DEXPLORE: Scalable Neural Control for Dexterous Manipulation from Reference-Scoped Exploration

Sirui Xu[1,2*] **Yu-Wei Chao**[2] **Liuyu Bian**[1] **Arsalan Mousavian**[2]
Yu-Xiong Wang[1†] **Liang-Yan Gui**[1†] **Wei Yang**[2†]
[1] University of Illinois Urbana-Champaign [2] NVIDIA
[†] Equal Advising
https://sirui-xu.github.io/dexplore

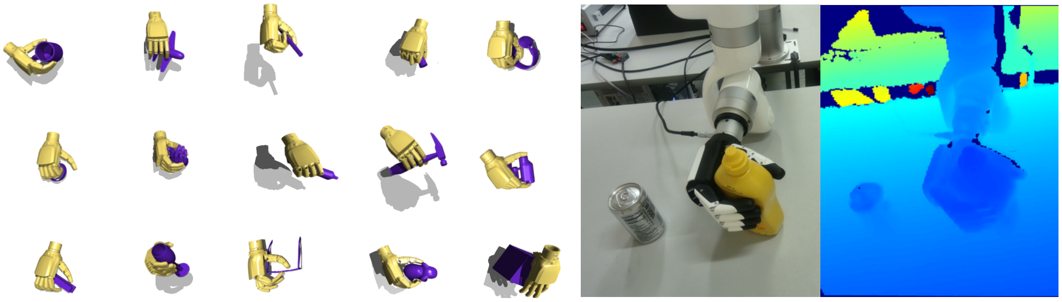

Figure 1: We propose DEXPLORE, a unified control policy that tracks diverse hand-object MoCap references on a dexterous robotic hand [1]. *Left*: By using human demonstrations as soft references, we train a robot hand to discover motions that align with its physical form and a given task's intent through large-scale reinforcement learning. *Right*: We transfer the learned policy into a skill-conditioned generative control policy that takes a partial depth image as input, and successfully deploy the policy on a real robot system.

Abstract: Hand-object motion-capture (MoCap) repositories offer large-scale, contact-rich demonstrations and hold promise for scaling dexterous robotic manipulation. Yet demonstration inaccuracies and embodiment gaps between human and robot hands limit the straightforward use of these data. Existing methods adopt a three-stage workflow, including retargeting, tracking, and residual correction, which often leaves demonstrations underused and compound errors across stages. We introduce DEXPLORE, a unified single-loop optimization that jointly performs retargeting and tracking to learn robot control policies directly from MoCap at scale. Rather than treating demonstrations as ground truth, we use them as soft guidance. From raw trajectories, we derive adaptive spatial scopes, and train with reinforcement learning to keep the policy in-scope while minimizing control effort and accomplishing the task. This unified formulation preserves demonstration intent, enables robot-specific strategies to emerge, improves robustness to noise, and scales to large demonstration corpora. We distill the scaled tracking policy into a vision-based, skill-conditioned generative controller that encodes diverse manipulation skills in a rich latent representation, supporting generalization across objects and real-world deployment. Taken together, these contributions position DEXPLORE as a principled bridge that transforms imperfect demonstrations into effective training signals for dexterous manipulation.

Keywords: Dexterous Manipulation, Reinforcement Learning from Demonstrations, Retargeting

[*] Work done during a part-time internship at NVIDIA Research.

9th Conference on Robot Learning (CoRL 2025), Seoul, Korea.

1 Introduction

Achieving human-level dexterity remains a fundamental and long-standing challenge in robotics and autonomous systems, with broad implications for tasks ranging from object sorting and packaging to food preparation and assisted living [2, 3, 4]. Despite decades of progress, robotic manipulators still struggle with many tasks that humans perform effortlessly [5, 6], highlighting the persistent gap between human and robotic manipulation capabilities.

A natural path toward closing this gap is to learn control policies from human demonstrations. However, translating human data into effective robot control is nontrivial: human hands combine high-dimensional kinematics, compliance, and dense tactile sensing; by contrast, robotic hands often differ markedly in morphology, provide fewer independently actuated degrees of freedom, and offer limited sensing and force control. This embodiment mismatch renders direct transfer inherently challenging. To address this, prevailing approaches retarget human demonstrations to the robot's kinematics, track them with a low-level controller, and add residual or corrective terms to compensate for tracking errors and embodiment gaps [7, 8, 9]. However, retargeting errors can propagate and bias downstream learning, for example, a grasp that is feasible for a human may be infeasible or strongly suboptimal for a robotic hand when, if, *e.g.*, strict fingertip correspondence is enforced.

In this work, we propose DEXPLORE, a paradigm that avoids strict retargeting and post-hoc residual correction. The core idea is to treat demonstrations as soft references that preserve intent while allowing the robot to discover motions compatible with its own embodiment. Concretely, during contact-rich segments, we replace rigid kinematic tracking with adaptive, reference-scoped termination envelopes: at each timestep, the reference induces *spatial scopes* within which a rollout is considered successful. Training begins with wide envelopes and progressively tightens them based on observed success rates, encouraging early exploration and promoting precise tracking whenever feasible. To enable real-world deployment under partial observations, we then distill the learned tracker into a vision-based, skill-conditioned generative control policy: latent skill embeddings capture high-level manipulation intent, and a decoder produces low-level actions conditioned on these latent codes. Training combines imitation from a teacher tracker with distribution matching between privileged and partial state encodings. This structure yields a scalable policy that captures diverse manipulation skills (left of Figure 1) and robustly handles partial observability.

We validate this design in real-world deployment (right of Figure 1) by running the distilled, vision-based policy on a physical dexterous robotic hand using only single-view depth and proprioception at test time. The policy closes the loop at typical control rates on a standard workstation and issues low-level commands to the hand controller; no mocap references, pose estimators, or force sensors are required at runtime. Conditioned on a compact skill code specifying high-level intent, the policy executes grasping with embodiment-aware adaptation.

Our contributions are threefold: **(I)** Our DEXPLORE is a unified single-loop optimization that learns dexterous manipulation directly from human MoCap by treating demonstrations as soft references within adaptive spatial scopes, without explicit retargeting and residual correction. **(II)** We distill the learned state-based tracker into a vision-based, skill-conditioned generative control policy that maps single-view depth and proprioception, together with a latent skill code, to low-level actions. **(III)** We demonstrate successful real-world deployment on a dexterous hand using only single-view depth sensing. Overall, DEXPLORE redefines the role of MoCap data in learning dexterous manipulation, as adaptable guidance that bridges demonstrations and robot-executable skills.

