

FlashDexRetarget: Accelerating Dexterous Manipulation Data Generation through Multi-Motion Retargeting

Kyungmin Lee^{1*}, Sibeon Kim^{1*}, Dongyoon Hwang^{1*}, Yoonsang Oh^{1*}, Donghu Kim², Youngdo Lee²,
I Made Aswin Nahrendra², Jaegul Choo^{1†}, Hojoon Lee^{2†}

¹KAIST AI ²Holiday Robotics

[holiday-robot.github.io/FlashDexRetarget](https://github.com/holiday-robot/FlashDexRetarget)

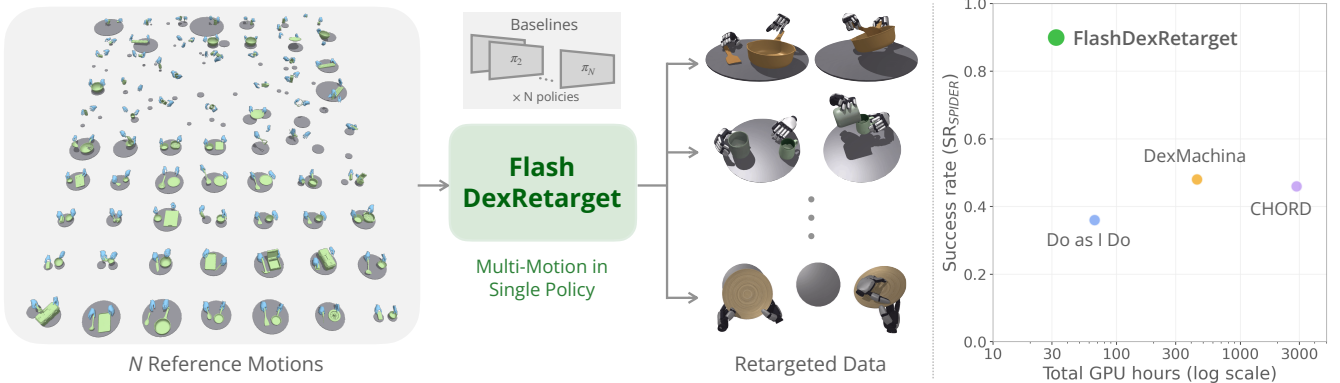


Fig. 1. **FlashDexRetarget at a glance.** FlashDexRetarget converts collections of human hand-object demonstrations into physically grounded robot trajectories through multi-reference RL-based retargeting. Left- and right-hand policies are jointly trained across reference motions, avoiding repeated optimization for individual demonstrations, and successful rollouts are stored as robot data. The right plot shows cumulative conversion success rate versus total GPU-hours on a logarithmic scale. CHORD results are obtained with its released implementation, which trains policies with PPO [1].

Abstract—Human hand-object demonstrations provide a scalable source of data for dexterous robot learning, but transferring them across embodiments requires physically feasible retargeting. Existing physics-based methods typically optimize each demonstration independently, leading to either limited success under finite simulation budgets or training costs that grow with dataset size. We introduce FlashDexRetarget, an RL framework for multi-reference dexterous retargeting. We formulate retargeting as multi-reference tracking, jointly learning a single policy across many demonstrations with off-policy RL and geometric supervision of the demonstrated interactions. This shared training formulation amortizes optimization across references while enabling the policy to track diverse hand-object interactions. On a 50-motion benchmark from TACO, OakInk2, and HOT3D using XHand and Sharpa Wave Hand as target embodiments, FlashDexRetarget retargets 90% of demonstrations using about 30 GPU-hours, compared with about 46% at about 3,000 GPU-hours for CHORD. This corresponds to about 100 times lower training compute and a 44-percentage-point improvement in retargeting success. Ablations examine the key design choices, while experiments with up to 1,000 motions and real-world replay further demonstrate the scalability and practical applicability of our method.

I. INTRODUCTION

Learning dexterous manipulation requires large amounts of robot interaction data. Human hand-object demonstrations [2–4] provide a promising source of such data, as diverse manipulation behaviors can be collected at scale

without directly operating a robot. To use these demonstrations for robot learning, however, human motions must be translated into actions executable by a target robot hand. Physics-based retargeting addresses this issue by adapting human demonstrations to the morphology and actuation of the robot in simulation, while leveraging physical contact to reproduce the demonstrated object motion. To make large human motion datasets practical sources of robot data, retargeting must achieve high success at low compute cost.

Existing physics-based retargeting methods adopt a single-reference formulation, optimizing each demonstration independently. Sampling-based methods [5, 6] search directly over action sequences through simulated rollouts, but high-dimensional actions and long contact sequences make this search challenging under a finite simulation budget. RL-based methods [7–9] instead learn closed-loop policies to track the demonstrated hand-object interaction, but train a separate policy for each reference. Despite their different optimization strategies, both approaches retarget each demonstration independently, causing computational cost to grow linearly with the number of demonstrations.

To avoid this repeated optimization, we study *multi-reference tracking*, where a single reference-conditioned policy is trained jointly across many demonstrations. By sharing learning across references, this formulation has the potential to amortize training cost over an entire dataset while retaining accurate tracking. Multi-reference tracking

*Equal contribution. †Corresponding author.

has been effective for humanoid whole-body motion [10], but dexterous manipulation introduces additional challenges.

First, object shapes and hand-object configurations vary across demonstrations, requiring the policy to distinguish interaction geometries. Second, reconstructed human demonstrations, even when captured using motion-capture equipment, contain contact errors, making exact contact matching unreliable. Third, multi-reference training broadens the state distribution, increasing the need for model capacity and effective experience reuse. Finally, for bimanual motions, the two hands assume distinct roles, such as primary and supporting roles, requiring asymmetric learning signals.

