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

Kyungmin Lee[1*], Sibeen Kim[1*], Dongyoon Hwang[1*], Yoonsang Oh[1*], Donghu Kim[2], Youngdo Lee[2], I Made Aswin Nahrendra[2], Jaegul Choo[1], Hojoon Lee[2]

[1]KAIST AI [2]Holiday Robotics

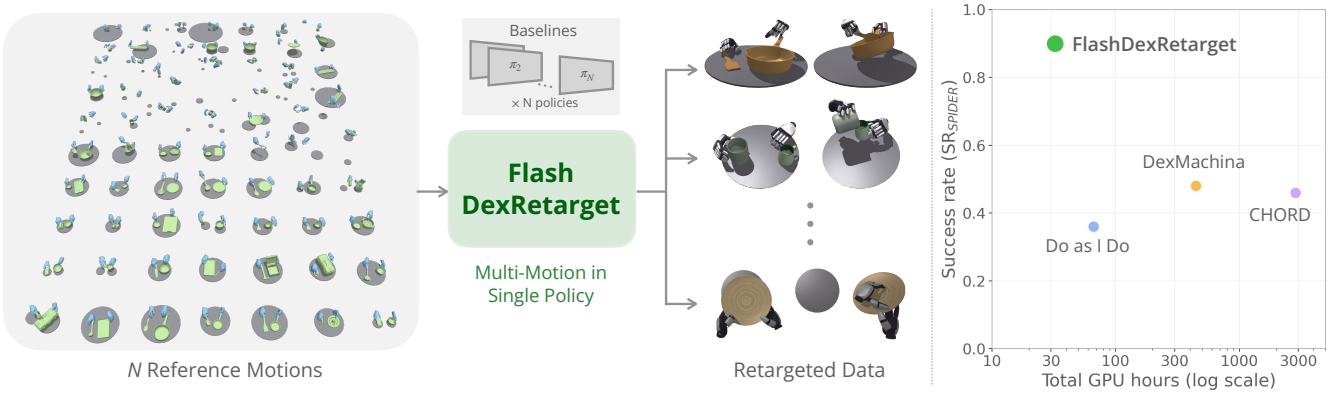

Fig. 1. **FlashDexRetarget at a glance.** FlashDexRetarget converts collections of human hand-object demonstrations into physically grounded robot trajectories through multi-reference RL-based retargeting. Left and right hand policies are jointly trained across reference motions, avoiding repeated optimization 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.

***Abstract*— Human hand-object demonstrations offer a reusable source of dexterous robot manipulation data, but transferring them across embodiments requires physically feasible retargeting. Existing physics-based approaches face limitations in retargeting success, motion-specific training efficiency, or both. To address these limitations, we introduce FlashDexRetarget, an RL-based framework for high-success, efficient dexterous motion retargeting. To make the demonstrated interaction easier to learn, we combine object point-cloud observations, hand-object signed-distance features, and future trajectory encodings with complementary rewards that supervise object motion and reference hand-object relationships. To further accelerate learning, we employ separate left- and right-hand actor critic networks and adapt the off-policy algorithm, FlashSAC to dexterous motion tracking. On a benchmark of 50 motions spanning single-object and two-object interactions, FlashDexRetarget achieves a 90% success rate, approximately 2.5× that of the evaluated sampling-based baselines, while requiring up to 100× less training compute than the evaluated RL-based baselines. Evaluations on both XHand and Sharpa Wave Hand show consistent gains, and component-wise ablations examine the contributions of our design choices. Beyond the 50-motion benchmark, experiments with 200, 500, and 1000 motions demonstrate that our method remains stable at larger scales and produces successful retargeted motions more efficiently as the training set grows. Qualitative replay results using real-world-captured demonstrations further illustrate the applicability of our framework to recorded human manipulation.**

*Equal contribution.

I. Introduction

Learning dexterous manipulation requires large amounts of robot interaction data. Human hand-object demonstrations [1–3] 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 that are 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, this process must achieve high retargeting success at low compute cost.

Existing physics-based retargeting methods adopt a single-reference formulation, optimizing each demonstration independently. Sampling-based methods [4, 5] 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 [6–8] instead learn closed-loop policies to track the demonstrated hand-object interaction, but train a separate policy for each reference. Despite their difference in 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 has been effective for humanoid whole-body motion [9], but dexterous manipulation introduces additional challenges.