2 Related Work

Learning dexterous manipulation from demonstrations. Human demonstrations are a powerful substrate for learning dexterous skills, typically collected via motion capture (MoCap) or teleoperation and leveraged through imitation learning (IL). At scale, community platforms and toolkits facilitate collecting and exploiting large demonstration corpora [10, 11]. Building on this foundation, recent approaches for multi-fingered hands show that closely aligning robot control with demon-

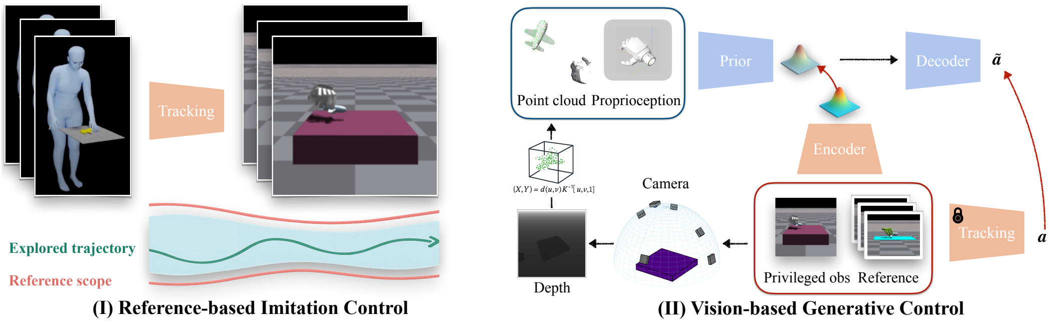

Figure 2: **Overview of DEXPLORE.** (**I**) We first train a state-based policy that acquires scalable dexterous manipulation skills across diverse objects from raw MoCap data, without relying on external retargeting. Rather than rigidly imitating demonstration trajectories, the robot is encouraged to explore within reference-scoped regions, allowing embodiment-specific strategies to emerge. (**II**) We then distill these skills into a vision-based policy that embeds diverse manipulation behaviors within a unified latent space. By sampling from this latent representation and integrating perception with hand proprioception, the policy generates human-like behaviors that generalize across tasks.

strations can reproduce complex hand manipulation behaviors [12, 13, 14, 15, 16, 17, 18, 19, 20]. Generative IL has further improved robustness and long-horizon control, via diffusion policies and action-chunking transformers [21, 22] and, more recently, vision–language–action models [23, 24]. However, learning directly from human data, rather than robot demonstrations, still remains challenging because of embodiment mismatches between human and robotic hands.

Human-to-robot data transfer. To bridge the human-robot embodiment gap, human-to-robot transfer typically proceeds via three routes: (**I**) video-to-robot pipelines that infer hand motion from human videos and synthesize robot trajectories [25]; (**II**) vision-based and AR/VR teleoperation that records robot-compatible demonstrations [26, 27, 28, 29, 30, 31, 32]; and (**III**) task-aware retargeting followed by low-level tracking [33, 7]. While effective, these routes can be labor-intensive (teleoperation), sensitive to perception noise and occlusion (video/vision-based retargeting), and vulnerable to kinematic and force-control mismatches that accumulate during tracking. In contrast, DEXPLORE relaxes strict human-to-robot adherence, preserving demonstrator intent while encouraging embodiment-consistent motion discovery.

Dexterous manipulation via reinforcement learning. Reinforcement learning (RL) provides an alternative to direct imitation, and dexterous skills can be learned even with limited or no demonstration supervision. In practice, demonstrations are often used to bootstrap RL and improve stability and sample efficiency, via on-policy fine-tuning initialized by behavior cloning [34] or off-policy integration using Q-filtering and prioritized replay [35, 36]. Sim-to-real systems such as the Rubik's Cube hand mitigate sensing and dynamics gaps through domain randomization and hindsight experience replay [37, 38, 39]. More recent advances in policy-guided RL from demonstrations and related variants continue to raise in-hand dexterity performance [40, 41, 42, 43, 44, 45, 46, 47, 48, 49]. Nevertheless, state-of-the-art pipelines such as DexTrack [9] and ManipTrans [8] follow multi-stage cascades that involve explicit human-to-robot transfer, tracking, and residual correction, which can underutilize the demonstration signal and propagate errors across stages. In contrast, our DEXPLORE integrates reference guidance and reinforcement learning within a single optimization loop, unifying data retargeting, curation, and policy learning.

3 Methodology

As illustrated in Figure 2, Dexplore learns dexterous manipulation from human demonstrations in two stages. First, a state-based imitation control policy (Sec. 3.1) is trained with Reference-Scoped Exploration (RSE), enabling the robot to discover embodiment-specific manipulation strategies. This policy is then distilled into a vision-based generative control policy (Sec. 3.2)

3.1 Reference-based Imitation Control

Task Formulation. The goal is to learn a policy π that enables a dexterous robot hand to manipulate objects by following reference motions derived from human demonstrations. Due to anatomical and actuation mismatches between the robot and the human hand, the policy must compensate for geometric discrepancies rather than replay the reference verbatim. For the robot hand, the wrist is modeled as a floating root joint in world coordinates, with all remaining joints expressed relative to their parents. All manipulated objects are assumed rigid in the simulation, and their configurations are given by full 6-DoF poses (position and orientation), while we show that our robot system is adaptable for non-rigid objects as shown in Figure A. We formulate reference imitation as a reinforcement learning problem defined over a Markov Decision Process (MDP). The MDP consists of state representations, robot actions, and transition dynamics, together with a reward function designed to encourage faithful yet feasible tracking of the reference motion. We employ Proximal Policy Optimization (PPO) [50] to optimize the policy.

State. At every timestep t, our policy π acts on a hand proprioception and privileged object observation, $\boldsymbol{x}_t$, combined with a goal-oriented reference component $\hat{\boldsymbol{x}}_t$. It aggregates hand kinematics, privileged object positioning, and coarse geometry and tactile cues, namely: $\{\{\boldsymbol{R}^{\mathrm{h}}_t, \boldsymbol{J}^{\mathrm{h}}_t, \boldsymbol{\omega}^{\mathrm{h}}_t, \boldsymbol{v}^{\mathrm{h}}_t\}, \{\boldsymbol{R}^{\mathrm{o}}_t, \boldsymbol{p}^{\mathrm{o}}_t, \boldsymbol{\omega}^{\mathrm{o}}_t, \boldsymbol{v}^{\mathrm{o}}_t\}, \{\boldsymbol{D}_t, \boldsymbol{C}_t\}\}$, where $\boldsymbol{R}^{\mathrm{h}}_t, \boldsymbol{J}^{\mathrm{h}}_t$ represent the joint rotations and positions of the robot's hand, respectively; $\boldsymbol{\omega}^{\mathrm{h}}_t, \boldsymbol{v}^{\mathrm{h}}_t$ denote the angular and linear velocities of the hand joints. Similarly, $\boldsymbol{R}^{\mathrm{o}}_t, \boldsymbol{p}^{\mathrm{o}}_t$ represent the object's orientation and position, with $\boldsymbol{\omega}^{\mathrm{o}}_t, \boldsymbol{v}^{\mathrm{o}}_t$ capturing its angular and linear velocities. We integrate coarse object geometry cues and simplified contact information through two sensory modalities following [51]: (i) $\boldsymbol{D}_t$, consisting of vectors [52] pointing from each hand joint to the nearest surface points of the object, and (ii) $\boldsymbol{C}_t$, binary contact indicators that reflect whether each rigid body on hand is in contact, mimicking tactile sensing [53]. The goal reference component $\hat{\boldsymbol{x}}_t$ is derived from the reference MANO demonstrations, structured as: $\{\hat{\boldsymbol{x}}_{t+k}\}_{k \in K}$, with each future state $\hat{\boldsymbol{x}}_{t+k}$ defined explicitly as: $\{\{\boldsymbol{M}(\hat{\boldsymbol{R}}^{\mathrm{h}}_{t+k}) \ominus \boldsymbol{M}(\boldsymbol{R}^{\mathrm{h}}_t), \boldsymbol{M}(\hat{\boldsymbol{J}}^{\mathrm{h}}_{t+k}) - \boldsymbol{M}(\boldsymbol{J}^{\mathrm{h}}_t)\}, \{\hat{\boldsymbol{R}}^{\mathrm{o}}_{t+k} \ominus \boldsymbol{R}^{\mathrm{o}}_t, \hat{\boldsymbol{p}}^{\mathrm{o}}_{t+k} - \boldsymbol{p}^{\mathrm{o}}_t\}, \{\boldsymbol{M}(\hat{\boldsymbol{D}}_{t+k}) - \boldsymbol{M}(\boldsymbol{D}_t), \boldsymbol{M}(\hat{\boldsymbol{C}}_{t+k}) - \boldsymbol{M}(\boldsymbol{C}_t\}), \{\hat{\boldsymbol{R}}^{\mathrm{h}}_{t+k}, \hat{\boldsymbol{J}}^{\mathrm{h}}_{t+k}, \hat{\boldsymbol{R}}^{\mathrm{o}}_{t+k}, \hat{\boldsymbol{p}}^{\mathrm{o}}_{t+k}\}\}$, with $\ominus$ indicating a rotation difference. Here, K specifies the reference indices that define the goal horizon. Because the robot hand and the demonstration state $(\hat{\cdot})$ are defined on different skeletal structures, we introduce a mapping $\boldsymbol{M}$ that projects both representations onto a common subset of key joints before computing deltas. This mapping plays a role analogous to the correspondence used in retargeting objectives, and we adopt the one provided in [33]. For the Inspire hand [1], for example, the mapping simplifies to the five fingertips. All continuous components are expressed in a coordinate frame anchored at the hand's floating root joint and canonicalized by its current rotation.

