

TopoRetarget: Interaction-Preserving Retargeting for Dexterous Manipulation

Jielin Wu*, Shenzhe Yao*, Guanqi He*[‡], Xiaohan Liu*[‡], Zhaoqing Zeng, Xiangrui Jiang, Han Yang, Wentao Zhang, Hang Zhao[†]
HIS, Tsinghua University

*Equal contribution [‡]Project lead [†]Corresponding author

Project Page: <https://tsinghua-mars-lab.github.io/toporetaretg-web/>

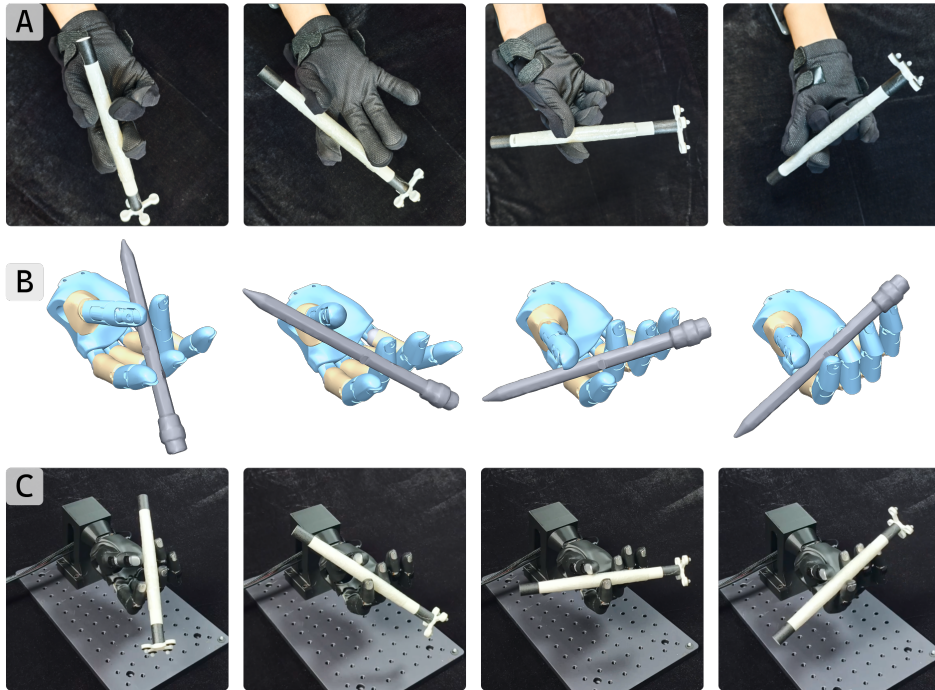


Figure 1: **TopoRetarget** converts human hand-object trajectories collected with a motion-capture glove (A) into robot references for contact-rich dexterous manipulation. These retargeted references are used to train RL policies that execute dexterous skills in simulation (B) and transfer zero-shot to Wuji Hand (C).

Abstract: Human hand-object demonstrations provide dense reference motions for training dexterous manipulation reinforcement learning (RL) policies through reference tracking. However, to use such demonstrations for RL policy learning, retargeting must preserve hand pose and task-relevant hand-object contact structure. Otherwise, contact and feasibility artifacts can degrade downstream RL policy performance. We introduce **TopoRetarget**, an interaction-preserving retargeting framework that uses a single set of parameters across diverse retargeting conditions while maintaining task-relevant hand-object interaction and adapting human demonstrations to dexterous robot hands. The method constructs a sparse interaction graph over hand and object keypoints and optimizes distance-weighted Laplacian deformation with directional consistency, kinematic constraints, and penetration handling. Evaluations show that the generated references improve both interaction fidelity and policy learning: **TopoRetarget** achieves the best contact precision and alignment over all baselines on the ContactPose Dataset, improves Pen-Spin training success by 40.6 percentage points over the existing baseline methods, and enables zero-shot transfer to Wuji Hand hardware on cube reorientation and pen spinning. Project website: [TopoRetarget](https://toporetaretg.com).