To address these challenges, we introduce **FlashDexRetarget**, an RL framework for multi-reference dexterous retargeting. First, we condition the policy on object geometry and K future reference frames to distinguish diverse interactions across references. Second, we replace exact contact matching with distance-based interaction rewards to provide dense hand-object supervision despite imperfect and sparse reconstructed contact information. Third, we adopt the off-policy learner FlashSAC [11] and increase both the replay buffer size and critic capacity to support learning across a broader multi-reference distribution. Finally, we use separate left- and right-hand actor-critic networks to handle the distinct roles and learning signals of the two hands.

We evaluate FlashDexRetarget on 50 hand-object motions from TACO, OakInk2, and HOT3D [2–4] using XHand and Sharpa Wave Hand as target embodiments. Against sampling-based and RL-based baselines, we evaluate retargeting success, measured as the fraction of successfully retargeted demonstrations, together with the total training compute required. A single FlashDexRetarget policy retargets 90% of the benchmark using about 30 GPU-hours, compared with about 46% success at about 3,000 GPU-hours for CHORD [7]. This corresponds to approximately $100\times$ lower training compute and a 44-percentage-point improvement in retargeting success. To understand these gains, we ablate how conditioning on K future reference frames, hand-object interaction objectives, off-policy training and capacity scaling, and separate left- and right-hand networks contribute to performance in the multi-reference setting. We further show that the number of successfully retargeted trajectories continues to grow as the dataset scales. Finally, we validate the retargeted trajectories through real-world replay.

Our main contributions are:

- (i) We introduce FlashDexRetarget, an RL framework that jointly retargets human hand-object demonstrations through multi-reference tracking with a single policy.
- (ii) We combine object point-cloud observations, hand-object interaction objectives, off-policy learning with larger replay and model capacities, and left- and right-hand actor-critic networks for joint dexterous tracking.
- (iii) We evaluate on 50 motions and two robot embodiments, achieving approximately $100\times$ lower training compute and 44 percentage points higher success than CHORD, with ablations and real-world replay.

II. RELATED WORK

A. Human Hand-Object Interaction Data

Human hand-object datasets cover diverse objects and manipulation behaviors. DexYCB provides annotated hand-object poses [12], while GRAB and OakInk capture grasping configurations and interaction geometry [13, 14]. HOI4D and HOT3D extend this coverage to egocentric interactions [4, 15], and TACO, OakInk2, and ARCTIC capture coordinated bimanual manipulation [2, 3, 16]. GigaHands offers large-scale recordings of activities such as pouring, assembling objects, and playing cards [17]. These datasets provide diverse manipulation experience for robot learning.

However, these recordings capture human hands, which a robot cannot execute directly. Kinematic retargeting is a common first step to close this embodiment gap. UniDex [18] converts human hand-object trajectories into robot trajectories by solving fingertip inverse kinematics and manually adjusting the hand base so that the retargeted fingers still appear to touch the object. VideoManip [19] retargets human hand-object trajectories through keypoint matching and repairs the resulting contacts with contact optimization. In both, contact is imposed geometrically after the fact rather than produced by the robot hand acting on the object, so it is not guaranteed that executing the trajectory reproduces the demonstrated object motion. Retargeting must therefore be physics-based, producing trajectories with verified contacts.

B. Dexterous Retargeting from Human Demonstrations

Kinematic and planning-based pipelines adapt human demonstrations to robot motions [20–24], while physics-based optimization additionally accounts for hand-object contact dynamics in simulation [25, 26]. Within this physics-based setting, sampling-based methods search over simulated control rollouts [27–29] to support dexterous planning [30] and demonstration refinement [31]. For demonstration retargeting, SPIDER [6] introduces temporary virtual forces between robot fingers and the object to guide annealed sampling toward trajectories that reproduce the demonstrated contacts. Building on this approach, Do as I Do [5] incorporates hand-object reconstruction and physics-based refinement to retarget video demonstrations. Although these methods avoid training a tracking policy, they require a separate trajectory search for each reference, with effectiveness dependent on initialization and reference quality.

Rather than searching trajectories directly, RL-based methods learn dexterous manipulation policies from human motion references and task objectives [7–9, 32–37]. ManipTrans [8] learns residual corrections to a pretrained hand-motion imitator, while DexMachina [9] progressively reduces object assistance during training. CHORD [7] instead uses contact-wrench guidance to align human and robot object-motion capabilities. Although these methods replace trajectory search with policy learning, they still require separate training or adaptation for each demonstration. In contrast, FlashDexRetarget amortizes this cost by jointly learning hand-object tracking across demonstrations.