Action. The control is divided into two parts: a 6-DoF floating-wrist signal and a set of local finger commands. Each element is a proportional-derivative (PD) target in which rotations are expressed with an exponential-map parametrization. These targets are converted to joint torques to drive the robot. For kinematically coupled (mimic) joints, *e.g.*, in the Inspire hand, the controller scales the driving joint's PD target by its predefined coupling coefficient (originally specified for joint positions) to reproduce the mechanical linkage in hardware. While this approximation may not perfectly capture the true coupling dynamics, it provides a practical control strategy. The policy operates in a residual action space for floating-wrist control, where a 6-DoF positional offset is added to the current wrist state to form the PD target, preventing the network from having to learn unbounded absolute trajectories and thereby improving stability and generalization.

Reward. At each timestep, the reward consists of two complementary components: a state-reference matching term and an energy regularization term. The matching term, $R_{\text{match}}(\boldsymbol{x}_t, \hat{\boldsymbol{x}}_t)$, encourages the simulated state $\boldsymbol{x}_t$ to align with the demonstration $\hat{\boldsymbol{x}}_t$. Following [51], R_{match} combines (i) kinematic alignment of hand joint rotations and positions ($R^{\mathrm{h}}_{\boldsymbol{R}}$, $R^{\mathrm{h}}_{\boldsymbol{J}}$), and object orientation and position ($R^{\mathrm{o}}_{\boldsymbol{R}}$, $R^{\mathrm{o}}_{\boldsymbol{p}}$); and (ii) coarse geometric $R_{\boldsymbol{D}}$ and contact $R_{\boldsymbol{C}}$ correspondences. To discourage excessive actuation, we add energy terms $R_{\text{energy}}(\boldsymbol{a}_t)$, including a penalty for large changes in consecutive PD targets as well as high joint accelerations and velocities, thereby promoting smooth and energy-

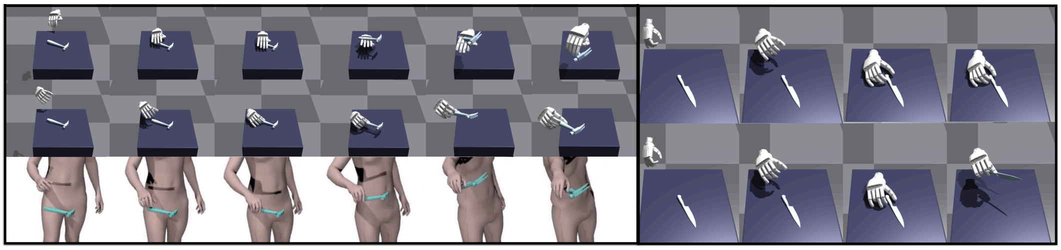

Figure 3: *Left*: Comparison with AnyTeleop [33] (**top**) and retargeted MoCap data (**bottom**), both without task awareness. For hands with limited DoFs, retargeting often yields unnatural poses, unlike our tracking results (**middle**). *Right*: Comparison with DexTrack [9] (**top**). For objects with small grasp regions, the baseline fails to grasp reliably, while our method (**bottom**) succeeds.

efficient motion. The reward formulation below is designed to support retargeting by emphasizing correspondence in behaviorally relevant features, enabling robust transfer across embodiments.

Kinematic State Matching. The primary objective is to align the robot hand with demonstration, emphasizing the matching of joints and links specified by the mapping $\boldsymbol{M}$, as discussed in state formulation. Formally, we define $R^{\mathrm{h}}_{\boldsymbol{J}} = \exp\Big(-\sum \lambda^{\mathrm{h}}_{\boldsymbol{J}} \big\|\boldsymbol{M}(\hat{\boldsymbol{p}}) - \boldsymbol{M}(\boldsymbol{p})\big\|^2\Big), R^{\mathrm{h}}_{\boldsymbol{R}} = \exp\Big(-\sum \lambda^{\mathrm{h}}_{\boldsymbol{R}} \big\|\boldsymbol{M}(\hat{\boldsymbol{R}}) \ominus \boldsymbol{M}(\boldsymbol{R})\big\|^2\Big)$, where $\lambda^{\mathrm{h}}_{\boldsymbol{J}}, \lambda^{\mathrm{h}}_{\boldsymbol{R}}$ are weighting hyperparameters.

Dynamic State Matching. The policy is also encouraged to maintain appropriate interaction dynamics, such as keeping fingertips near functional affordances and preserving contact correspondences. The associated rewards are $R_{\boldsymbol{D}} = \exp\big(-\lambda_{\boldsymbol{D}}\|\hat{\boldsymbol{D}} - \boldsymbol{D}\|^2\big), R_{\boldsymbol{C}} = \exp\big(-\lambda_{\boldsymbol{C}}\|\hat{\boldsymbol{C}} - \boldsymbol{C}\|^2\big)$. These terms promote consistent manipulation with embodiment gaps preventing exact kinematics mimic.

Early Termination. We employ early termination criteria to halt episodes when simulation significantly deviates from feasible or desired trajectories. Formally, termination conditions are defined as: **(I)**$R^{\mathrm{h}}_{\boldsymbol{J}} < \kappa^{\mathrm{h}}_{\boldsymbol{J}}$, **(II)**$R^{\mathrm{o}}_{\boldsymbol{p}} < \kappa^{\mathrm{o}}_{\boldsymbol{p}}$, **(III)**$R^{\mathrm{o}}_{\boldsymbol{R}} < \kappa^{\mathrm{o}}_{\boldsymbol{R}}$, **(IV)**$R_{\boldsymbol{D}} < \kappa_{\boldsymbol{D}}$, **(V)**$\boldsymbol{C}_t \neq \hat{\boldsymbol{C}}_t, \forall t \in [t_0, t_0 + 10]$, These criteria promptly terminate episodes when interactions deviate irrecoverably from reference trajectories, thus facilitating effective policy learning via collecting meaningful experiences [54].