Keywords: motion retargeting, reinforcement learning, dexterous manipulation, trajectory optimization

1 Introduction

Learning long-horizon, contact-rich dexterous manipulation with reinforcement learning often relies on carefully designed training pipelines [1, 2, 3]. Human hand-object demonstrations provide dense reference signals that guide reinforcement learning through tracking objectives [4, 5, 6, 7]. Using these demonstrations for robot policy learning requires retargeting human hand-object motion across a substantial embodiment gap, but matching hand pose alone is insufficient since task execution depends on local hand-object interaction. Therefore, preserving hand-object interaction during retargeting is a key bottleneck for learning dexterous manipulation skills from human demonstrations.

Existing dexterous hand retargeting methods have made human hand motion useful for teleoperation and demonstration collection [8, 9, 10, 11]. Most of these pipelines define hand-centric objectives over hand pose [12], fingertip correspondences [8], and pinch relations [13]. Such objectives align the robot hand with the human hand, but do not directly specify where and how the robot should contact the object. Recent methods address this limitation by incorporating object motion [14, 6, 7], contact consistency [15, 16], or physical feasibility [17] into retargeting or policy learning. However, the resulting motions often still require downstream refinement, policy correction, or task-specific tuning before they can serve as reliable tracking references.

This motivates **TopoRetarget**, a retargeting framework that focuses on the structure of the local hand-object interaction. It preserves object-relative positional and directional relationships between hand links and object regions, while also maintaining intra-hand relationships among hand links. Our approach reduces interaction artifacts and supports real-time retargeting.

Our contributions are threefold:

1. An interaction-preserving retargeting framework that maintains local hand-object interaction while enforcing dexterous hand kinematic and penetration constraints.
2. A lightweight reference-based RL tracking pipeline that enables zero-shot sim-to-real transfer for contact-rich dexterous skills, including pen spinning and cube reorientation.
3. A dexterous hand-object interaction dataset of retargeted trajectories, task references, and trained policies for reproducible reference-based dexterous manipulation.

2 Related Work

2.1 Dexterous Hand Motion Retargeting

Dexterous hand motion retargeting maps human hand observations to robot hand motion for teleoperation and demonstration collection. DexPilot formulates vision-based dexterous hand-arm teleoperation around task-space hand objectives [8], and AnyTeleop extends vision-based hand-arm teleoperation across different robotic embodiments [9]. Later studies analyze how hand-pose, fingertip, keyvector, joint-limit, smoothness, and collision objectives affect retargeting quality [12], while GeoRT proposes an ultrafast geometric retargeting model for hand motion transfer [13]. These methods enable practical teleoperation and demonstration collection, but their objectives remain largely hand-centric. For contact-rich policy learning, references must preserve object-relative local interaction, including contacts on fingertips, intermediate phalanges, finger sides, and the palm. DexFlow incorporates hand-object interaction into hand pose retargeting [15], FunGrasp focuses on functional grasping for diverse dexterous hands [16], and GenHand studies generalized human grasp kinematic retargeting [18]. These methods improve grasp or contact transfer, but frequent contact-mode transitions still require references that preserve evolving hand-object relations. **TopoRetarget** treats

retargeting as interaction-preserving reference generation, maintaining local hand-object interaction while enforcing robot kinematics and penetration constraints.

2.2 Reinforcement Learning for Dexterous Manipulation

Reinforcement learning for dexterous manipulation trains policies for high-dimensional contact-rich skills through task-specific objectives, domain randomization, curriculum learning, and sim-to-real adaptation. OpenAI demonstrates cube reorientation with large-scale simulation and domain randomization [1]; DeXtreme transfers agile in-hand manipulation from simulation to real hardware [2]; RMA-based in-hand rotation adapts to object and dynamics variations through learned adaptation [19]; and recent pen-spinning systems still require carefully designed training pipelines or real-world trial-and-error search for dynamic multi-finger contact [3, 20]. To reduce this burden, reference-based learning turns long-horizon exploration into trajectory tracking. DeepMimic establishes this paradigm for physics-based character skills [21], and OmniH2O applies human-to-humanoid motion tracking to whole-body control [22]. Dexterous-hand systems have begun to adopt reference-guided policy learning: DexTrack trains neural tracking control from human references [4], Dexplore uses reference-scoped exploration [5], DexMachina studies functional retargeting for bimanual dexterous manipulation [6], and ManipTrans uses residual learning for dexterous bimanual transfer [7]. Reference-based learning depends directly on reference quality: policies must be able to follow the provided references. **TopoRetarget** targets this bottleneck by producing references that preserve task-relevant local interaction before RL training.

