. Then, we compute $\bar{\mathrm{p}}_t^{\text{ball}} = \mathrm{p}_t^{\text{ball}} - \mathrm{p}^{\text{target}}$, which is the component of $\mathrm{p}_t^{\text{ball}}$ in a new coordinate frame that is constructed by $\mathrm{p}^{\text{target}}$ being the origin. We additional include the ball orientation $\mathrm{o}_t^{\text{ball}}$ and their corresponding velocities $\dot{\mathrm{p}}_t^{\text{ball}}$, $\dot{\mathrm{o}}_t^{\text{ball}}$ (all $\mathbb{R}^3$). Finally, we have $\mathrm{o}_t = [\bar{\mathrm{p}}_t^{\text{ball}}, \mathrm{o}_t^{\text{ball}}, \dot{\mathrm{p}}_t^{\text{ball}}, \dot{\mathrm{o}}_t^{\text{ball}}] \in \mathcal{O} \subseteq \mathbb{R}^{12}$. We use the $\bar{\mathrm{p}}_t^{\text{ball}}$ as the object tracking states.

Door opening: For this task, the floating wrist base can only move along the direction that is perpendicular to the door plane but rotate freely, so we have $\mathrm{h}_t \in \mathcal{H} \subseteq \mathbb{R}^{28}$. Regarding the object states, we define the fixed door position p^{door}, which can provide with case-specific information (similar to p^{nail} in *Tool Use* task), and the handle positions $\mathrm{p}_t^{\text{handle}}$ (both $\mathbb{R}^3$). In order to take into consideration the status of door being opened, we include the angular velocity of the opening angle $v_t(\mathbb{R}^1)$. Finally, we have $\mathrm{o}_t = [\mathrm{p}_t^{\text{handle}}, v_t, \mathrm{p}^{\text{door}}] \in \mathcal{O} \subseteq \mathbb{R}^7$. We use the $\mathrm{p}_t^{\text{handle}}$ as the object tracking states.

APPENDIX B
CIMER HYPERPARAMETERS

Table II summarizes the hyper-parameter settings for training CIMER policies. For the three tasks, the hyper-parameter settings are the same.

Table II: Training details

MLP Hidden Layer	RL Learning Rate	GAE_γ
$(256, 128)$	2e-6	0.98
PPO clip threshold	PPO epochs	GAE_λ
0.2	8	0.97

In addition, we used $k^h = 5, k^o = 5, r_t^b = 25$ in all tasks to compute the tracking rewards introduced in Sec. III-D.

APPENDIX C
ROBUSTNESS EXPERIMENTS

For the robustness experiments in Sec. IV-C, we varied mass and damping since they are an important subset of differences between simulation and hardware. Specifically, we varied all mass and damping coefficients along eleven equally spaced evaluation points at $[0\%, 10\%, 20\%, ..., 100\%]$ variation. For each variation, we sampled mass and damping coefficients

from a uniform distribution, with the lower and upper bounds being multiples of the default values. The lower and upper bound values (l_j and u_j) for each variation case (j) are

$$\forall j \in \{0, 0.1, 0.2, ..., 1.0\}, l_j = 1 - 0.8j, u_j = 1 + 4.0j.$$

After sampling a particular combination of mass and damping parameters, we tested each policy for 200 episodes with varying goal locations. We repeated this process 20 times for each variation, and report the resulting average success rates.

APPENDIX D
CIMER PSEUDO-CODE

The overall pseudo-code for CIMER is given below.

Algorithm 1: CIMER

Inputs: State-only observation dataset $\mathcal{D}$, Koopman lifting function $g(\cdot)$;
Motion Imitation
Formulate Motion Generation Policy Φ as a Koopman-based dynamical system: $g(\mathrm{h}_{t+1}, \mathrm{o}_{t+1}) = \mathbf{K}\ g(\mathrm{h}_t, \mathrm{o}_t)$
Regress Koopman Matrix $\mathbf{K}$ on $\mathcal{D}$;
Motion Emulation
Warm-start the Motion Refinement Policy Ψ to reconstruct trajectories generated by Φ;
for $iter \in \{1, ..., \max\}$ **do**
  Initialize replay buffer $\mathcal{B} = \varnothing$;
  for $k \in \{1, ..., M\}$ **do**
    Specify random initial states $\{\mathrm{h}_1^{(k)}, \mathrm{o}_1^{(k)}\}$;
    Generate reference motions $\{\mathrm{h}_t^{(k)}, \mathrm{o}_t^{(k)}\}_{t=2}^{t=T^{(k)}}$ by rolling out Φ;
    for $t \in \{1, ..., T^{(k)} - 1\}$ **do**
      Motion Refinement Policy Ψ inputs s_t, and outputs a_t for execution;
      Compute the tracking reward r_t;
    end
    Add $\{\mathrm{s}_t^{(k)}, \mathrm{a}_t^{(k)}, r_t^{(k)}\}_{t=1}^{t=T^{(k)}-1}$ to $\mathcal{B}$;
  end
  Update the Motion Refinement Policy Ψ with $\mathcal{B}$;
end
Return: Trained policies Φ, Ψ