State Initialization. Because of the anatomical and kinematic differences between the robot and the MANO hand model [55], directly assigning MANO poses as initial robot states is infeasible. At the start of training, we initialize only the global rotation and translation of the robot palm according to the reference pose, while setting all other degrees of freedom to zero. Following [51], we then cache high-quality rollout states and subsequently reuse them as initialization. To further improve training efficiency, we adopt prioritized sampling: rollouts are more likely to start from challenging phases rather than motions for hand approaching.

Learning with Reference Scope. Our key idea is to enable flexible trajectory exploration within a unified imitation learning framework that integrates retargeting, tracking, and correction. The central mechanism is to reduce reliance on strict tracking rewards and instead shape the policy through adaptive early termination criteria. (**I**) We introduce a reward weighting scheme in which the kinematic matching weights $\lambda^{\mathrm{h}}_{\boldsymbol{R}}$ and $\lambda^{\mathrm{h}}_{\boldsymbol{J}}$ are proportional to the hand–object surface distance, while the energy penalty $R_{\text{energy}}(\boldsymbol{a}_t)$ is inversely proportional to this distance. This design encourages smoother, lower-energy motions when tracking constraints are relaxed. For example, the kinematic weight is given by $w(\boldsymbol{D}) = \min\Big(1, \frac{\boldsymbol{D}}{0.20\,\mathrm{m}}\Big)$. Although these weights vary across timesteps, they are deterministic functions of the reference trajectory and therefore the reward still remains stationary from the policy's perspective, ensuring that the optimization target is consistent, and thus, the varying weights do not bias the policy toward unintended behaviors. (**II**) We shape the reward via adaptive early termination thresholds. For each termination criterion κ, we initialize with a large value and progressively tighten it according to the ratio of failed rollouts: $\kappa = \kappa^{\text{init}} \cdot \frac{N_{\text{fail}}}{N_{\text{total}}}$, where κ^{init} is the initial large threshold, N_{fail} is the number of unsuccessful rollouts (early termination events), and N_{total} is the total number of rollouts visited at the current frame.

Embodiment	Tracking w/ Retargeting (optional)	R_{err} (rad, ↓)	T_{err} (cm, ↓)	E_{finger} (cm, ↓)	Success Rate (%, ↑)
Inspire [1]	DexTrack [9] w/ AnyTeleop [33]	0.475 / 0.216	4.93 / 2.29	6.71 / 5.67	7.4
	DexTrack [9] w/ **Ours**	0.515 / 0.467	**3.75** / 3.26	6.43 / 6.28	69.8
	DEXPLORE (w/o RSE)	0.514 / 0.333	3.88 / 2.59	6.12 / 5.71	29.9
	DEXPLORE	**0.474** / 0.452	3.77 / 3.65	**6.06** / 6.04	**87.7**
Allegro [59]	DexTrack [9] w/ AnyTeleop [33]	0.595 / 0.471	4.21 / 3.39	9.38 / 9.36	45.9
	DexTrack [9] w/ **Ours**	0.560 / 0.512	4.79 / 4.32	8.00 / 7.91	77.4
	DEXPLORE (w/o RSE)	0.549 / 0.481	**4.19** / 3.68	**7.98** / 7.92	29.9
	DEXPLORE	**0.494** / 0.454	4.48 / 4.06	8.10 / 8.01	**78.7**

Table 1: **Quantitative evaluations** comparing our method with the baseline on the GRAB [60] dataset. Tracking errors are reported as averages computed either over successful rollouts / over all frames. Note that averaging errors across all frames generally yields lower values when the success rate is low, since failed sequences mainly involve easy approaching without manipulating the object.

3.2 Vision-based Generative Control

Task Formulation. The objective of generative control is to learn a policy $\tilde{\pi}$ that enables a robotic hand to manipulate objects using partial observations and optionally sparse goals.

Observation. As illustrated in Figure 2 II), the partial observations consist of (i) hand proprioception ($\tilde{\boldsymbol{x}}$), including the global wrist position and orientation as well as local joint rotations, and (ii) an object point cloud ($\boldsymbol{P}$) reconstructed from a single-view depth image. Depth images are captured by a camera randomly positioned on a hemisphere above the table, with intrinsic parameters matched to our Femto Bolt sensor [56]. Each depth map is converted into a 3D point cloud using the known camera model. To encode the visual input, we employ a PointNet++ [57] backbone that jointly processes the object point cloud together with observed human joint positions from the camera, yielding a compact representation of both object geometry and hand-object interaction context. Other proprioceptive signals, such as joint velocities and tactile information, are excluded due to their large sim-to-real gap. In addition, the observation includes short-horizon historical states and a binary indicator specifying whether the current timestep occurs before or after the onset of manipulation.

Goal. Following [58], we define sparse goals by selectively unmasking portions of the reference motion, while the remaining components are either masked during training or replaced by mask tokens at inference. For example, unmasking a desired wrist trajectory, obtained from reference data during training or generated by an RRT-based motion planner in real-world evaluation, serves as the goal condition. The policy then leverages its latent representation to infer the masked components and generate diverse, human-like manipulation behaviors.

Learning the Generative Control. Introducing a latent embedding is essential in order to (i) capture and modulate diverse behaviors, (ii) eliminate reliance on complete reference trajectories, and (iii) model the uncertainty that arises from missing privileged information at inference time. Our framework consists of an encoder $q_\phi(z \mid \boldsymbol{x}, \hat{\boldsymbol{x}})$, a prior network $p_\psi(z \mid \tilde{\boldsymbol{x}}, \boldsymbol{P})$, and a decoder policy $\tilde{\pi}_\theta(\tilde{\boldsymbol{a}} \mid z, \tilde{\boldsymbol{x}})$. The encoder takes both the privileged robot-object state $\boldsymbol{x}$ and reference $\hat{\boldsymbol{x}}$, and outputs a Gaussian latent distribution $\mathcal{N}(\boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q)$. In parallel, the prior network produces a *state-dependent* Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_p, \boldsymbol{\Sigma}_p)$. At each timestep, the latent skill embedding is sampled as $z = \boldsymbol{\mu}_p + \boldsymbol{\mu}_q + \boldsymbol{\Sigma}_q^{1/2}\epsilon, \epsilon \sim \mathcal{N}(0, I)$, with the noise ϵ held fixed within an episode to ensure temporal consistency of the executed skill, following [58, 61]. The framework is trained via online imitation learning using DAgger, with the imitation control policy introduced in Sec. 3.1 serving as the expert teacher. It minimizes the reconstruction loss $\mathcal{L}_{\text{rec}} = \|\boldsymbol{a} - \tilde{\boldsymbol{a}}\|^2$, where $\tilde{\boldsymbol{a}}$ are predicted actions and $\boldsymbol{a}$ are expert-provided reference actions. A KL regularization term further encourages the encoder distribution to remain close to the prior: $\mathcal{L}_{\text{KL}} = D_{\text{KL}}\big(\mathcal{N}(\boldsymbol{\mu}_p + \boldsymbol{\mu}_q, \boldsymbol{\Sigma}_q) \parallel \mathcal{N}(\boldsymbol{\mu}_p, \boldsymbol{\Sigma}_p)\big).$, ensuring that the encoder does not capture information unavailable at inference. The overall training objective is $\mathcal{L} = \mathcal{L}_{\text{rec}} + \beta\, \mathcal{L}_{\text{KL}}$, with β gradually increased during training to promote a more structured latent space, following [62]. At inference time, the encoder is omitted, and latents are sampled directly from the learned prior: $z \sim \mathcal{N}(\boldsymbol{\mu}_p, \boldsymbol{\Sigma}_p)$, yielding a generative policy $\tilde{\pi}$ that produces goal-conditioned dexterous manipulation behaviors using only partial observations.