2.3 Interaction-Preserving Data Generation

Interaction-preserving data generation turns human hand-object behavior into policy references that retain task-relevant spatial relationships. DexMV and Robotic Telekinesis recover motion from human videos [23, 10]; From One Hand to Multiple Hands uses single-camera teleoperation for multi-hand imitation [11]; DexCap provides portable MoCap collection [24]; DexMimicGen scales a small set of human demonstrations into large bimanual dexterous datasets [25]; and the Contact-Pose Dataset provides hand-object contact annotations [26]. These sources collect hand-object motion, but embodiment gaps can still produce contact misalignment, penetration, or infeasible robot poses. ObjDex guides policy learning with object goals and human wrist motion [14], SPIDER uses physics-based sampling and virtual contact guidance for feasible dexterous trajectories [17], and OmniRetarget preserves robot-object relationships through contact-aware mesh deformation [27]. **TopoRetarget** provides real-time retargeting with local contact fidelity using a fixed set of parameters across dexterous in-hand manipulation settings.

3 TopoRetarget: Interaction-Preserving Hand-Object Retargeting

We propose **TopoRetarget**, an interaction-preserving retargeting framework for converting human hand-object demonstrations into robot references. The algorithm overview is shown in Fig. 2. We first define the notation and optimization variables in Sec. 3.1, then describe the relative bone-direction initialization in Sec. 3.2, the shared hand-object interaction mesh in Sec. 3.3, and the topology-aware Laplacian optimization in Sec. 3.4.

3.1 Notation Definition

We formulate hand-object retargeting as a constrained optimization problem. The inputs are a human-hand trajectory $P_{1:N}^h$, an object pose trajectory $q_{1:N}^o$, and an object mesh \mathcal{M} . Here, $P_t^h \in \mathbb{R}^{21 \times 3}$ denotes the MediaPipe [28] hand keypoints at time t , and N denotes the trajectory length. The goal is to compute an optimal robot base-pose trajectory $q_{1:N}^{\text{base}}$ and a robot joint trajectory $q_{1:N}^\theta \in \mathbb{R}^{N \times n_\theta}$. We write $q_t^r = (q_t^{\text{base}}, q_t^\theta)$ and use q for q_t^r in single-frame objectives.

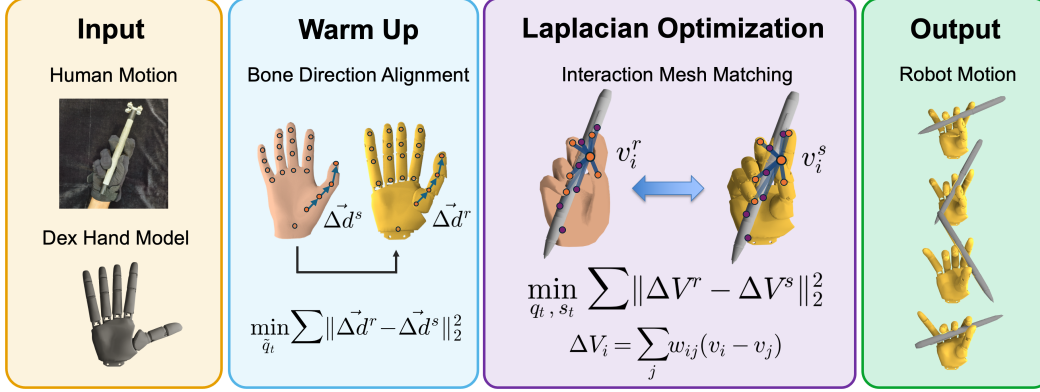


Figure 2: **TopoRetarget** overview. Given a human demonstration, object mesh, and target hand model, the method aligns bone directions during initialization, constructs source and robot interaction meshes, and computes the robot configuration via topology-aware Laplacian optimization. The output robot motion reference preserves hand-object interaction.