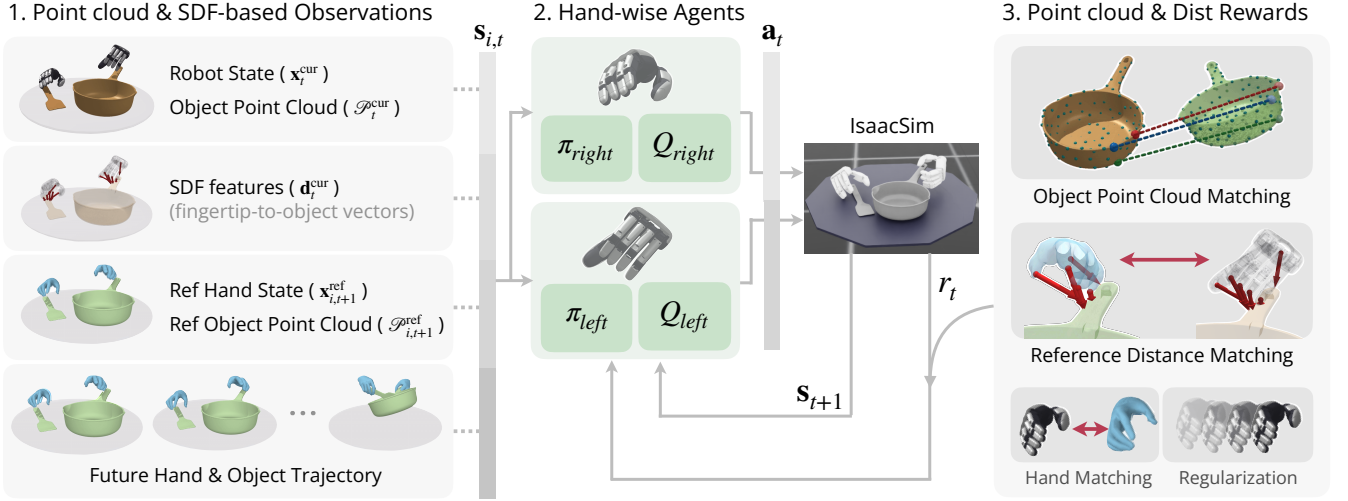


Fig. 2. **Overview of FlashDexRetarget.** (1) Observations combine robot states, current and reference object point clouds, local signed-distance features, and an encoding of future reference motion. (2) Left- and right-hand actor-critic pairs are trained with FlashSAC and jointly control the simulated interaction, with each pair sharing its parameters across all reference motions. (3) Object surface alignment and reference-conditioned fingertip proximity provide geometric learning signals, complemented by hand tracking and regularization. Together, these components support learning across diverse interactions.

III. METHOD

Our goal is to efficiently generate a large number of physically grounded robot trajectories from an extensive collection of human hand-object demonstrations.

Given a dataset of reference hand-object trajectories,

$$\mathcal{D} = \{\tau_i^{\text{ref}}\}_{i=1}^N, \quad \tau_i^{\text{ref}} = \{s_{i,t}^{\text{ref}}\}_{t=1}^{H_i}, \quad (1)$$

where i indexes the reference demonstration, $s_{i,t}^{\text{ref}}$ contains the recorded human hand states and object poses at time t , and H_i denotes the trajectory length, the goal is to reproduce the demonstrated interaction on a target robot.

Unlike single-reference retargeting, where a separate policy is optimized for each demonstration, we jointly train a reference-conditioned policy over the entire collection $\mathcal{D}$. During training, reference trajectories are sampled from $\mathcal{D}$ and assigned to parallel simulation environments. At each step, the policy receives the current simulated state together with the next reference state:

$$s_{i,t} = (s_t^{\text{cur}}, s_{i,t+1}^{\text{ref}}). \quad (2)$$

A policy $\pi_\theta(a_t | s_{i,t})$ predicts a robot action a_t to track the reference object motion and hand-object interaction. The policy is optimized using tracking rewards defined from the resulting robot and object states. By sharing policy learning across demonstrations, our formulation enables efficient conversion of large reference collections.

FlashDexRetarget combines geometry- and interaction-aware observations with conditioning on K future reference frames (Section III-A), object surface tracking and hand-object distance rewards (Section III-B), hand-wise actor-critic networks for the left and right hands (Section III-C), and off-policy learning (Section III-D).

A. Observation

To track multiple references with a shared policy, we provide a current-state input s_t^{cur} and a reference input $s_{i,t+1}^{\text{ref}}$. The latter includes the next target state and motion information over the next K steps:

$$s_t^{\text{cur}} = (x_t^{\text{cur}}, \mathcal{P}_t^{\text{cur}}, d_t^{\text{cur}}), \quad s_{i,t+1}^{\text{ref}} = (x_{i,t+1}^{\text{ref}}, \mathcal{P}_{i,t+1}^{\text{ref}}, z_{i,t+1}^{\text{ref}}), \quad (3)$$

where x_t^{cur} contains the current robot and object states, and $x_{i,t+1}^{\text{ref}}$ contains the reference hand-object states at time $t+1$.

a) *Interaction-aware observations:* To explicitly represent object geometry and hand-object relationships across references, we use object point clouds and signed distance field (SDF)-based hand-object distance features. We express these features in the wrist-local frame to provide a consistent hand-centered representation across different hand poses and object motions. Specifically, we represent the object surface using 128 uniformly sampled points. The point clouds at the current simulated and next reference poses are $\mathcal{P}_t^{\text{cur}}$ and $\mathcal{P}_{i,t+1}^{\text{ref}}$, respectively. Furthermore, to describe local hand-object relationships, we compute distances from each fingertip and wrist to the object surface, together with the corresponding surface normal vectors in the wrist frame. We concatenate these distances and normals to form d_t^{cur} . Here, the distances are queried from a precomputed SDF of each object, discretized on a 128^3 voxel grid covering a 60cm cube centered at the object origin, since computing signed distances directly from the object mesh at every simulation step is computationally expensive.

b) *K future reference conditioning:* Most prior dexterous retargeting methods condition the policy solely on the next reference frame [7–9], providing limited temporal context for anticipating subsequent motion. Prior locomotion tracking methods have shown that conditioning on multiple future reference frames over a short temporal horizon can

provide useful motion context [38, 39]. We therefore encode the reference hand and object states from $t + 1$ to $t + K$, represented relative to the wrist frame at time t :

$$\mathbf{z}_{i,t+1}^{\text{ref}} = \text{Enc}_{\psi}(\mathbf{x}_{i,t+1:t+K}^{\text{ref}}), \quad (4)$$

where $K = 10$ and Enc_{ψ} is a temporal encoder that maps the future reference sequence to a 128-dimensional latent representation. The encoder is trained from scratch jointly with the actor and critic. The policy is thus conditioned on both the next reference state and the future latent, combining immediate tracking targets with broader temporal context.