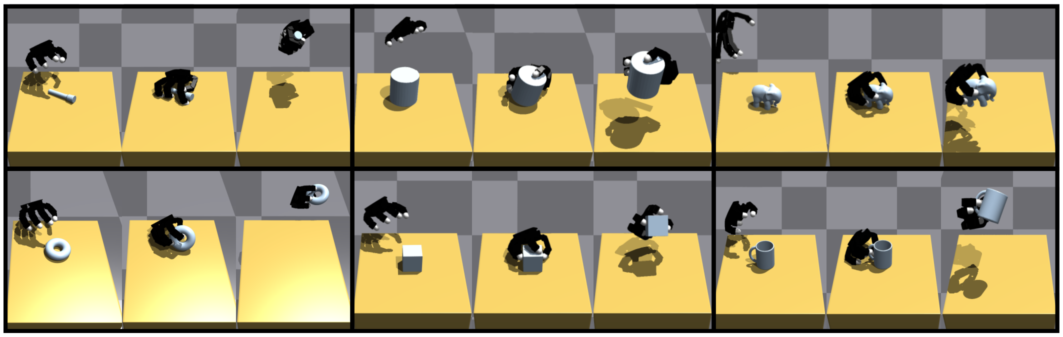

Figure 4: Our imitation control framework is also suitable for Allegro hands [59]. Compared to Inspire hands [1], Allegro hands feature a different morphology with four fingers instead of five, and offer more degrees of freedom (DoFs), but are significantly larger in size.

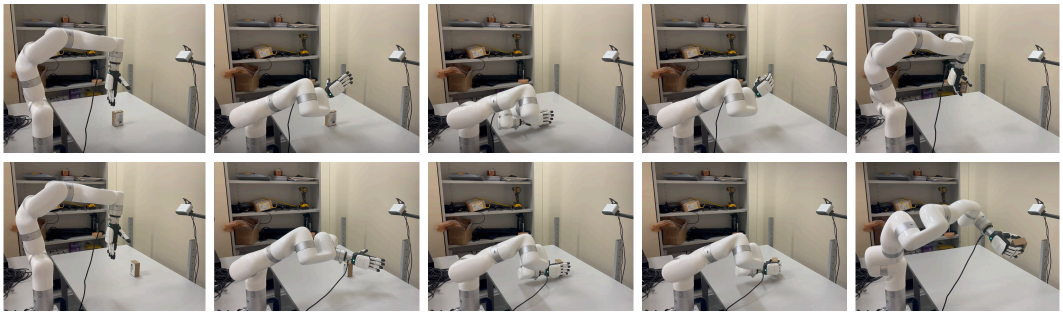

Figure 5: The vision-based policy deployed on an XArm-7 robotic arm [65] equipped with an Inspire dexterous hand [1] and a Fremto Bolt depth camera [56].

4 Experiments

We evaluate the state-based policy's ability to imitate human demonstrations and generalize to unseen scenarios. We then assess the vision-based policy framework's effectiveness in achieving manipulation within simulation and its successful transfer to real-world deployment.

Data and Robot Models. We evaluate our state-based policy using two robotic hands in simulation: (**I**) the Inspire hand [1], which is less dexterous than a human hand, featuring 12 degrees of freedom (DoFs) with only 6 actuated. We simulate mimic joint effects in IsaacGym [63], as described in Sec. 3.1; and (**II**) the Allegro hand [59], a fully actuated robotic hand with 16 DoFs, notable for its larger size compared to a human hand. These two hands comprehensively test our algorithm's robustness across embodiment gaps in both actuation complexity and physical dimensions. We primarily utilize the GRAB [60] dataset as our source for human demonstrations, selecting 658 sequences involving 51 objects from an original set of 1,269 sequences, ensuring that selected sequences focus on single-hand (primarily right-hand) manipulations without bimanual interactions. Our framework does not require explicit retargeting; however, we evaluate retargeting algorithms separately for comparison purposes. We further investigate the generalizability of our trained policy using novel data from the TACO dataset [64], evaluating whether our model benefits from scalability and effectively adapts to diverse object geometries.

Metrics. For evaluating the state-based policy, we introduce four metrics to assess tracking accuracy and task success. The *Object Rotation Error* (R_{err}) measures how closely the tracked object's orientation aligns with the reference orientation per frame. The *Object Translation Error* (T_{err}) captures the positional discrepancy between the tracked and reference object positions on a per-frame basis. Additionally, the *Fingertip Error* (E_{finger}) quantifies the average positional difference of fingertips between tracked and reference poses at each frame. Finally, the *Success Rate* is defined as the proportion of tracking attempts completed without dropping the object or exhibiting large deviations as indicated by the aforementioned metrics. To evaluate the vision-based policy, since explicit reference trajectories are unavailable, we rely on the *Success Rate* and *Contact Ratio* metrics. Here, success indicates maintaining stable hand-object contact over a duration and returning the object to the table, while the contact ratio measures the proportion of time the hand remains contact.

Baselines. We consider two types of baselines: (**I**) AnyTeleop [33]: a retargeting algorithm for robot hands from human demonstrations. We compare our policy's simultaneous retargeting and tracking capabilities to the kinematic replay from AnyTeleop's retargeting. (**II**) DexTrack [9]: We adapt it for comparison with our imitation control policy. Since DexTrack's primary contributions, including homotopy optimization and the combination of imitation learning (IL) and reinforcement learning (RL), have not been publicly released, we adapt the available components and train them in alignment with our experimental setting. Given DexTrack requires pre-processed retargeting, we also provide the high-quality imitation results from our DEXPLORE as the source of their training.

Quantitative Evaluation. Table 1 shows that our method consistently outperforms baseline methods on each robot hand. Our DEXPLORE achieves notably higher success rates. The ablation study on RSE demonstrates the effectiveness of scope exploration in compensating for robot-specific strategies. Additionally, since our policy is designed with retargeting in mind, we observe substantial performance improvements in the baseline DexTrack when utilizing rollouts from our policy as a reference. Table B evaluates our vision-based policy. Even with randomly sampled camera views, which might lead to severe hand-object occlusion, without privileged information, the policy still demonstrates strong performance in object manipulation and maintaining contact.