First, object shapes and hand-object configurations vary across demonstrations, requiring the policy to distinguish different interaction geometries. Second, reconstructed human demonstrations 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 may play different roles with asymmetric 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 contact reconstruction. Third, we adopt the off-policy learner FlashSAC [10] and increase its replay 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's ability to convert a collection of demonstrations efficiently on 50 hand-object motions from TACO, OakInk2, and HOT3D [1–3] using XHand and Sharpa Wave Hand as target embodiments. Against sampling-based and RL-based baselines, we measure the fraction of successfully retargeted demonstrations as a function of total training compute. A jointly trained FlashDexRetarget policy retargets 90% of the benchmark using approximately 30 GPU-hours, compared with about 46% success at approximately 3,000 GPU-hours for CHORD [6]. This corresponds to approximately $100\times$ lower training compute and a 44-percentage-point improvement in retargeting success. Ablations examine how hand-object interaction objectives, off-policy training and capacity scaling, and left and right hand networks contribute to these gains in the multi-reference setting. We further show that retargeting performance remains stable as the dataset scales, so larger training sets yield proportionally more successful retargeted trajectories. We also validate the generated trajectories through real-world replay experiments.

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 [11], while GRAB and OakInk capture grasping configurations and interaction geometry [12, 13]. HOI4D and HOT3D extend this coverage to egocentric interactions [3, 14], and TACO, OakInk2, and ARCTIC capture coordinated bimanual manipulation [1, 2, 15]. GigaHands offers large-scale recordings of activities such as pouring, assembling objects, and playing cards [16]. These datasets provide diverse manipulation experience for robot learning.

However, these recordings capture human hands, so a robot cannot execute them directly, and kinematic retargeting is the common first step to close the embodiment gap. UniDex [17] converts human hand-object trajectories into robot trajectories by solving fingertip inverse kinematics and adjusting the hand base by hand so that the retargeted fingers still appear to touch the object. VideoManip [18] 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 [19–23], while physics-based optimization additionally accounts for hand-object contact dynamics in simulation [24, 25]. Within this physics-based setting, sampling-based methods search over simulated control rollouts [26–28] to support dexterous planning [29] and demonstration refinement [30]. For demonstration retargeting, SPIDER [5] 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 [4] 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 [6–8, 31–36]. ManipTrans [7] learns residual corrections to a pretrained hand-motion imitator, while DexMachina [8] progressively reduces object assistance during training. CHORD [6] instead uses

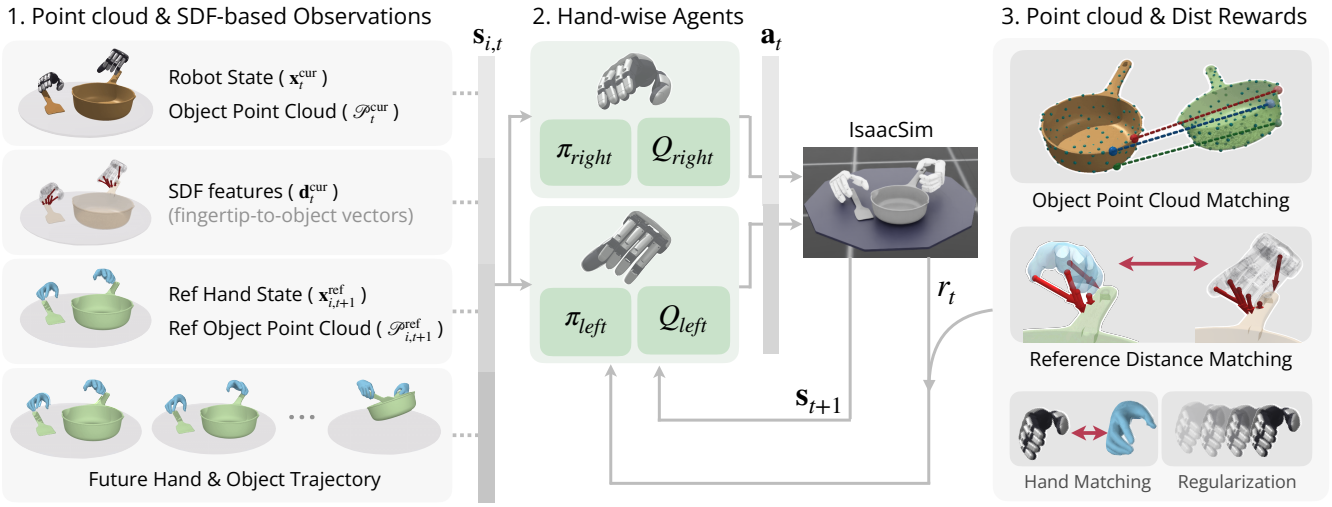

Fig. 2. **Overview of FlashDexRetarget.** (1) Observations combine robot states, current and reference object point clouds, local signed-distance features, and an encoding of future reference motion. (2) Left and right hand actor-critic pairs are trained with FlashSAC and jointly control the simulated 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.