B. Reward

For each hand $h \in \{\text{left}, \text{right}\}$, we define a hand-specific reward as a weighted combination of object tracking, hand-object interaction, hand tracking, and regularization terms:

$$r_t^h = w_{\text{obj}} r_t^{\text{obj},h} + w_{\text{int}} r_t^{\text{int},h} + w_{\text{track}} r_t^{\text{track},h} + w_{\text{reg}} r_t^{\text{reg},h}. \quad (5)$$

a) *Object tracking reward*: Existing methods [8, 9] typically combine object translation and rotation errors using manually chosen weights. Such weighting requires balancing errors with different units and scales. Instead, we measure object tracking using corresponding surface points. We reuse the 128 object-frame surface points from Section III-A, $\mathcal{P}_{\text{obj},h}^{\text{h}} = \{\mathbf{p}_m^h\}_{m=1}^{128}$, and normalize them to a common radius ρ to make the reward comparable across object sizes. We then transform the normalized points using the current and reference object poses to obtain $\tilde{\mathcal{P}}_{\text{cur},h}$ and $\tilde{\mathcal{P}}_{\text{ref},h}$ in the world frame. We set $\rho = 0.058\text{m}$ so that a 30° rotation corresponds to approximately 3cm of point displacement, consistent with the success thresholds [8]. To emphasize the largest object misalignments, we use the mean of the three largest pointwise errors rather than averaging over all points. Let $\mathcal{J}_{h,t}^{(k)}$ denote the indices of the $k = 3$ largest pointwise errors. We define

$$\begin{aligned} \tilde{\mathbf{p}}_m^h &= \rho \left(\frac{\mathbf{p}_m^h}{\max_n \|\mathbf{p}_n^h\|_2} \right), \\ e_t^{\text{obj},h} &= \frac{1}{k} \sum_{m \in \mathcal{J}_{h,t}^{(k)}} \left\| \tilde{\mathcal{P}}_{t+1,m}^{\text{cur},h} - \tilde{\mathcal{P}}_{t+1,m}^{\text{ref},h} \right\|_2, \\ r_t^{\text{obj},h} &= \exp\left(-e_t^{\text{obj},h} / \sigma_{\text{obj}}\right), \end{aligned} \quad (6)$$

where $\sigma_{\text{obj}} = 0.1\text{m}$. Thus, the metric measures translation and rotation in a common unit without separate weights.

b) *Hand-object interaction reward*: Prior work [7, 9] infers contact using a fixed hand-object distance threshold. However, reconstruction gaps can exceed this threshold, causing demonstrated contacts to be incorrectly labeled as non-contact. We instead use continuous hand-object distances to supervise proximity without assigning binary contact labels. For the robot hand, we reuse the SDF-based distances introduced in Section III-A, retaining only the five fingertip distances and excluding the wrist distance. Let $d_{h,t}^{(f)}$ denote the signed distance from robot fingertip f to the object surface. For the corresponding reference fingertip, we compute the unsigned distance $d_{h,t}^{\text{ref},(f)}$ to the nearest

vertex of the object mesh. Using unsigned reference distances avoids requiring the robot to reproduce penetrations present in the reconstructed demonstration. Rather than matching the reference distance exactly, we penalize only the amount by which the robot distance exceeds it:

$$\begin{aligned} \varepsilon_{h,t}^{(f)} &= \max\left(0, d_{h,t}^{(f)} - d_{h,t}^{\text{ref},(f)}\right), \\ r_t^{\text{int},h} &= \frac{1}{5} \sum_{f=1}^5 \exp\left(-\varepsilon_{h,t}^{(f)} / \sigma_{\text{int}}\right), \end{aligned} \quad (7)$$

where $\sigma_{\text{int}} = 0.01\text{m}$. Each fingertip receives the maximum reward when $d_{h,t}^{(f)} \leq d_{h,t}^{\text{ref},(f)}$, with the reward decreasing as the excess increases. This allows the robot to close reconstruction gaps rather than reproduce them.

c) *Hand tracking reward*: To keep the robot hand motion close to the demonstration and exploit human manipulation priors, we follow prior work [8, 9] and include a hand tracking term $r_t^{\text{track},h}$. It combines exponential rewards on wrist translation, wrist rotation, and fingertip position errors with a behavior-cloning term that encourages commanded joint positions to match kinematic retargets.

d) *Regularization*: To keep the resulting robot motions physically executable, we penalize large actions $\mathbf{a}_t^h$ in $r_t^{\text{reg},h}$. We also penalize tracking failures, which occur when the object or any fingertip deviates from its reference position by more than a predefined threshold. Terminations due to timeouts or successful clip completion are not penalized.

C. Bimanual Architecture

Prior RL-based retargeting methods [7–9] control both hands with a single actor-critic pair trained on their combined reward, which makes it difficult to attribute success or failure to either hand. We therefore assign each hand a dedicated actor-critic pair trained with its own reward r_t^h . For each hand $h \in \{\text{left}, \text{right}\}$, we maintain an actor π_{θ_h} and critic Q_{ϕ_h} . Although the rewards are hand-specific, each actor and critic receives the full observation $\mathbf{s}_{i,t}$, including the states of both hands. This allows each hand to incorporate cross-hand interactions, such as when the object manipulated by one hand contacts or constrains the object manipulated by the other. Despite receiving the full observation, each actor outputs only the action of its corresponding hand:

$$\mathbf{a}_t^h \sim \pi_{\theta_h}(\cdot \mid \mathbf{s}_{i,t}), \quad \mathbf{a}_t = (\mathbf{a}_t^{\text{left}}, \mathbf{a}_t^{\text{right}}). \quad (8)$$

D. Off-Policy Learning

On-policy algorithms such as PPO [1] discard transitions after each policy update, which is data-inefficient, whereas off-policy algorithms reuse past transitions stored in a replay buffer. We therefore adopt FlashSAC [11], an off-policy RL algorithm with high sample efficiency. Unlike single-reference training, however, multi-reference training involves diverse objects as well as hand motions, which yields a much broader state distribution. To accommodate this, we enlarge the replay buffer from the default 10M to 50M transitions to retain experience across references, and increase the critic hidden dimension from the default 256 to 1024 to better estimate values over such diverse data.