3.2 Relative Bone-Direction Initialization

We first compute an initial guess for the robot-hand configuration by matching local finger articulation between the source and robot hands. For each non-terminal finger keypoint k , define d_k as the unit vector from keypoint k to its child keypoint along the same finger. Let d_k^s and $d_k^r(q)$ denote the source and robot bone directions at frame t , expressed in their respective wrist-centered hand frames, and let \mathcal{A}_B contain adjacent bone pairs along each finger. The bone-direction mismatch is defined as

$$E_{\text{bone}}(q) = \sum_{(k,l) \in \mathcal{A}_B} \|(d_k^r(q) - d_l^r(q)) - (d_k^s - d_l^s)\|_2^2. \quad (1)$$

The initial estimate for frame t is

$$\tilde{q}_t^r := \arg \min_q \lambda_{\text{warm}} E_{\text{bone}}(q) + \lambda_{\text{smooth}} \|q - \tilde{q}_{t-1}^r\|_2^2. \quad (2)$$

The first term encourages the robot hand to reproduce the source hand’s relative bone directions, while the second term promotes temporal consistency with the previous initial estimate. The scalar weights λ_{warm} and λ_{smooth} balance these two objectives. We use λ_{warm} for the initialization-stage bone-direction weight.

3.3 Interaction Mesh Construction

At each frame t , we build source and robot hand-object vertex sets from the human keypoints P_t^h , the corresponding robot keypoints $P_t^r(q)$, and N_o object surface samples O_t drawn from the mesh \mathcal{M} posed by q_t^o :

$$V_t^s = [P_t^h; O_t], \quad V_t^r(q) = [P_t^r(q); O_t]. \quad (3)$$

Here, $N_v = 21 + N_o$ is the total number of graph vertices. We write $V_t^s = \{v_{i,t}^s\}_{i=1}^{N_v}$ and $V_t^r(q) = \{v_{i,t}^r(q)\}_{i=1}^{N_v}$, where the first 21 vertices are hand keypoints and the remaining vertices are object samples. We run Delaunay tetrahedralization on the source vertices V_t^s to obtain the interaction edge set \mathcal{I}_t , then reuse the same connectivity for the robot vertices:

$$G_t^s = (V_t^s, \mathcal{I}_t), \quad G_t^r(q) = (V_t^r(q), \mathcal{I}_t). \quad (4)$$

This shared graph lets the refinement compare the demonstrated and retargeted configurations under the same local hand-object neighborhood structure. It also supports augmentation across object scales and hand embodiments: after scaling or replacing the object, O_t follows the new surface, and the rebuilt graph lets the same Laplacian refinement preserve local interaction without new contact targets.

3.4 Topology-Aware Laplacian Optimization

Starting from \tilde{q}_t^r , we refine the robot configuration by matching topology-aware Laplacian coordinates on the shared interaction graph. Let $\mathcal{N}_t(i)$ be the neighbors of vertex i in \mathcal{I}_t . For each interaction edge $(i, j) \in \mathcal{I}_t$, we compute source-frame distance-aware weights

$$\tilde{w}_{ij,t} = \exp\left(-\kappa \|v_{i,t}^s - v_{j,t}^s\|_2\right), \quad w_{ij,t} = \frac{\tilde{w}_{ij,t}}{\sum_{j' \in \mathcal{N}_t(i)} \tilde{w}_{ij',t}}. \quad (5)$$

Here, κ is a spatial decay parameter. The weights are computed on the source configuration and reused for the robot configuration. For a vertex set $V = \{v_i\}_{i=1}^{N_v}$, its weighted Laplacian coordinate at vertex i is

$$\Delta_t(V)_i = \sum_{j \in \mathcal{N}_t(i)} w_{ij,t}(v_i - v_j). \quad (6)$$

We define the interaction-mesh energy as

$$E_{\text{IM}}(q) = \frac{1}{N_v} \sum_{i=1}^{N_v} \|\Delta_t(V_t^r(q))_i - \Delta_t(V_t^s)_i\|_2^2. \quad (7)$$