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.

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,

$$\mathscr{D} = \{\tau_i^{\text{ref}}\}_{i=1}^{N}, \qquad \tau_i^{\text{ref}} = \{\mathbf{s}_{i,t}^{\text{ref}}\}_{t=1}^{H_i}, \tag{1}$$

where i indexes the reference demonstration, $\mathbf{s}_{i,t}^{\text{ref}}$ 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 $\mathscr{D}$. During training, reference trajectories are sampled from $\mathscr{D}$ and assigned to parallel simulation environments. At each step, the policy receives the current simulated state together with the next reference state:

$$\mathbf{s}_{i,t} = \left(\mathbf{s}_t^{\text{cur}}, \mathbf{s}_{i,t+1}^{\text{ref}}\right). \tag{2}$$

A policy $\pi_\theta(\mathbf{a}_t \mid \mathbf{s}_{i,t})$ predicts a robot action $\mathbf{a}_t$ 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 hand–object interaction-aware observations with future reference conditioning (Section III-A), object surface tracking and hand–object distance rewards (Section III-B), hand-specific actor–critic networks for 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 $\mathbf{s}_t^{\text{cur}}$ and a reference input $\mathbf{s}_{i,t+1}^{\text{ref}}$. The latter includes the next target state and motion information over the next K steps:

$$\begin{aligned}\mathbf{s}_t^{\text{cur}} &= \left(\mathbf{x}_t^{\text{cur}}, \mathscr{P}_t^{\text{cur}}, \mathbf{d}_t^{\text{cur}}\right),\\ \mathbf{s}_{i,t+1}^{\text{ref}} &= \left(\mathbf{x}_{i,t+1}^{\text{ref}}, \mathscr{P}_{i,t+1}^{\text{ref}}, \mathbf{z}_{i,t+1}^{\text{ref}}\right)\end{aligned} \tag{3}$$

where $\mathbf{x}_t^{\text{cur}}$ contains the current robot and object states, and $\mathbf{x}_{i,t+1}^{\text{ref}}$ 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 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 in wrist-local coordinates. The point clouds at the current simulated and next reference poses are $\mathscr{P}_t^{\text{cur}}$ and $\mathscr{P}_{i,t+1}^{\text{ref}}$, 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 hand's wrist frame. We concatenate these distances and normals to form $\mathbf{d}_t^{\text{cur}}$.

b) Future reference conditioning: Most prior dexterous retargeting methods condition the policy solely on the next reference frame [6–8], providing limited temporal context for

anticipating subsequent motion. Prior locomotion tracking methods have shown that short future reference windows can provide useful motion context [37, 38]. We therefore encode the reference hand and object states from $t+1$ to $t+K$, represented relative to the wrist frame at time t:

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

where $K = 10$ and Enc_ψ is a temporal encoder that maps the future reference sequence to a 128-dimensional latent representation. This provides the policy with temporal context beyond the next reference state.

B. Reward

For each hand $h \in \{\text{left}, \text{right}\}$, we combine object tracking, hand-object interaction, hand tracking, and regularization rewards as

$$r_t^h = r_t^{\text{obj},h} + w_{\text{int}}\, r_t^{\text{int},h} + w_{\text{track}}\, r_t^{\text{track},h} + w_{\text{reg}}\, r_t^{\text{reg},h}, \tag{5}$$

where w_{int}, w_{track}, and w_{reg} weight the interaction, hand tracking, and regularization terms, respectively.

a) Object tracking reward: Existing methods [7, 8] 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 directly from corresponding surface points in the world frame. We reuse the 128 object surface points introduced in Sec. III-A and compute their pointwise errors against the reference object pose. For a uniform reward signal across object sizes, we rescale the object-frame surface points $\mathscr{P}^{\text{obj}} = \{\mathbf{p}_m\}$ to a common radius ρ before applying the object poses, yielding $\tilde{\mathscr{P}}^{\text{cur}}$ and $\tilde{\mathscr{P}}^{\text{ref}}$. We set $\rho = 0.058\,\text{m}$ so that a 30° rotation yields a 3 cm error, matching the success thresholds. Rather than averaging over all points, we use the mean of the three largest errors to emphasize the most significant object misalignments. Let $\mathscr{I}_t^{(k)}$ denote the indices of the $k = 3$ largest pointwise errors. We define