Qualitative Evaluation. We compare our DEXPLORE against the DexTrack baseline [9] in Figure 3, with the example where the affordance region is particularly constrained, requiring delicate and precise movements to grasp a knife. Our approach successfully accomplishes this challenging task. In Figure 3, we also highlight that task-unaware retargeting [33], which optimize fingertip distances as key features, is highly sensitive when the robot hand has fewer DoFs compared to a human hand, since certain finger flexions easily achievable by humans are infeasible for robot hands, resulting in unrecoverable retargeting failures. In contrast, our DEXPLORE effectively discovers solutions that achieve natural manipulation motions, albeit slightly deviating from the original trajectories. Furthermore, our algorithm remains effective even with increased DoFs, as demonstrated by the Allegro hand example in Figure 4. Our imitation control policy, enhanced by inherently encoded geometric information and guided exploration allowing deviation from the reference, demonstrates strong generalization capabilities to settings unseen during training – such as manipulating larger and heavier objects, as illustrated in Figure B. For the distilled vision-based generative control, we evaluate it under conditions where object geometries are entirely unseen during training. As demonstrated in Figure C, our policy successfully generalizes to manipulate these novel geometries. Additionally, skills from human demonstrations are distilled into a rich latent space learned from the MoCap data. By sampling from this latent space, the policy generates diverse and plausible manipulation styles, as illustrated in Figure D. We further validate this policy for real-world deployment, with successful examples shown in Figures 5 and A.

5 Conclusion

We introduce DEXPLORE, a novel paradigm for scalable dexterous robotic manipulation that unifies retargeting and tracking through learning from human demonstrations. Rather than rigidly replicating trajectories, our approach leverages flexible guidance, enabling robots to discover natural and efficient manipulation strategies adapted to their morphology. The learned state-based imitation policy is distilled into a vision-based generative control policy, removing reliance on privileged object observations and dense human references, thereby facilitating sim-to-real transfer. Experiments show that our policy outperforms baselines and that the distilled visual policies generalize effectively and are deployable in real-world scenarios.

6 Limitation

Our approach currently faces several limitations. First, it sometimes struggles with reliably manipulating very small or thin objects, where slight inaccuracies in fingertip positioning significantly affect success. Second, we focus exclusively on single-hand interactions; while extension to multi-hand manipulation is straightforward, coordinated behaviors require further exploration. Additionally, although our method directly learns from MoCap data for scalability, our training dataset is relatively modest, limiting generalization. Future work includes training on larger, more diverse datasets, and extending our approach to coordinated multi-hand manipulation scenarios.

Acknowledgments. We thank the Toyota Research Institute for the partial support of the robotic hardware used in this research. S. Xu and Y.-X. Wang are supported in part by NSF Grant 2106825 and NIFA Award 2020-67021-32799. This work used computational resources, including the NCSA Delta and DeltaAI and the PTI Jetstream2 supercomputers through allocations CIS230012, CIS230013, and CIS240311 from the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS) program, as well as the TACC Frontera supercomputer and Amazon Web Services (AWS) through the National Artificial Intelligence Research Resource (NAIRR) Pilot.

References

[1] Inspire-robots. Smaller and higher-precision motion control experts. https://inspire-robots.store/.

[2] Z. Jiang, Y. Xie, K. Lin, Z. Xu, W. Wan, A. Mandlekar, L. Fan, and Y. Zhu. Dexmimicgen: Automated data generation for bimanual dexterous manipulation via imitation learning. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, 2025.

[3] H. Jiang, Y. Wang, H. Zhou, and D. Seita. Learning to Singulate Objects in Packed Environments using a Dexterous Hand. In *International Symposium on Robotics Research (ISRR)*, 2024.

[4] R. Bhirangi, A. DeFranco, J. Adkins, C. Majidi, A. Gupta, T. Hellebrekers, and V. Kumar. All the feels: A dexterous hand with large-area tactile sensing. *IEEE Robotics and Automation Letters*, 8(12):8311–8318, 2023.

[5] A. Billard and D. Kragic. Trends and challenges in robot manipulation. *Science*, 364(6446): eaat8414, 2019.

[6] O. Kroemer, S. Niekum, and G. Konidaris. A review of robot learning for manipulation: Challenges, representations, and algorithms. *Journal of machine learning research*, 22(30): 1–82, 2021.

[7] 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.

[8] K. Li, P. Li, T. Liu, Y. Li, and S. Huang. ManipTrans: Efficient dexterous bimanual manipulation transfer via residual learning. *arXiv preprint arXiv:2503.21860*, 2025.

[9] X. Liu, J. Adalibieke, Q. Han, Y. Qin, and L. Yi. DexTrack: Towards generalizable neural tracking control for dexterous manipulation from human references. In *ICLR*, 2025.

[10] A. Mandlekar, Y. Zhu, A. Garg, J. Booher, M. Spero, A. Tung, J. Gao, J. Emmons, A. Gupta, E. Orbay, et al. Roboturk: A crowdsourcing platform for robotic skill learning through imitation. In *CoRL*, 2018.

[11] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. *arXiv preprint arXiv:2108.03298*, 2021.

[12] S. P. Arunachalam, S. Silwal, B. Evans, and L. Pinto. Dexterous imitation made easy: A learning-based framework for efficient dexterous manipulation. In *ICRA*, 2023.

[13] 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 arXiv:2209.14284*, 2022.

[14] B. Zhou, H. Yuan, Y. Fu, and Z. Lu. Learning diverse bimanual dexterous manipulation skills from human demonstrations. *arXiv preprint arXiv:2410.02477*, 2024.

[15] S. Li, Z. Huang, T. Chen, T. Du, H. Su, J. B. Tenenbaum, and C. Gan. Dexdeform: Dexterous deformable object manipulation with human demonstrations and differentiable physics. In *ICLR*, 2023.

[16] 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 *CoRL*, 2023.

[17] S. An, Z. Meng, C. Tang, Y. Zhou, T. Liu, F. Ding, S. Zhang, Y. Mu, R. Song, W. Zhang, et al. Dexterous manipulation through imitation learning: A survey. *arXiv preprint arXiv:2504.03515*, 2025.

[18] J. Ye, K. Wang, C. Yuan, R. Yang, Y. Li, J. Zhu, Y. Qin, X. Zou, and X. Wang. Dex1b: Learning with 1b demonstrations for dexterous manipulation. In *RSS*, 2025.

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

[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*, 7(4): 10873–10881, 2022.

[21] C. Chi, B. Tjaden, S. Xu, et al. Diffusion policy: Visuomotor policy learning via action diffusion. In *Robotics: Science and Systems (RSS)*, 2023.

[22] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *RSS*, 2023.

[23] H. Luo, Y. Feng, W. Zhang, S. Zheng, Y. Wang, H. Yuan, J. Liu, C. Xu, Q. Jin, and Z. Lu. Being-h0: Vision-language-action pretraining from large-scale human videos. *arXiv preprint arXiv:2507.15597*, 2025.

[24] R. Yang, Q. Yu, Y. Wu, R. Yan, B. Li, A.-C. Cheng, X. Zou, Y. Fang, H. Yin, S. Liu, et al. Egovla: Learning vision-language-action models from egocentric human videos. *arXiv preprint arXiv:2507.12440*, 2025.

[25] Y. Qin, Y.-H. Wu, S. Liu, H. Jiang, R. Yang, Y. Fu, and X. Wang. Dexmv: Imitation learning for dexterous manipulation from human videos. In *ECCV*, 2022.