APPENDIX E
DETAILS OF KODEX TRAINING

Modeling Dexterous Manipulation Skills: A central principle behind KODex is that the desired behavior of a robot can be represented using a dynamical system. To this end, we define the state at time t as $\mathrm{x}(t) = [\mathrm{h}_t{}^\top, \mathrm{o}_t{}^\top]^\top$, where $\mathrm{h}_t \in \mathcal{H} \subseteq \mathbb{R}^n$ and $\mathrm{o}_t \in \mathcal{O} \subseteq \mathbb{R}^m$ represent the state of the robot and the object, respectively, at time t. As such, the dynamical system we wish to capture is

$$\mathrm{x}(t+1) = F^*(\mathrm{x}(t)) \tag{1}$$

where $F^*(\cdot) : \mathcal{H} \times \mathcal{O} \to \mathcal{H} \times \mathcal{O}$ denotes the dynamics that govern the *interdependent* motions of the robot and the object.

A key challenge in learning the dynamical system in (1) is that it can be arbitrarily complex and highly nonlinear, depending on the particular skill of interest. KODex leverages Koopman operator theory to learn a *linear* dynamical system that can effectively approximate such complex nonlinear dynamics. To this end, we first define the Koopman lifting function $g(\cdot)$ as follows

$$g(\mathrm{x}(t)) = [\mathrm{h}_t{}^\top, \psi_h(\mathrm{h}_t), \mathrm{o}_t{}^\top, \psi_o(\mathrm{o}_t)]^\top, \forall t \tag{2}$$

where $\psi_h : \mathbb{R}^n \to \mathbb{R}^{\bar{n}}$ and $\psi_o : \mathbb{R}^m \to \mathbb{R}^{\bar{m}}$ are vector-valued lifting functions that transform the robot and object state respectively.

Now, we introduce the Koopman matrix $\mathbf{K}$, which approximates the system evolution in the lifted state space as follows

$$g(\mathrm{x}(t+1)) = \mathbf{K} g(\mathrm{x}(t)) \tag{3}$$

Learning the Koopman Matrix $\mathbf{K}$: The next step is to learn the Koopman matrix $\mathbf{K}$ from dataset $\mathcal{D} = [\{\mathrm{h}_t^{(1)}, \mathrm{o}_t^{(1)}\}_{t=1}^{t=T^{(1)}}, \cdots, \{\mathrm{h}_t^{(N)}, \mathrm{o}_t^{(N)}\}_{t=1}^{t=T^{(N)}}]$ (Section. III-B). Recall that CIMER's Motion Generation Policy Φ is formulated as the linear dynamical system shown in (3).

As described in [45], we can efficiently compute the Koopman matrix as $\mathbf{K} = \mathbf{A}\mathbf{G}^\dagger$, where $\mathbf{A}$ and $\mathbf{G}$ are shown as follows

$$\begin{aligned} \mathbf{A} &= \sum_{n=1}^{n=N} \sum_{t=1}^{t=T^{(n)}-1} \frac{g(\mathrm{x}^{(n)}(t+1)) \otimes g(\mathrm{x}^{(n)}(t))}{N(T^{(n)}-1)}, \\ \mathbf{G} &= \sum_{n=1}^{n=N} \sum_{t=1}^{t=T^{(n)}-1} \frac{g(\mathrm{x}^{(n)}(t)) \otimes g(\mathrm{x}^{(n)}(t))}{N(T^{(n)}-1)} \end{aligned} \tag{4}$$

where $\mathbf{G}^\dagger$ denotes the Moore–Penrose inverse[1] of $\mathbf{G}$, and $\otimes$ denotes the outer product.

We use the computed Koopman matrix $\mathbf{K}$ to generate rollouts in the lifted state space. However, we need to obtain the rollouts in the original state states to retrieve the predicted robot and object states. Since we designed $g(\mathrm{x}(t))$ such that robot state h_t and the object state o_t are parts of lifted states in (2), we can easily retrieve the reference motions of both the robot and the object $\{\mathrm{h}_t, \mathrm{o}_t\}$ by selecting the corresponding elements in $g(\mathrm{x}(t))$.