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

where $\sigma_{\text{obj}} = 0.1\text{m}$. This single metric captures both translation and rotation errors.

b) Hand-object interaction reward: Contact information in motion-capture demonstrations can be imprecise, making exact contact matching unreliable. In particular, reconstructed hand and object meshes are often separated by small gaps even when the demonstrated interaction indicates contact. Therefore, prior works [6, 8] infer contact states by considering hand–object vertex pairs within 1 cm as contacts. However, due to reconstruction errors, many physically meaningful contacts can still have distances larger than this threshold, making binary contact labels insufficient for reliable supervision. We therefore use continuous hand–object distance signals instead of discrete contact labels. For each reference fingertip f, we compute the unsigned distance $d_t^{\text{ref},(f)}$ to the nearest vertex of the object mesh. We use unsigned reference distances to avoid encouraging penetration when the MANO fingertip lies inside the object. For the robot hand, we query the pre-defined signed distance field at each fingertip position:

$$d_t^{(f)} = \text{SDF}\left(\mathbf{x}_t^{(f)}\right), \qquad f \in \{1,\ldots,5\}. \tag{7}$$

We penalize only the distance exceeding the corresponding reference distance and define the interaction reward as:

$$\begin{aligned}
\varepsilon_t^{(f)} &= \max\left(0,\ d_t^{(f)} - d_t^{\text{ref},(f)}\right), \\
r_t^{\text{int}} &= \frac{1}{5}\sum_{f=1}^{5} \exp\left(-\varepsilon_t^{(f)}/\sigma_{\text{int}}\right),
\end{aligned} \tag{8}$$

where $\sigma_{\text{int}} = 0.01\text{m}$. Thus, each fingertip receives the maximum reward when its distance to the object is no larger than the corresponding reference distance.

c) Hand tracking reward: To keep the robot motion close to the demonstration, we use an auxiliary hand-tracking reward r_t^{track}. It combines exponential penalties on wrist position, wrist orientation, and mean fingertip position errors with a behavior-cloning term toward kinematic retargets.

d) Regularization: We penalize large action magnitudes and failures caused by the hand or object exceeding predefined distance thresholds. Timeouts and successful clip completion are not counted as failures. For bimanual clips, we evaluate each hand relative to its associated object, which may be shared, and use its reward r_t^h to supervise its critic.

C. Bimanual Architecture

With a single actor-critic pair, rewards from both hands are aggregated into a scalar learning signal that does not explicitly distinguish their contributions. We therefore assign each hand a dedicated actor-critic pair trained with its hand-specific reward, providing a direct learning signal for each hand. Our ablations show that this decomposition improves retargeting success, particularly when each hand manipulates a separate object and must satisfy distinct interaction objectives. For each hand $h \in \{\text{left}, \text{right}\}$, we maintain an actor π_{θ_h} and critic Q_{ϕ_h}. Each actor predicts the action for its hand:

$$\mathbf{a}_t^h \sim \pi_{\theta_h}(\cdot \mid \mathbf{s}_{i,t}), \qquad \mathbf{a}_t = \left(\mathbf{a}_t^{\text{L}}, \mathbf{a}_t^{\text{R}}\right). \tag{9}$$

The two actions are executed jointly in simulation. Each actor observes the full bimanual hand-object state but controls only its own hand. Thus, the two controllers receive separate learning signals while remaining physically coupled through the simulated interaction.

D. Off-policy Learning

To improve data efficiency in multi-reference training, we adopt FlashSAC [10], a state-of-the-art off-policy RL algorithm that reuses transitions across references through a replay buffer. To accommodate the broader state distribution induced by multi-reference training, we scale the default

Fig. 3. **Qualitative retargeting results.** Representative snapshots of dexterous robot manipulation generated by FlashDexRetarget in simulation. The examples span tool use, container handling, and manipulation of everyday objects, illustrating the variety of object geometries and hand-object configurations covered by the retargeted motions.

FlashSAC configuration by increasing the replay-buffer capacity from 10M to 50M transitions and the critic hidden dimension from 256 to 1024.