The final interaction-preserving optimization is

$$\begin{aligned} (q_t^{r,*}, s_t^*) = \arg \min_{q,s} \quad & \lambda_{\text{IM}} E_{\text{IM}}(q) + \lambda_{\text{bone}} E_{\text{bone}}(q) + E_{\text{reg}}(q; q_{t-1}^{r,*}) + \frac{w_s}{2} \sum_{i \in \mathcal{Q}_t} s_i^2 \\ \text{s.t.} \quad & \phi_i(q) \geq -b, \quad \phi_i(q) + s_i \geq -\tau, \quad 0 \leq s_i \leq b - \tau, \quad i \in \mathcal{Q}_t. \end{aligned} \quad (8)$$

Here, the λ terms are scalar weights, E_{IM} maintains the demonstration’s local hand-object interaction, E_{bone} keeps the initialization bone-direction prior, and E_{reg} regularizes smoothness. λ_{bone} denotes the refinement-stage bone-direction prior weight. The implementation of E_{reg} and the fixed parameter values are given in Appendix A.1. We use one fixed parameter setting across our experiments rather than performing extensive per-case tuning; detailed parameter values are provided in Table 3. The query set \mathcal{Q}_t contains hand-object pairs checked at frame t ; for pair i , $\phi_i(q)$ is the signed distance, s_i is the slack variable, τ is the soft tolerance, b is the hard bound, and w_s is the slack penalty weight.

4 Minimal RL Tracking Controller

We use a minimal reinforcement-learning setup to track the retargeted trajectories. We formulate the hand-object tracking task as a finite-horizon Markov decision process (MDP) and optimize the policy with Proximal Policy Optimization (PPO) [29]. Each reference clip is $\xi^{\text{ref}} = \{q_k^{\theta,\text{ref}}, q_k^{o,\text{ref}}, p_{1:L,k}^{\text{ref}}\}_{k=0}^{N-1}$, where $q_k^{\theta,\text{ref}}$ denotes the joint component $q_k^{\theta,*}$ of the retargeted configuration $q_k^{r,*}$, $q_k^{o,\text{ref}}$ denotes the object pose expressed in the robot base frame, and $p_{1:L,k}^{\text{ref}}$ denotes the base-frame positions of the L tracked hand links. Expressing object and link references in the base frame makes the controller focus on relative hand-object interaction rather than global motion.

Action space. At each control step t , the policy tracks the active reference frame k_t using a residual action, $q_t^{\theta,\text{tar}} = q_{k_t}^{\theta,\text{ref}} + a_t$.

Observation. The observation is $o_t = [o_t^{\text{prop}}, o_t^{\text{obj}}, o_t^{\text{ref}}]$, where o_t^{prop} contains the joint state and previous action, o_t^{obj} contains object-axis points, and o_t^{ref} contains lookahead joint, object, and hand-link references in the robot base frame.

Reference initialization and sampling. We use reference-state initialization: at reset, a start frame k_0 is sampled uniformly from the reference trajectory, and each environment is initialized with the corresponding hand and object states.

Reward. The reward is $r_t = w_{\text{obj}} r_{\text{obj}} + w_{\text{link}} r_{\text{link}} + w_{\text{joint}} r_{\text{joint}} + w_{\text{smooth}} r_{\text{smooth}}$, where the four terms track object pose, hand-link positions, finger joints, and smoothness, respectively; the weights are listed in Appendix A.5.

Domain randomization. During training, we randomize object mass and center of mass, contact parameters, actuator gains, joint damping, and hand-link inertial properties, and apply external object perturbations during rollout.

5 Experiments

Our experiments evaluate the proposed retargeting pipeline through three questions:

1. Does **TopoRetarget** preserve local hand-object interactions across diverse human motions?
2. How do retargeted references from different methods affect downstream RL tracking policy training and performance?
3. Does **TopoRetarget** support generalization and augmentation across different objects and robot hands without per-case retuning?

Across the retargeting and downstream tracking experiments, we compare the proposed method with four baselines: OmniRetarget [27] (see details at Appendix A.2), Mink [30], DexPilot [8], and GeoRT [13]. All applicable methods use the same human references and object geometry.

5.1 Interaction-Preserving Retargeting Quality

To answer Q1, we compare different retargeting algorithms in terms of their ability to preserve local contact information in hand-object interaction. We treat source in-contact bone segments as the key interaction units and evaluate preservation of object-relative positions and directions after retargeting. We report two metrics on the ContactPose Dataset [26]: E_{prec} , the mean object-relative position displacement of contact links, and E_{align} , the mean object-relative orientation error of contact links. Lower values indicate better interaction preservation. Full definitions are given in Appendix A.3.