Fig. 3. **Qualitative retargeting results.** Representative snapshots of dexterous robot manipulation generated by FlashDexRetarget in simulation, spanning tool use, container handling, and everyday object manipulation across diverse object geometries and hand-object configurations.

IV. EXPERIMENTS

We evaluate FlashDexRetarget in terms of retargeting success, computational efficiency, scalability, and real-world executability through four research questions:

RQ1: How does FlashDexRetarget compare with prior methods in retargeting success and computational efficiency?

RQ2: Which design components improve multi-reference retargeting performance and learning efficiency?

RQ3: Does multi-reference retargeting remain effective as the number of reference motions increases?

RQ4: Do retargeted trajectories transfer to real-world robot execution?

A. Experimental Setup

a) Datasets and preprocessing: We use human hand-object demonstrations from TACO [2], OakInk2 [3], and HOT3D [4], following the preprocessing procedure of CHORD [7]. Our collection includes single-object interactions from HOT3D and two-object interactions from TACO and OakInk2. To avoid trivial successes, we exclude inactive clips in which neither hand performs manipulation and the objects remain stationary, since such clips can satisfy the tracking criteria without meaningful robot action.

b) Evaluation protocol: For the main comparison, we select 50 reference motions comprising 25 single-object and 25 two-object interactions. We evaluate retargeting to two robot embodiments, XHand and Sharpa Wave Hand, and compare FlashDexRetarget with Do as I Do [5], DexMachina [9], and CHORD [7]. We use the released implementations of Do as I Do and CHORD, where CHORD is trained with PPO [1], and modify DexMachina to initialize from random states along the reference trajectory rather than always from the motion start. For the scaling experiments, we use reference collections of 200, 500, and 1,000 motions, maintaining approximately balanced proportions of single- and two-object interactions. All simulation experiments are conducted in Isaac Sim on NVIDIA RTX 3090 GPUs.

c) Metrics: We report three success rates: SR_{MT-Obj} , SR_{MT} , and SR_{SPIDER} . The ManipTrans metric SR_{MT} [8] uses object position and rotation error thresholds of 3cm and 30° , together with fingertip and hand-joint position error thresholds of 6cm and 8cm, respectively. Its object-only variant, SR_{MT-Obj} , retains the object-tracking criteria while omitting the hand-tracking requirements. For comparison with prior baselines, we also report SR_{SPIDER} [6], which applies object position and rotation error thresholds of 10cm and 0.5rad to object errors averaged over the active hands rather than requiring each hand’s object to satisfy them. This averaging can overestimate success when one object remains nearly stationary in the reference. We therefore use SR_{MT-Obj} for component-wise ablations, as it measures object retargeting without cross-object averaging, while retaining SR_{SPIDER} for the main comparison with prior methods. To evaluate the quality of successfully retargeted trajectories, we additionally report the mean object position error (MOPE) and mean object rotation error (MORE) among trajectories that satisfy the SR_{SPIDER} criterion.

d) Computational cost: We report computational cost in total GPU-hours. For single-reference baselines, we periodically evaluate each per-motion run, stop it once a successful rollout is obtained and stored, and record the total GPU-hours consumed up to that point. We sum these costs over all attempted motions, including the full computational cost of unsuccessful runs. For multi-reference training, we use a fixed training-step budget for each experiment and report the total GPU-hours required to reach that budget. Multi-reference success is measured cumulatively over training: a reference is counted as successful once a rollout satisfying the SR_{MT-Obj} criterion is stored. The training budgets used for each experiment are specified in the corresponding sections.

B. Retargeting Performance and Computational Efficiency

To evaluate retargeting performance and computational efficiency, we compare FlashDexRetarget with prior baselines

TABLE I
QUANTITATIVE COMPARISON OF DEXTEROUS MOTION RETARGETING METHODS

Method	Motion	SR _{SPIDER} ↑	SR _{MT-Obj} ↑	SR _{MT} ↑	MOPE (mm) ↓	MORE (rad) ↓	GPU-h ↓
<i>(a) XHand</i>							
Do as I Do [5]	Single	0.36	0.04	0.00	47.32	0.28	66
DexMachina [9]	Single	0.48	0.32	0.12	37.49	0.29	447
CHORD [7]	Single	0.46	0.02	0.00	55.70	0.16	2,847
FlashDexRetarget	Multi	0.90	0.86	0.86	10.95	0.25	29
<i>(b) Sharpa Wave Hand</i>							
Do as I Do	Single	0.64	0.10	0.10	58.33	0.24	62
CHORD	Single	0.50	0.22	0.00	60.07	0.19	3,314
FlashDexRetarget	Multi	0.86	0.80	0.80	16.45	0.32	44

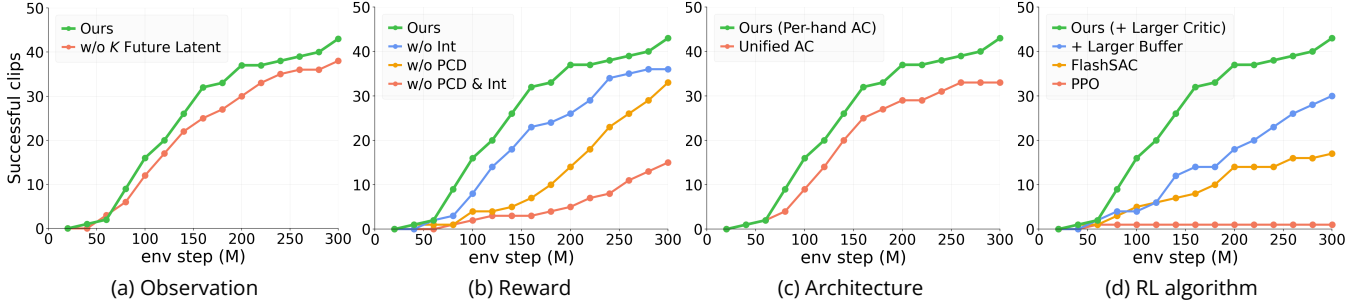


Fig. 4. **Ablation of the components of FlashDexRetarget.** Curves show the cumulative number of successfully converted reference motions from the 50-motion benchmark over 300 million environment steps. Each reference is counted once a rollout satisfies SR_{MT-Obj}. We examine (a) future reference encoding; (b) point-cloud (PCD) and interaction (Int) reward components, with the corresponding observations and rewards ablated together; (c) a unified actor-critic versus left- and right-hand actor-critic pairs; and (d) PPO, default FlashSAC, and FlashSAC with a larger replay buffer, a wider critic, or both.