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 [1], OakInk2 [2], and HOT3D [3], following the preprocessing procedure of CHORD [6]. 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 [4], DexMachina [8], and CHORD [6]. For DexMachina, we modify the original initialization scheme to use random state initialization (RSI) along the reference trajectory, rather than always starting from the beginning of the motion. For Do as I Do and CHORD, we use the released implementations; the open-source CHORD implementation uses PPO [39]. All methods are evaluated in IsaacSim on NVIDIA RTX 3090 GPUs. For the scaling experiments, we use reference collections of 200, 500, and 1,000 motions, maintaining roughly equal proportions of single- and two-object interactions. Object-size normalization (Sec. III-B) is applied only to our XHand result in Table I; the Sharpa Wave Hand, ablation, and scaling results do not yet use it.

c) Metrics: We report three success rates: $\mathrm{SR}_{\text{MT-Obj}}$, SR_{MT}, and $\mathrm{SR}_{\text{SPIDER}}$. The ManipTrans metric SR_{MT} [7] uses object position and rotation error thresholds of 3 cm and 30°, together with fingertip and hand-joint position error thresholds of 6 cm and 8 cm, respectively. Its object-only variant, $\mathrm{SR}_{\text{MT-Obj}}$, retains the object-tracking criteria while omitting the hand-tracking requirements. We also report $\mathrm{SR}_{\text{SPIDER}}$ [5] to facilitate comparison with prior baselines. It applies object position and rotation error thresholds of 10 cm and 0.5 rad after averaging errors across the objects associated with the active hands. This averaging can overestimate success when one object remains nearly stationary in the reference. We therefore use $\mathrm{SR}_{\text{SPIDER}}$ for the main compute–success comparison with prior methods, while $\mathrm{SR}_{\text{MT-Obj}}$ is used for component-wise ablations to more directly measure changes in object retargeting performance without cross-object averaging. 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 $\mathrm{SR}_{\text{SPIDER}}$ criterion.

d) Computational cost: We report computational cost in total GPU-hours. For single-reference baselines, we periodically evaluate intermediate solutions during optimization. Once a successful rollout is obtained, we store it, stop the per-motion run, and record the GPU-hours consumed up to that point. We sum the costs across all attempted motions, including the full computational cost of unsuccessful runs.

TABLE I

QUANTITATIVE COMPARISON OF DEXTEROUS MOTION RETARGETING METHODS

Method	Motion	SR_{SPIDER} ↑	$SR_{MT\text{-}Obj}$ ↑	SR_{MT} ↑	MOPE (mm) ↓	MORE (rad) ↓	GPU-h ↓
(a) XHand							
Do as I Do [4]	Single	0.36	0.04	0.00	47.32	0.28	66
DexMachina [8]	Single	0.48	0.32	0.12	37.49	0.29	447
CHORD [6]	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	**32**
(b) Sharpa Wave Hand							
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**	**0.72**	**0.60**	**0.60**	**17.28**	0.29	**36**

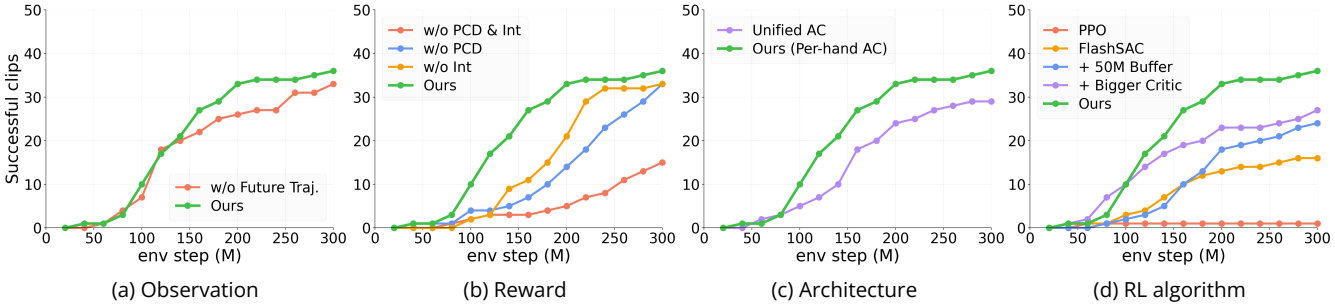

Fig. 4. **Ablation of the components of FlashDexRetarget.** Curves show the cumulative number of successfully converted reference motions from the 50-motion benchmark over 300 million environment steps. Each reference is counted once a rollout satisfies $SR_{MT\text{-}Obj}$. We examine (a) future reference 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.

For multi-reference training, we use a fixed budget of 300 million steps and report the total GPU-hours required to reach this budget. Multi-reference success is measured cumulatively over training: a reference is counted as successful once a rollout satisfying the $SR_{MT\text{-}Obj}$ criterion is stored.

B. Retargeting Performance and Computational Efficiency

To evaluate retargeting performance and computational efficiency, we compare FlashDexRetarget with prior baselines on the 50-motion benchmark. 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× less compute than Do as I Do and nearly 100× 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 higher retargeting success with lower computational cost, suggesting that the improvement is not specific to a single robot hand.