Table 1 shows that **TopoRetarget** achieves the lowest contact precision and contact alignment errors, with 7.71 mm and 15.67° , respectively. The residual is attributable to the human-dexterous embodiment gap and source-side ContactPose interpenetration. **TopoRetarget** also keeps penetration small, with a maximum penetration of 1.07 mm, and avoids the severe interpenetration observed in several baselines, indicating that the method better preserves local contact geometry. Moreover, the average solve time is below 5 ms per frame, enabling real-time retargeting.

Table 1: Quantitative retargeting evaluation on the ContactPose Dataset [26]

Metric	TopoRetarget (Ours)	OmniRetarget	Mink	DexPilot	GeoRT
Contact precision (mm) ↓	7.71	14.15	14.12	14.13	26.77
Contact alignment ($^\circ$) ↓	15.67	30.80	37.36	33.71	25.74
Max hand-object penetration (mm) ↓	1.07	1.15	20.12	11.87	22.22
% frames with penetration > 2 mm ↓	0.00	0.00	84.00	88.00	96.00
Solve time (ms/frame) ↓	4.70	40.96	4.37	1.74	1.17

Fig. 3 summarizes common retargeting failure modes, including contact-region drift, invalid grasps or penetration, misplaced non-tip contacts, and unnatural joint configurations. In contrast, **TopoRetarget** preserves the hand-object interaction of the source while reducing artifacts. Detailed examples are shown in the figure.

5.2 Downstream Tracking Policy Performance

To answer Q2, we use downstream RL tracking policy training as a practical test of trajectory quality. For each retargeting method, we generate references on the Ho-cap [31] and our self-collected MoCap Pen-Spin Dataset (see details in Appendix A.4). We then train RL tracking policies with the setup described in Sec. 4, using identical policy architecture, reward terms, domain randomization, and termination criteria. We evaluate the trained policies in simulation and report success rate and object tracking error. A rollout is counted as successful if it reaches the final reference frame without triggering the object-pose termination criteria in Appendix A.5.

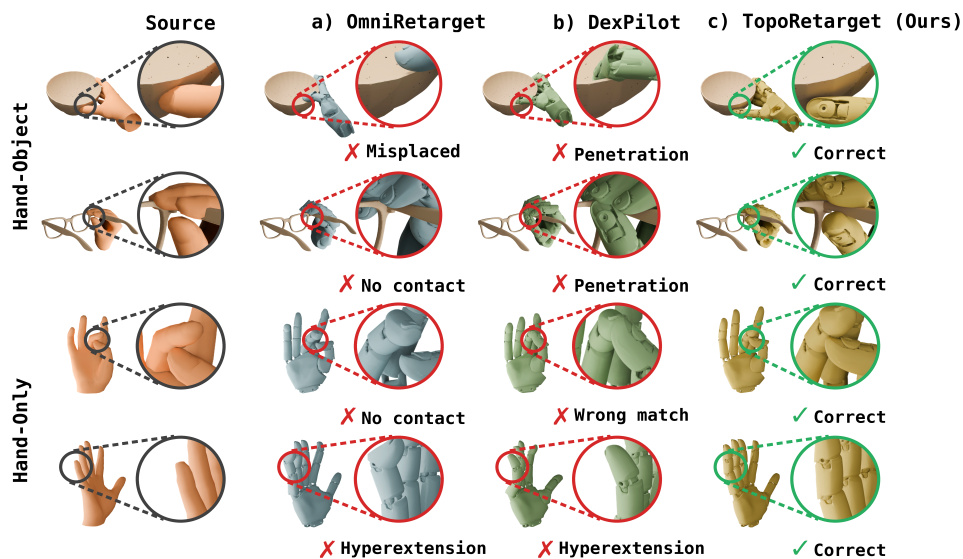


Figure 3: Retargeting artifacts of existing methods under hand-object and hand-only cases.