on the 50-motion benchmark. We train FlashDexRetarget for 300 million environment steps on XHand. For Sharpa Wave Hand, which has a higher-dimensional control space, we use a slightly larger budget of 400 million steps. As shown in Table I, FlashDexRetarget achieves higher retargeting success while requiring substantially less computation. On XHand, under the SR_{SPIDER} criterion, FlashDexRetarget achieves at least a 40-percentage-point higher success rate than the baselines, while using approximately $2\times$ less compute than Do as I Do and nearly $100\times$ less compute than CHORD. The difference is more pronounced under the stricter SR_{MT} criterion, which additionally requires the robot hand to track the reference hand: FlashDexRetarget achieves 86% success, compared with 12% for DexMachina and 0% for Do as I Do and CHORD. To examine whether these gains depend on the target embodiment, we additionally evaluate on Sharpa Wave Hand, which is also used by Do as I Do and CHORD. The same trend holds. FlashDexRetarget achieves 86% under SR_{SPIDER}, compared with 64% for Do as I Do and 50% for CHORD, while requiring less computation. Under the stricter SR_{MT} criterion, FlashDexRetarget achieves 80% success, compared with 10% for Do as I Do and 0% for CHORD. These results show that the improvement extends across both robot embodiments.

C. Key Components for Multi-Reference Retargeting

We ablate the observation design, rewards, actor-critic architecture, and training configuration of FlashDexRetarget.

All variants are trained for 300 million environment steps. Figure 4 shows their learning progress and cumulative retargeting success measured by SR_{MT-Obj}.

a) K future reference conditioning: To evaluate the importance of K future reference conditioning, we remove the future trajectory encoding while retaining the next-step reference input. Compared with this variant, FlashDexRetarget with future reference conditioning accumulates successful trajectories faster and converts more references by the end of training. This demonstrates that temporal context beyond the immediate target helps the policy anticipate subsequent motion, improving hand pose and object rotation tracking.

b) Geometry-aware tracking and interaction rewards: To evaluate the contribution of geometry-aware tracking and hand-object interaction supervision, we ablate the two components separately and jointly. In *w/o PCD matching*, we remove the point-cloud observations and replace the point-cloud object tracking reward with the object tracking reward used in DexMachina. In *w/o interaction reward*, we remove the hand-object distance features $\mathbf{d}_i^{\text{cur}}$ together with the hand-object interaction reward. We also evaluate a variant that removes both components. Removing either component reduces learning speed and retargeting success, while removing both causes the largest degradation. The larger drop without PCD matching indicates that geometry-aware object representation and tracking play a larger role in multi-reference retargeting. At the same time, the interaction reward provides a complementary gain, with the full model