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 under the $SR_{MT\text{-}Obj}$ criterion.

a) Future reference conditioning: To evaluate the importance of future reference conditioning, we remove the future trajectory encoder while retaining the next-step reference input. Compared with the variant without future conditioning, 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 reference target helps the policy anticipate subsequent motion. In particular, it improves hand pose and object rotation tracking by providing additional guidance on the desired interaction flow.

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 our point-cloud object tracking reward with the object tracking reward used in DexMachina. In *w/o interaction reward*, we remove the SDF-based hand-object features together with the hand-object interaction reward. We also evaluate a variant that removes both components. Removing both

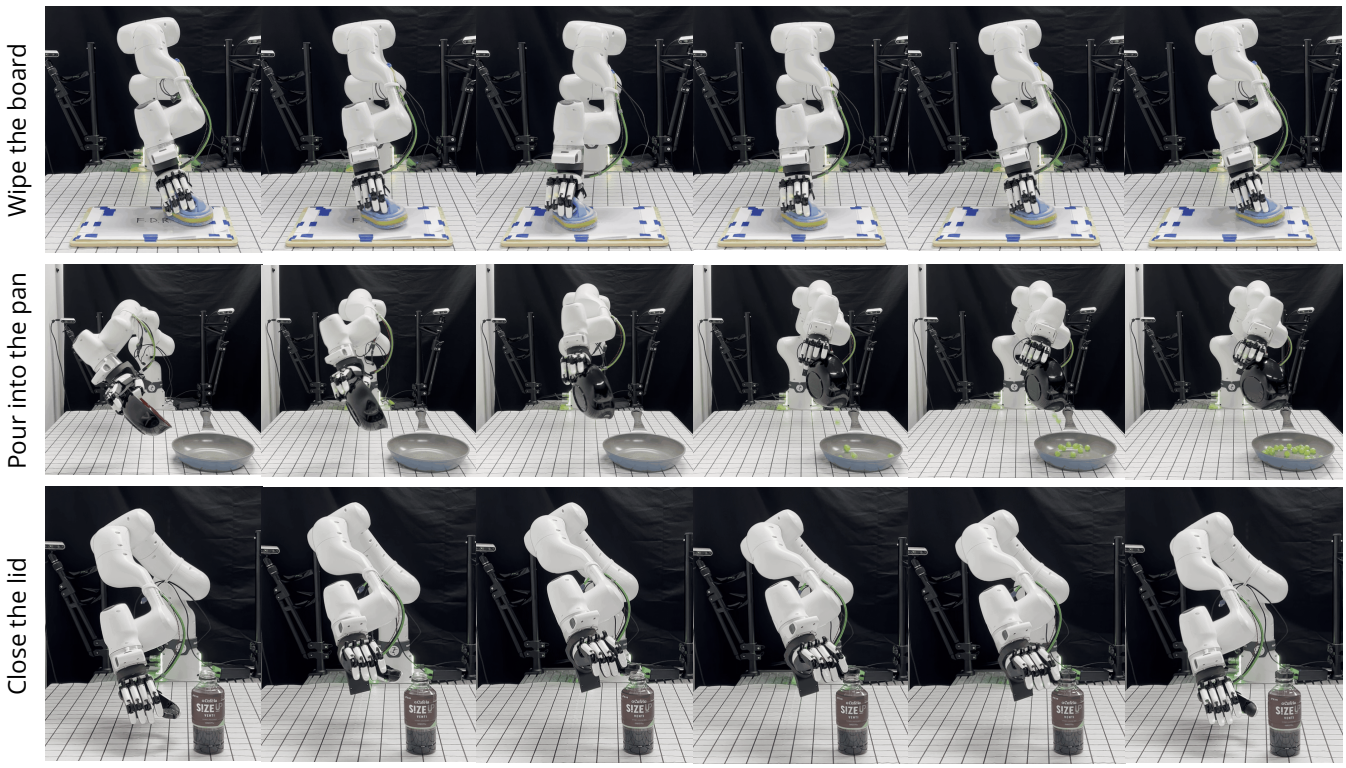

Fig. 5. **Real-world validation of retargeted trajectories.** The robot successfully executes three retargeted manipulation tasks in the real world: wiping a board, pouring into a pan, and closing a lid.

components causes the largest degradation in learning speed and cumulative retargeting success. Retaining either component improves performance, while using both leads to the fastest accumulation of successful trajectories and the largest number of converted references.