Table 2: Downstream RL tracking performance

Metric	TopoRetarget (Ours)	OmniRetarget	Mink	DexPilot	GeoRT
<i>Ho-cap [31] Dataset (32 clips)</i>					
Success rate \uparrow	84.4% (27/32)	56.2% (18/32)	75.0% (24/32)	75.0% (24/32)	75.0% (24/32)
Obj. pos err (cm) \downarrow	0.87	1.07	0.91	0.92	0.90
Obj. rot err (deg) \downarrow	5.76	7.87	5.55	4.99	6.07
<i>MoCap Pen-Spin Dataset (32 clips)</i>					
Success rate \uparrow	87.5% (28/32)	46.9% (15/32)	21.9% (7/32)	40.6% (13/32)	31.2% (10/32)
Obj. pos err (cm) \downarrow	0.98	1.45	1.61	1.29	1.19
Obj. rot err (deg) \downarrow	9.25	14.26	15.25	17.66	18.62

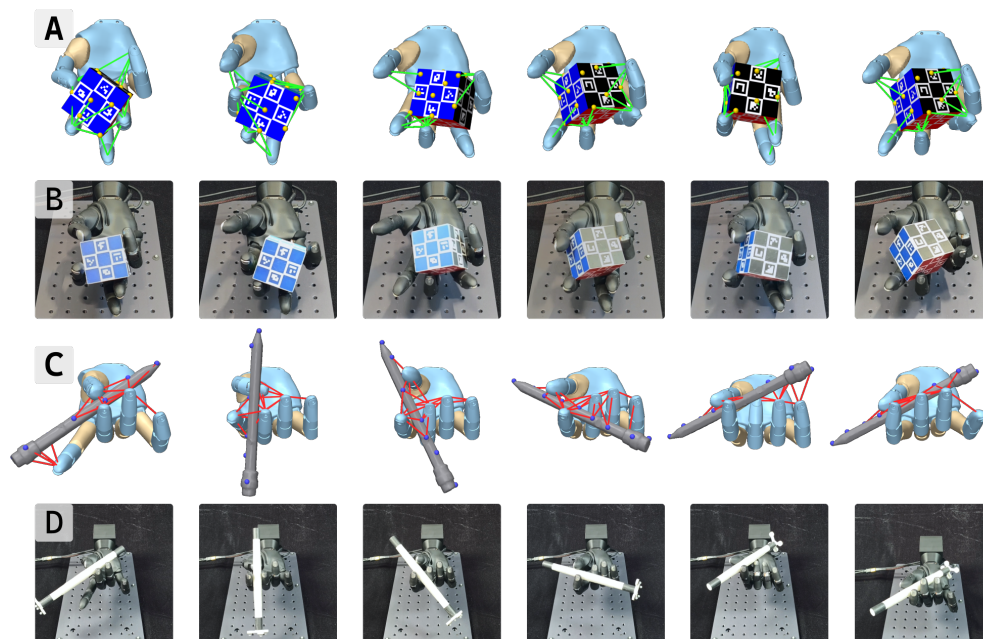


Figure 4: Interaction-mesh visualizations of retargeted reference trajectories (A,C) and corresponding zero-shot real-world executions (B,D) for cube reorientation and pen spinning on the Wuji Hand.

As shown in Table 2, **TopoRetarget** achieves the highest success rate at 84.4% and the lowest position error at 0.87 cm on the Ho-cap Dataset, while rotation errors remain comparable across methods because the trajectories are primarily grasp-centric. On the MoCap Pen-Spin Dataset, rapid motion and frequent non-tip contact transitions demand high-quality reference motion; **TopoRetarget** reaches 87.5% success, over 40 percentage points above all baselines, with the lowest position and rotation errors. The widened performance gap on pen spinning highlights the applicability of **TopoRetarget** to contact-rich manipulation, where the active hand-object contact changes frequently over time. Fig. 4 further shows zero-shot Wuji Hand sim-to-real transfer on cube reorientation and pen spinning, demonstrating that learned tracking policies can execute contact-rich skills with repeated in-hand contact changes.

5.3 Generalization and Augmentation without Per-Case Retuning

To answer Q3 on generalization and augmentation without per-case retuning, we retarget the same human demonstrations to different dexterous hand embodiments and object scales while keeping all retargeting parameters fixed. Fig. 5 shows that the proposed approach transfers both the hand motion and the hand-object interaction under the embodiment and scale changes. The interaction-mesh formulation supports this transfer by rebuilding the local hand-object graph with edge weights recomputed from the scaled object mesh, providing a simple way to augment human demonstrations.