APPENDIX F
EMULATION MIGHT NOT BE NECESSARY FOR NON-PREHENSILE MANIPULATION

We present the task success rate on another non-prehensile manipulation task, *In-hand Reorientation*, which are as follows: *Expert Obs. + PD*: 91.0%, *Motion Gen. + PD*: 69.5%, and *CIMER*: 72.7($\pm$1.0)%. Contrary to the results presented in Table. I, where *Expert Obs. + PD* shows poor performance on three dexterous prehensile manipulation tasks, in this scenario, it could achieves satisfactory results. This disparity arises because the challenge of the *In-hand Reorientation* task

[1]It could be efficiently computed using the`scipy.linalg.pinv(`$\mathbf{G}$`)` function from Scipy library.

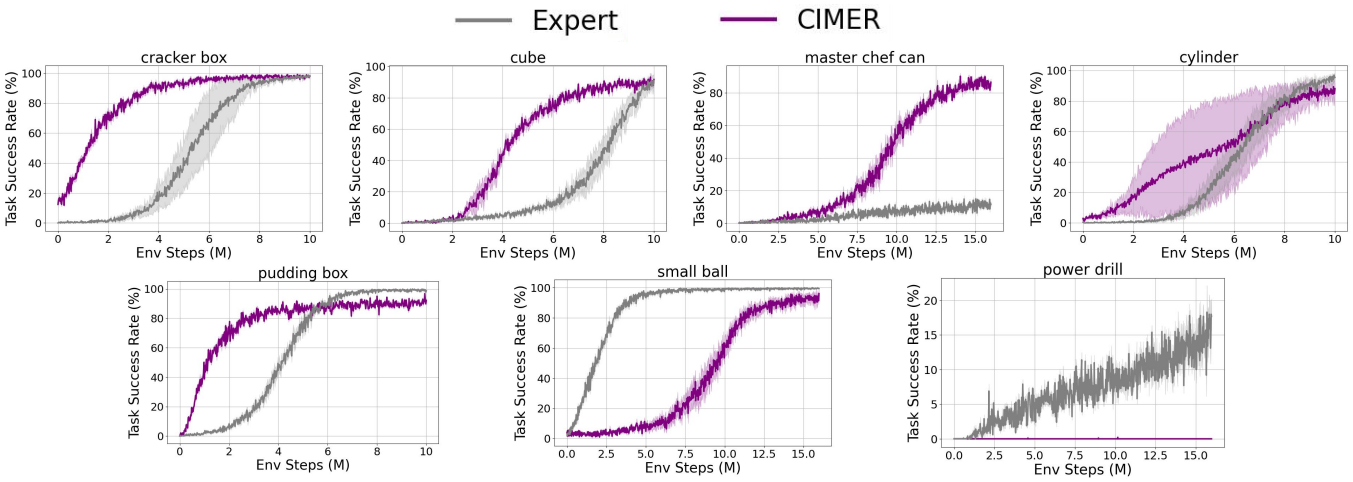

Figure 11: We compare the RL sample efficiency between CIMER and expert policies when being trained on new objects. For each, solid lines indicate mean trends and shaded areas show $\pm$ standard deviation over five random seeds. We find that both methods will gradually learn the manipulation strategy of the new objects, except for the power drill. Moreover, CIMER policies exhibit the better sample efficiency.

primarily pertains to the generation of precise and diverse hand motions, rather than dealing with heightened sensitivity to applied forces. Therefore tracking the observed expert motion directly translates to achieving the desired object motion. This is also why both the *Motion Gen. + PD* and *CIMER* are not comparable to *Expert Obs. + PD* in this *In-hand Reorientation* task. Therefore, we exclusively apply CIMER policies to the three dexterous prehensile manipulation tasks, aligning with the intended purpose of motion refinement policies. We leave the improvement of motion generation policy for non-prehensile tasks as a topic for future work.

Appendix G
Adaptation to New Objects

We also evaluated CIMER's ability to adapt to novel objects. Specifically, we finetuned the CIMER on each of the seven objects for which its success rates was lower than 50%. To ensure that CIMER remains intervention-free, we did not collect or rely on additional observations. Instead, we reused the old motion generation policy (trained on the default object), and simply changed the object used in the simulator when training the motion refinement policy. We similarly trained the expert policies to ensure a fair comparison.

As shown in Fig. 11, we find that both methods gradually learn to manipulate the new objects, with CIMER exhibiting significantly better sample efficiency almost all objects. The power drill presents a notable exception to this trend, with CIMER particularly struggling to adapt. This is due to the power drill likely requiring a significantly different grasping strategy than the default object (Ball) on which CIMER's motion generation policy was trained. Indeed, CIMER's motion refinement only adjusts finger and palm motions, and does not account for grasp synthesis. This could potentially be addressed by also learning to refine the hand base motions with the help of appropriate regularizers [58] that encourage refined motions to remain close to the reference motion.