c) Left and right hand actor critic architecture: We compare our separate left- and right-hand actor critic networks with a unified actor critic architecture that controls both hands. The left and right hand architecture improves both learning speed and cumulative retargeting success, supporting separate left- and right-hand control.

d) Off-policy learning and capacity scaling: We compare PPO with four FlashSAC configurations: default FlashSAC, FlashSAC with a larger replay buffer, FlashSAC with a wider critic, and our full configuration. Default FlashSAC uses a replay-buffer capacity of 10 million transitions and a two-block critic with a hidden dimension of 256. The *FlashSAC (50M buffer)* variant increases only the buffer capacity to 50 million transitions. The *FlashSAC (Bigger Critic)* variant increases only the critic hidden dimension to 1024, retaining the two-block structure. Our full configuration combines the larger buffer and wider critic.

The choice of training algorithm has a substantial impact on multi-reference retargeting. Under the same budget of 300 million environment steps, PPO achieves a near-zero cumulative success rate, while default FlashSAC successfully converts approximately 32% of the reference collection. This improvement is achieved without increasing the replay-buffer capacity or critic width, highlighting the importance of the underlying learning algorithm in our setting. Increasing either the buffer capacity or the critic width further improves cumulative success, and combining both changes yields the best performance among the evaluated configurations.

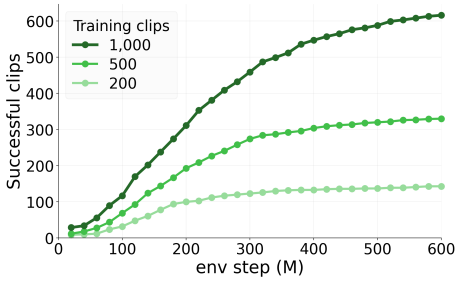

Fig. 6. **Scaling to larger reference collections.** Cumulative successful conversions when training a shared policy on collections of 200, 500, and 1,000 reference motions. A reference is counted once a rollout satisfying $\mathrm{SR}_{\text{MT-Obj}}$ is collected and stored. Larger collections yield more successfully converted motions within the same environment-interaction budget.

D. Scaling to Larger Reference Collections

We evaluate scalability beyond the 50-motion setting by increasing the reference collection to 200, 500, and 1,000 motions. Because larger collections require more interaction to expose the policy to the increased number of references, we increase the training budget from 300 million to 600 million environment steps for these experiments. For each collection size, we report the cumulative number of successfully converted references under the $\mathrm{SR}_{\text{MT-Obj}}$ criterion. As shown in Figure 6, larger reference collections yield more

successful conversions under the same training-step budget. These results show that FlashDexRetarget can generate more physically grounded robot trajectories as the collection grows, without requiring additional environment interactions.

E. Real-World Validation

We evaluate whether trajectories generated through simulation-based retargeting remain executable under real-world contact dynamics by replaying 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 distinct interaction requirements, ranging from sustained surface contact during wiping to controlled object tilting during pouring and contact-guided closure. Successful execution provides evidence that the retargeted motions preserve task-relevant hand-object interactions beyond simulation. These experiments validate the physical executability of the demonstrated trajectories rather than closed-loop policy robustness.

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 per-hand 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 that larger reference collections yield more successful trajectories within a fixed environment-interaction budget. Together, these results support joint retargeting as an efficient approach to generating reusable robot data at scale.

Despite these results, 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 real-world deployment.

References

[1] Y. Liu *et al.*, "Taco: Benchmarking generalizable bimanual tool-action-object understanding," in *CVPR*, 2024.

[2] X. Zhan *et al.*, "Oakink2: A dataset of bimanual hands-object manipulation in complex task completion," in *CVPR*, 2024.

[3] P. Banerjee *et al.*, "Hot3d: Hand and object tracking in 3d from egocentric multi-view videos," in *CVPR*, 2025.

[4] B. Paliwal *et al.*, "Do as i do: Dexterous manipulation data from everyday human videos," *arXiv preprint arXiv:2606.19333*, 2026.

[5] C. Pan *et al.*, "Spider: Scalable physics-informed dexterous retargeting," *arXiv preprint arXiv:2511.09484*, 2025.

[6] X. Zhu *et al.*, "Learning dexterous manipulation using contact wrench guidance from human demonstration," *arXiv preprint arXiv:2607.00033*, 2026.

[7] K. Li *et al.*, "Maniptrans: Efficient dexterous bimanual manipulation transfer via residual learning," in *CVPR*, 2025.