[26] 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 *ICRA*, 2020.

[27] S. Chen, C. Wang, K. Nguyen, L. Fei-Fei, and C. K. Liu. Arcap: Collecting high-quality human demonstrations for robot learning with augmented reality feedback. *arXiv preprint arXiv:2410.08464*, 2024.

[28] X. Cheng, J. Li, S. Yang, G. Yang, and X. Wang. Open-television: Teleoperation with immersive active visual feedback. In *CoRL*, 2024.

[29] R. Ding, Y. Qin, J. Zhu, C. Jia, S. Yang, R. Yang, X. Qi, and X. Wang. Bunnyvisionpro: Real-time bimanual dexterous teleoperation for imitation learning. *arXiv preprint arXiv:2407.03162*, 2024.

[30] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *RSS*, 2023.

[31] P. Wu, Y. Shentu, Z. Yi, X. Lin, and P. Abbeel. Gello: A general, low-cost, and intuitive teleoperation framework for robot manipulators. In *IROS*, 2024.

[32] T. He, Z. Luo, W. Xiao, C. Zhang, K. Kitani, C. Liu, and G. Shi. Learning human-to-humanoid real-time whole-body teleoperation. In *IROS*, 2024.

[33] 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 *RSS*, 2023.

[34] A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov, and S. Levine. Learning complex dexterous manipulation with deep reinforcement learning and demonstrations. In *RSS*, 2018.

[35] A. Nair, B. McGrew, M. Andrychowicz, W. Zaremba, and P. Abbeel. Overcoming exploration in reinforcement learning with demonstrations. In *ICRA*, 2018.

[36] M. Vecerik, T. Hester, J. Scholz, F. Wang, O. Pietquin, B. Piot, N. Heess, T. Rothörl, T. Lampe, and M. Riedmiller. Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards. *arXiv preprint arXiv:1707.08817*, 2017.

[37] I. Akkaya, M. Andrychowicz, M. Chociej, and et al. Solving rubik's cube with a robot hand. *arXiv:1910.07113*, 2019.

[38] M. Andrychowicz, F. Wolski, A. Ray, J. Schneider, R. Fong, P. Welinder, B. McGrew, J. Tobin, O. Pieter Abbeel, and W. Zaremba. Hindsight experience replay. In *NeurIPS*, 2017.

[39] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *IROS*, 2017.

[40] S. Dasari, A. Gupta, and V. Kumar. Learning dexterous manipulation from exemplar object trajectories and pre-grasps. In *ICRA*, 2023.

[41] T. Wu, M. Wu, J. Zhang, Y. Gan, and H. Dong. Learning score-based grasping primitive for human-assisting dexterous grasping. In *NeurIPS*, 2023.

[42] H. Zhu, A. Gupta, A. Rajeswaran, S. Levine, and V. Kumar. Dexterous manipulation with deep reinforcement learning: Efficient, general, and low-cost. In *ICRA*, 2019.

[43] G. Khandate, S. Shang, E. T. Chang, T. L. Saidi, Y. Liu, S. M. Dennis, J. Adams, and M. Ciocarlie. Sampling-based exploration for reinforcement learning of dexterous manipulation. *arXiv preprint arXiv:2303.03486*, 2023.

[44] T. G. W. Lum, M. Matak, V. Makoviychuk, A. Handa, A. Allshire, T. Hermans, N. D. Ratliff, and K. Van Wyk. Dextrah-g: Pixels-to-action dexterous arm-hand grasping with geometric fabrics. *arXiv preprint arXiv:2407.02274*, 2024.

[45] R. Singh, A. Allshire, A. Handa, N. Ratliff, and K. Van Wyk. Dextrah-rgb: Visuomotor policies to grasp anything with dexterous hands. *arXiv preprint arXiv:2412.01791*, 2024.

[46] W. Wan, H. Geng, Y. Liu, Z. Shan, Y. Yang, L. Yi, and H. Wang. Unidexgrasp++: Improving dexterous grasping policy learning via geometry-aware curriculum and iterative generalist-specialist learning. In *ICCV*, 2023.

[47] Y. Xu, W. Wan, J. Zhang, H. Liu, Z. Shan, H. Shen, R. Wang, H. Geng, Y. Weng, J. Chen, et al. Unidexgrasp: Universal robotic dexterous grasping via learning diverse proposal generation and goal-conditioned policy. In *CVPR*, 2023.

[48] H. Zhang, Z. Wu, L. Huang, S. Christen, and J. Song. Robustdexgrasp: Robust dexterous grasping of general objects. *arXiv preprint arXiv:2504.05287*, 2025.

[49] L. Huang, H. Zhang, Z. Wu, S. Christen, and J. Song. Fungrasp: functional grasping for diverse dexterous hands. *IEEE Robotics and Automation Letters*, 2025.

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

[51] S. Xu, H. Y. Ling, Y.-X. Wang, and L.-Y. Gui. Intermimic: Towards universal whole-body control for physics-based human-object interactions. In *CVPR*, 2025.

[52] S. Christen, M. Kocabas, E. Aksan, J. Hwangbo, J. Song, and O. Hilliges. D-grasp: Physically plausible dynamic grasp synthesis for hand-object interactions. In *CVPR*, 2022.

[53] R. S. Dahiya, G. Metta, M. Valle, and G. Sandini. Tactile sensing—from humans to humanoids. *IEEE transactions on robotics*, 26(1):1–20, 2009.

[54] 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 (TOG)*, 37(4):1–14, 2018.

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

[56] ORBBEC. Orbbec femto bolt high-performance depth camera. https://www.orbbec.com/products/tof-camera/femto-bolt/.

[57] C. R. Qi, L. Yi, H. Su, and L. J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*, 2017.

[58] C. Tessler, Y. Guo, O. Nabati, G. Chechik, and X. B. Peng. Maskedmimic: Unified physics-based character control through masked motion inpainting. *ACM Transactions on Graphics (TOG)*, 43(6):1–21, 2024.

[59] Wonic-robotics. Allegro hand. https://www.allegrohand.com.

[60] O. Taheri, N. Ghorbani, M. J. Black, and D. Tzionas. GRAB: A dataset of whole-body human grasping of objects. In *ECCV*, 2020.

[61] H. Yao, Z. Song, B. Chen, and L. Liu. Controlvae: Model-based learning of generative controllers for physics-based characters. *ACM Transactions on Graphics (TOG)*, 41(6):1–16, 2022.

[62] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner. beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR*, 2017.

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

[64] Y. Liu, H. Yang, X. Si, L. Liu, Z. Li, Y. Zhang, Y. Liu, and L. Yi. Taco: Benchmarking generalizable bimanual tool-action-object understanding. In *CVPR*, 2024.

[65] UFACTORY. Ufactory x-arm 7. https://www.ufactory.us/product/ufactory-xarm-7.

DEXPLORE: Scalable Neural Control for Dexterous Manipulation from Reference-Scoped Exploration

Appendix

In this appendix, we provide additional experimental setups:

1. **Demo Video.** A demonstration video including the real world experiment is provided, as described in Sec. A.
2. **Simulation Setup.** The environment configuration for simulation is introduced in Sec. B.
3. **Real-world Setup.** Sec. C presents further details on our real-world deployment.