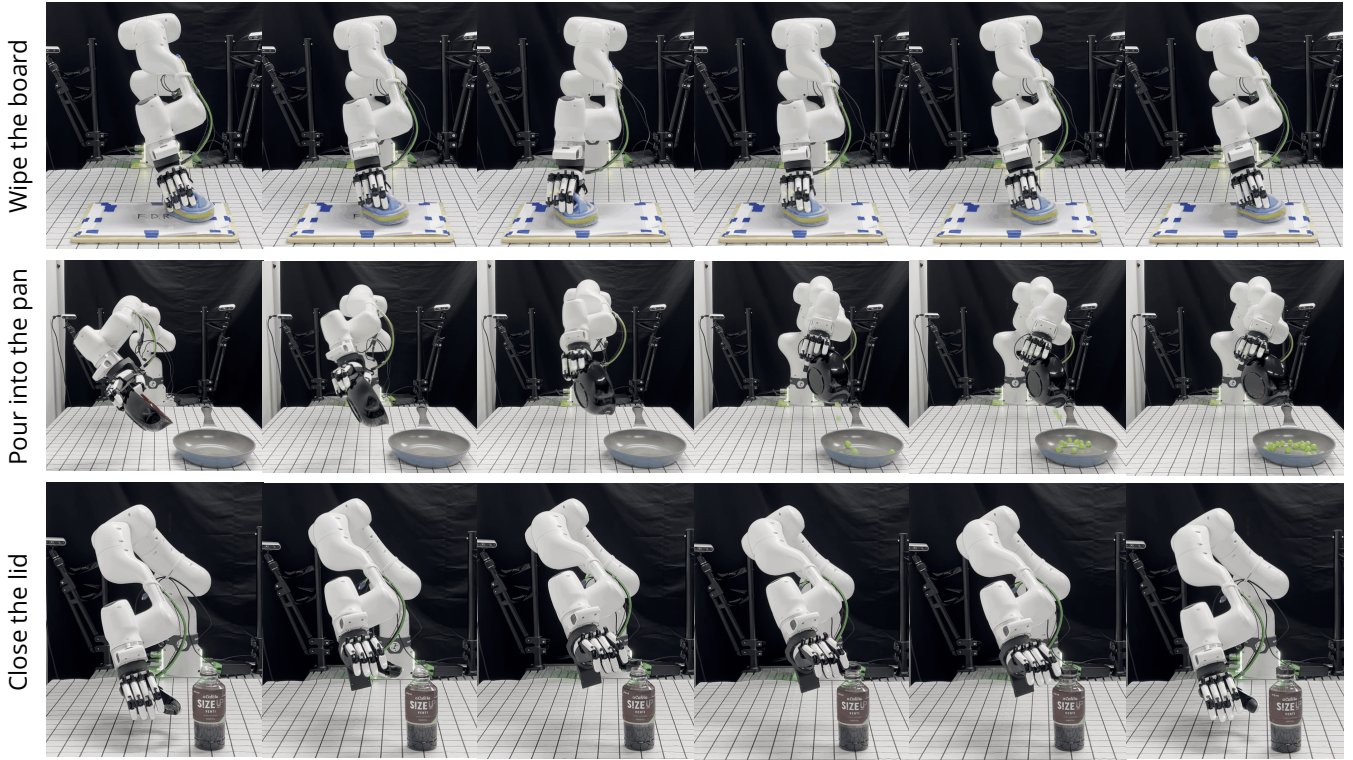


Fig. 5. **Real-world validation of retargeted trajectories.** The robot successfully executes three retargeted manipulation tasks in the real world: wiping a board, pouring into a pan, and closing a lid.

learning fastest and achieving the highest cumulative success.

c) Hand-wise actor-critic architecture: We compare our hand-wise design with a unified actor-critic that controls both hands and is trained with their combined reward. In contrast, our design assigns each hand a separate actor-critic pair trained with its own reward, while both actors observe the full bimanual state. With this separation, our method accumulates successful trajectories faster and achieves higher cumulative retargeting success. These results support training separate actor-critic pairs with hand-specific rewards to help each hand perform its role in bimanual manipulation.

d) Off-policy learning and capacity scaling: We evaluate the effects of off-policy learning and capacity scaling using PPO, default FlashSAC, FlashSAC with a larger replay buffer, and our full configuration with both a larger replay buffer and critic. Default FlashSAC uses a replay-buffer capacity of 10 million transitions and a critic hidden dimension of 256. The larger-buffer variant increases the capacity to 50 million, while our full configuration further increases the critic hidden dimension to 1024. This ablation shows the most pronounced performance gap among our design choices. Under the same budget of 300 million environment steps, PPO achieves near-zero cumulative success, whereas default FlashSAC converts approximately 34% of the reference collection. Increasing the replay-buffer capacity raises this to approximately 60%, and further scaling the critic reaches 86% with our full configuration. The large and consistent gains indicate that both effective experience reuse and sufficient critic capacity are particularly important for

learning over the broad state distribution induced by many reference trajectories.

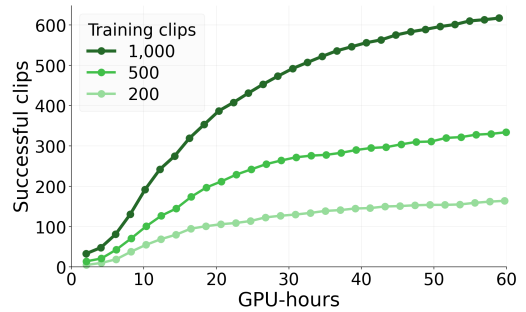


Fig. 6. **Scaling to larger reference collections.** Cumulative successful conversions for shared policies on 200, 500, and 1,000 reference motions. Larger collections yield more conversions for the same GPU-hour budget.

D. Scaling to Larger Reference Collections

To examine scalability beyond the 50-motion setting, we train shared policies on collections of 200, 500, and 1,000 reference motions. We use a budget of 600 million environment steps for all three settings, providing sufficient interaction as the number of reference motions increases. To compare their computational efficiency on a common scale, Figure 6 shows the cumulative number of successful conversions over the first 60 GPU-hours of training. Larger reference collections consistently produce more successful conversions within the same compute budget, with the gap increasing over the course of training. This indicates that exposing a shared policy to more reference motions allows

the computation spent on training to benefit a larger set of demonstrations. Overall, these results demonstrate that multi-reference training improves conversion throughput as the reference collection grows.

E. Real-World Validation

To test whether simulation-retargeted trajectories remain executable under real-world contact dynamics, we replay successful trajectories on the physical robot. As shown in Figure 5, the robot completes several manipulation tasks, including wiping a board, pouring into a pan, and closing a lid. These tasks involve different interaction requirements, from sustained surface contact during wiping to controlled object tilting during pouring and contact-guided closure. Successful execution across these tasks demonstrates that the retargeted trajectories preserve task-relevant hand-object interactions and remain physically executable beyond simulation.

V. CONCLUSION

We introduced FlashDexRetarget, an RL framework that efficiently converts human hand-object demonstrations into dexterous robot data through multi-reference tracking. The framework combines geometry- and interaction-aware observations, future reference conditioning, hand-object interaction rewards, and dedicated hand-wise actor-critic networks with off-policy learning and increased replay-buffer and critic capacity. These components enable experience reuse across demonstrations, achieving higher retargeting success at substantially lower computational cost than the evaluated baselines on XHand and Sharpa Wave Hand. Ablations demonstrate the contributions of individual components, while scaling experiments show higher conversion throughput with larger reference collections under the same compute budget. Together, these results support joint retargeting as an efficient approach to generating reusable robot data at scale.

Nevertheless, our framework relies on relatively clean human hand-object trajectories, accurate object geometry, and privileged simulator information, which may limit robustness and real-world deployment. Future work will explore robust reference processing from human videos, generalization to unseen motions and embodiments, task-aware evaluation beyond pose-based metrics, and teacher-student policy distillation to enable deployment with only real-world observations.