[8] Z. Mandi *et al.*, "Dexmachina: Functional retargeting for bimanual dexterous manipulation," *arXiv preprint arXiv:2505.24853*, 2025.

[9] Z. Luo *et al.*, "Sonic: Supersizing motion tracking for natural humanoid whole-body control," *Science Robotics*, 2026.

[10] D. Kim *et al.*, "Flashsac: Fast and stable off-policy reinforcement learning for high-dimensional robot control," *RSS*, 2026.

[11] Y.-W. Chao *et al.*, "Dexycb: A benchmark for capturing hand grasping of objects," in *CVPR*, 2021.

[12] O. Taheri *et al.*, "Grab: A dataset of whole-body human grasping of objects," in *ECCV*, 2020.

[13] L. Yang *et al.*, "Oakink: A large-scale knowledge repository for understanding hand-object interaction," in *CVPR*, 2022.

[14] Y. Liu *et al.*, "Hoi4d: A 4d egocentric dataset for category-level human-object interaction," in *CVPR*, 2022.

[15] Z. Fan *et al.*, "Arctic: A dataset for dexterous bimanual hand-object manipulation," in *CVPR*, 2023.

[16] R. Fu *et al.*, "Gigahands: A massive annotated dataset of bimanual hand activities," in *CVPR*, 2025.

[17] G. Zhang *et al.*, "Unidex: A robot foundation suite for universal dexterous hand control from egocentric human videos," 2026.

[18] H. Chen *et al.*, "Dexterous manipulation policies from rgb human videos via 3d hand-object trajectory reconstruction," *arXiv preprint arXiv:2602.09013*, 2026.

[19] Y. Qin *et al.*, "AnyTeleop: A general vision-based dexterous robot arm-hand teleoperation system," in *RSS*, 2023.

[20] A. S. Lakshmipathy *et al.*, "Kinematic motion retargeting for contact-rich anthropomorphic manipulations," *ACM Transactions on Graphics*, 2025.

[21] J. Mu *et al.*, "DexImit: Learning bimanual dexterous manipulation from monocular human videos," *arXiv preprint arXiv:2602.10105*, 2026.

[22] C. Xin *et al.*, "Analyzing key objectives in human-to-robot retargeting for dexterous manipulation," *IEEE Robotics and Automation Practice*, 2026.

[23] A. Handa *et al.*, "Dexpilot: Vision-based teleoperation of dexterous robotic hand-arm system," in *Proceedings of (ICRA) International Conference on Robotics and Automation*, 2020.

[24] X. Liu *et al.*, "Parameterized quasi-physical simulators for dexterous manipulations transfer," in *ECCV*, 2024.

[25] L. Yang *et al.*, "Physics-driven data generation for contact-rich manipulation via trajectory optimization," in *RSS*, 2025.

[26] P.-T. de Boer *et al.*, "A tutorial on the cross-entropy method," *Annals of Operations Research*, 2005.

[27] G. Williams *et al.*, "Model predictive path integral control: From theory to parallel computation," *Journal of Guidance, Control, and Dynamics*, 2017.

[28] T. Howell *et al.*, "Predictive sampling: Real-time behaviour synthesis with MuJoCo," *arXiv preprint arXiv:2212.00541*, 2022.

[29] A. H. Li *et al.*, "Drop: Dexterous reorientation via online planning," in *ICRA*, 2025.

[30] Z. Si *et al.*, "ExoStart: Efficient learning for dexterous manipulation with sensorized exoskeleton demonstrations," *arXiv preprint arXiv:2506.11775*, 2025.

[31] Y. Qin *et al.*, "Dexmv: Imitation learning for dexterous manipulation from human videos," in *ECCV*, 2022.

[32] Y. Chen *et al.*, "Object-centric dexterous manipulation from human motion data," in *CoRL*, 2025.

[33] T. G. W. Lum *et al.*, "Crossing the human-robot embodiment gap with sim-to-real RL using one human demonstration," in *CoRL*, 2025.

[34] 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.

[35] A. Rajeswaran *et al.*, "Learning complex dexterous manipulation with deep reinforcement learning and demonstrations," 2018.

[36] Y. Feng *et al.*, "A minimalist retargeting-guided reinforcement learning recipe for dexterous manipulation," *arXiv preprint arXiv:2607.11874*, 2026.

[37] J. Han *et al.*, "Kungfubot2: Learning versatile motion skills for humanoid whole-body control," *arXiv preprint arXiv:2509.16638*, 2025.

[38] K. Lee *et al.*, "Phuma: Physically reliable humanoid locomotion dataset," *CoRL*, 2026.

[39] J. Schulman *et al.*, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.