A Demo Video

In addition to the qualitative results presented in the main paper, we provide a demo video (`https://sirui-xu.github.io/dexplore/demo.mp4`) for more detailed visualizations of the tasks, further illustrating the efficacy of our approach. The demo video conveys the following key points:

1. We demonstrate that our vision-based policy successfully transfers to real-world dexterous grasping tasks.
2. Our state-based policy consistently outperforms baseline methods, such as DexTrack [9] and AnyTeleop [33], demonstrating superior task execution and generalization capabilities, such as to novel objects.
3. Our holistic method integrates retargeting and skill learning into a unified, cohesive framework, enhancing overall adaptability and performance.
4. Our framework exhibits significant versatility, successfully adapting to different robotic embodiments, including the Allegro [59] hand larger than hand hands, and Inspire [1] hand with unactuated joints.
5. Our vision-based policy, which operates without privileged information, reliably accomplishes tasks with high robustness, emphasizing the practicality and resilience of our approach in real-world scenarios.

B Simulation Setup

We use MANO models [55] to represent human hand reference data, and objects are represented using convex decomposition into 20 convex hulls for efficient simulation. Key simulation parameters are summarized concisely in Table A.

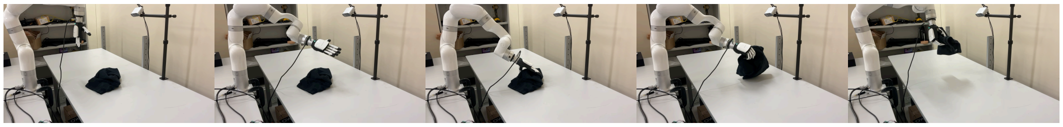

Figure A: our framework can successfully grasp deformable objects (*i.e.*, cloth).

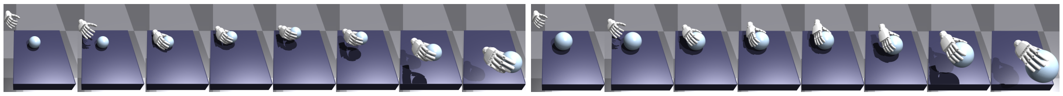

Figure B: Our imitation control policy generalizes effectively to manipulating larger and heavier objects unseen during training. Specifically, we scale the object's size by a factor of 1.5, resulting in a corresponding weight increase of 1.5^3 shown on the right.

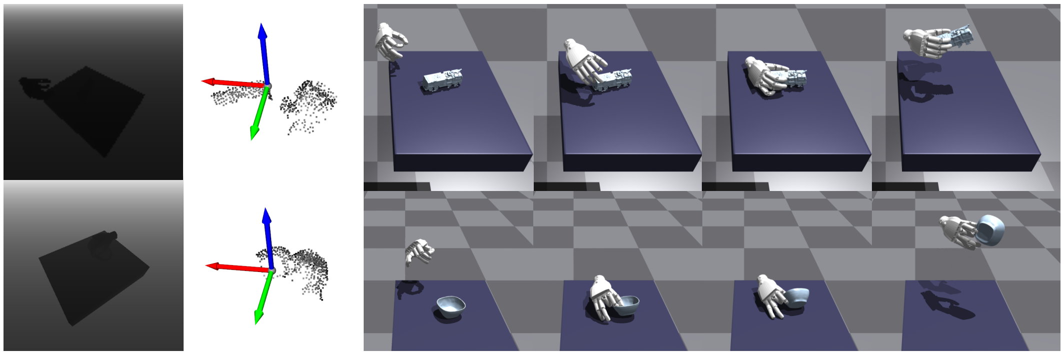

Figure C: Our vision-based generative control, though trained exclusively on the GRAB dataset [60], generalizes to manipulating novel objects, specifically, object 081 and object 194 from the TACO dataset [64].

C Real-World Deployment

Hardware Setup. As illustrated in Figure 1 of the main paper, our experimental setup includes an Xarm-7 robotic arm [65] coupled with an Inspire RH56DFTP dexterous right hand [1]. The Xarm-7 has 7 independently motor-driven joints, while the Inspire RH56DFTP hand comprises 12 joints actuated by 6 motors, meaning 6 joints serve as mimic joints. In simulation, the proportional-derivative (PD) targets for these mimic joints are proportionally scaled based on the targets of their actuated counterparts, as detailed in Sec. 3.1 of the main paper. In our real-world experiments, we directly utilize the PD targets from the actuated joints for robot control. The real robot employs internally defined position control parameters for proportional gain and derivative gain, whereas in simulation, we explicitly set these values to 100 and 10, respectively. Additionally, we utilize a Femto Bolt depth camera [56] to perform depth estimation and extract point cloud data.

Deployment. To achieve smooth and efficient robot motion during deployment, we do not directly apply every step of the 30 fps rollout during the approach and lifting stages. Instead, the full rollout is

Parameter	Value
Sim timestep	$1/60$ s
Control timestep	$1/30$ s
Environment count	4096
Physics substeps	8
Position solver iterations	8
Velocity solver iterations	1
Contact offset	0.02
Rest offset	0.0
Max depenetration vel.	10
Object restitution	0.7
Object friction	0.9
Object density	50

Table A: Summary of simulation parameters in Isaac Gym [63].

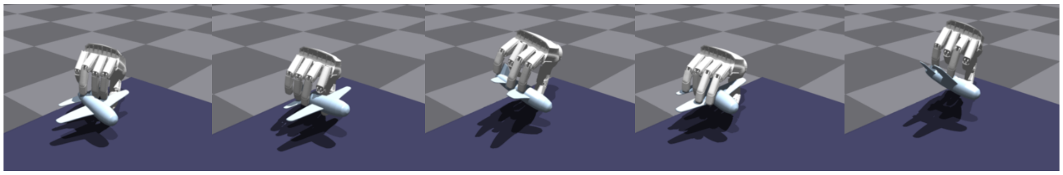

Figure D: Our vision-based generative control formulates a latent space for sampling, resulting in diverse manipulation styles. We visualize poses at the moment when the hand makes initial contact with the object.

activated only when the hand and object are sufficiently close. Notably, we initiate the thumb motion first, followed by the motion of the other four fingers. This sequence is adopted because the thumb, equipped with two actuators and having greater weight, moves more slowly compared to the other fingers. The wrist trajectory is generated via motion planning using the Rapidly-exploring Random Tree (RRT) algorithm, ensuring feasible and collision-free paths for robotic motion. Specifically, RRT planner moves the hand near the object and to return it afterward. Once the hand is fully within the camera's view, manipulation is entirely driven by our policy. No additional hand-crafted control is used during the core grasp.

Compute cost. At inference time, the policy runs on an RTX 4090 paired with an Intel i9-14900K, with a total latency of approximately 80 ms per step. Of this, around 60–80 ms is spent on descriptor computation, which includes acquiring a depth image, masking out the background and robot arm via forward kinematics.

Method	**Success Rate (%, ↑)**	**Contact Ratio (%, ↑)**
DEXPLORE (vision)	52.7	48.2
DEXPLORE (state)	87.7	69.3

Table B: Comparison of our vision-based policy with the state-based policy shows that it performs well in manipulating objects with only sparse reference and partial observations.