REFERENCES

- [1] J. Schulman *et al.*, “Proximal policy optimization algorithms,” *arXiv preprint arXiv:1707.06347*, 2017.
- [2] Y. Liu *et al.*, “TACO: Benchmarking generalizable bimanual tool-action-object understanding,” in *CVPR*, 2024.
- [3] X. Zhan *et al.*, “OakInk2: A dataset of bimanual hands-object manipulation in complex task completion,” in *CVPR*, 2024.
- [4] P. Banerjee *et al.*, “HOT3D: Hand and object tracking in 3D from egocentric multi-view videos,” in *CVPR*, 2025.
- [5] B. Paliwal *et al.*, “Do as I Do: Dexterous manipulation data from everyday human videos,” *arXiv preprint arXiv:2606.19333*, 2026.
- [6] C. Pan *et al.*, “SPIDER: Scalable physics-informed dexterous retargeting,” *arXiv preprint arXiv:2511.09484*, 2025.
- [7] X. Zhu *et al.*, “Learning dexterous manipulation using contact wrench guidance from human demonstration,” *arXiv preprint arXiv:2607.00033*, 2026.
- [8] K. Li *et al.*, “ManipTrans: Efficient dexterous bimanual manipulation transfer via residual learning,” in *CVPR*, 2025.
- [9] Z. Mandi *et al.*, “DexMachina: Functional retargeting for bimanual dexterous manipulation,” *arXiv preprint arXiv:2505.24853*, 2025.
- [10] Z. Luo *et al.*, “SONIC: Supersizing motion tracking for natural humanoid whole-body control,” *Science Robotics*, 2026.
- [11] D. Kim *et al.*, “FlashSAC: Fast and stable off-policy reinforcement learning for high-dimensional robot control,” in *RSS*, 2026.
- [12] Y.-W. Chao *et al.*, “DexYCB: A benchmark for capturing hand grasping of objects,” in *CVPR*, 2021.
- [13] O. Taheri *et al.*, “GRAB: A dataset of whole-body human grasping of objects,” in *ECCV*, 2020.
- [14] L. Yang *et al.*, “OakInk: A large-scale knowledge repository for understanding hand-object interaction,” in *CVPR*, 2022.
- [15] Y. Liu *et al.*, “HOI4D: A 4D egocentric dataset for category-level human-object interaction,” in *CVPR*, 2022.
- [16] Z. Fan *et al.*, “ARCTIC: A dataset for dexterous bimanual hand-object manipulation,” in *CVPR*, 2023.
- [17] R. Fu *et al.*, “GigaHands: A massive annotated dataset of bimanual hand activities,” in *CVPR*, 2025.
- [18] G. Zhang *et al.*, “UniDex: A robot foundation suite for universal dexterous hand control from egocentric human videos,” in *CVPR*, 2026.
- [19] H. Chen *et al.*, “Dexterous manipulation policies from RGB human videos via 3D hand-object trajectory reconstruction,” *arXiv preprint arXiv:2602.09013*, 2026.
- [20] Y. Qin *et al.*, “AnyTeleop: A general vision-based dexterous robot arm-hand teleoperation system,” in *RSS*, 2023.
- [21] A. S. Lakshmipathy *et al.*, “Kinematic motion retargeting for contact-rich anthropomorphic manipulations,” *ACM Transactions on Graphics*, 2025.
- [22] J. Mu *et al.*, “DexImit: Learning bimanual dexterous manipulation from monocular human videos,” *arXiv preprint arXiv:2602.10105*, 2026.
- [23] C. Xin *et al.*, “Analyzing key objectives in human-to-robot retargeting for dexterous manipulation,” *IEEE Robotics and Automation Practice*, 2026.
- [24] A. Handa *et al.*, “DexPilot: Vision-based teleoperation of dexterous robotic hand-arm system,” in *ICRA*, 2020.
- [25] X. Liu *et al.*, “Parameterized quasi-physical simulators for dexterous manipulations transfer,” in *ECCV*, 2024.
- [26] L. Yang *et al.*, “Physics-driven data generation for contact-rich manipulation via trajectory optimization,” in *RSS*, 2025.
- [27] P.-T. de Boer *et al.*, “A tutorial on the cross-entropy method,” *Annals of Operations Research*, 2005.
- [28] G. Williams *et al.*, “Model predictive path integral control: From theory to parallel computation,” *Journal of Guidance, Control, and Dynamics*, 2017.
- [29] T. Howell *et al.*, “Predictive sampling: Real-time behaviour synthesis with MuJoCo,” *arXiv preprint arXiv:2212.00541*, 2022.
- [30] A. H. Li *et al.*, “DROP: Dexterous reorientation via online planning,” in *ICRA*, 2025.
- [31] Z. Si *et al.*, “ExoStart: Efficient learning for dexterous manipulation with sensorized exoskeleton demonstrations,” *arXiv preprint arXiv:2506.11775*, 2025.
- [32] Y. Qin *et al.*, “DexMV: Imitation learning for dexterous manipulation from human videos,” in *ECCV*, 2022.
- [33] Y. Chen *et al.*, “Object-centric dexterous manipulation from human motion data,” in *CoRL*, 2025.
- [34] T. G. W. Lum *et al.*, “Crossing the human-robot embodiment gap with sim-to-real RL using one human demonstration,” in *CoRL*, 2025.
- [35] Z. Yuan *et al.*, “HERMES: Human-to-robot embodied learning from multi-source motion data for mobile dexterous manipulation,” *arXiv preprint arXiv:2508.20085*, 2025.
- [36] A. Rajeswaran *et al.*, “Learning complex dexterous manipulation with deep reinforcement learning and demonstrations,” in *RSS*, 2018.
- [37] Y. Feng *et al.*, “A minimalist retargeting-guided reinforcement learning recipe for dexterous manipulation,” *arXiv preprint arXiv:2607.11874*, 2026.
- [38] J. Han *et al.*, “KungfuBot2: Learning versatile motion skills for humanoid whole-body control,” *arXiv preprint arXiv:2509.16638*, 2025.
- [39] K. Lee *et al.*, “PHUMA: Physically reliable humanoid locomotion dataset,” in *CoRL*, 2026.