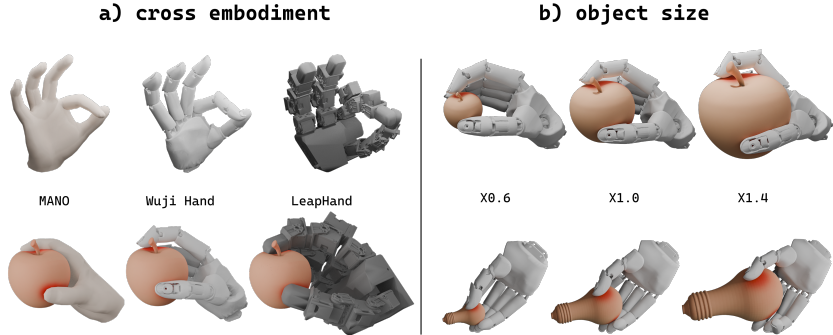


Figure 5: Augmentation across object scales and dexterous hand embodiments without per-case retuning.

6 Conclusion

We presented **TopoRetarget**, a fixed-parameter retargeting framework that converts human hand-object demonstrations into dexterous robot references. **TopoRetarget** maintains hand-object interaction in the object frame through distance-weighted Laplacian optimization, directional consistency, kinematic constraints, and penetration handling. This design improves reference-level retargeting quality, achieving better contact precision, contact alignment, and penetration metrics than representative baselines on the ContactPose Dataset. The improved references also lead to stronger downstream RL tracking, with higher success rates and lower object tracking errors, especially on agile pen-spinning demonstrations with frequent non-tip contact transitions. The same formulation supports augmentation across object scales and hand embodiments without task-specific tuning.

7 Limitations and Future Work

TopoRetarget depends on the quality of the upstream human reference motion. Our method can handle contact distortion caused by penetration in the source motion, but it is less effective for virtual contacts, where the source finger is intended to interact with the object but does not actually touch the object surface. Future work can incorporate source-motion preprocessing to correct such missing-contact cases before retargeting.

References

- [1] O. M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020.
- [2] A. Handa, A. Allshire, V. Makoviychuk, A. Petrenko, R. Singh, J. Liu, D. Makoviichuk, K. Van Wyk, A. Zhurkevich, B. Sundaralingam, et al. Dextreme: Transfer of agile in-hand manipulation from simulation to reality. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5977–5984. IEEE, 2023.
- [3] J. Wang, Y. Yuan, H. Che, H. Qi, Y. Ma, J. Malik, and X. Wang. Lessons from learning to spin” pens”. *arXiv preprint arXiv:2407.18902*, 2024.
- [4] X. Liu, J. Adalibieke, Q. Han, Y. Qin, and L. Yi. Dextrack: Towards generalizable neural tracking control for dexterous manipulation from human references. *arXiv preprint arXiv:2502.09614*, 2025.
- [5] S. Xu, Y.-W. Chao, L. Bian, A. Mousavian, Y.-X. Wang, L. Gui, and W. Yang. Dexplore: Scalable neural control for dexterous manipulation from reference scoped exploration. In *Conference on Robot Learning*, pages 2184–2199. PMLR, 2025.
- [6] Z. Mandi, Y. Hou, D. Fox, Y. Narang, A. Mandlekar, and S. Song. Dexmachina: Functional retargeting for bimanual dexterous manipulation. *arXiv preprint arXiv:2505.24853*, 2025.
- [7] K. Li, P. Li, T. Liu, Y. Li, and S. Huang. Maniptrans: Efficient dexterous bimanual manipulation transfer via residual learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6991–7003, 2025.
- [8] 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 *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9164–9170. IEEE, 2020.
- [9] 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. *arXiv preprint arXiv:2307.04577*, 2023.
- [10] A. Sivakumar, K. Shaw, and D. Pathak. Robotic telekinesis: Learning a robotic hand imitator by watching humans on youtube. *arXiv preprint arXiv:2202.10448*, 2022.
- [11] 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.
- [12] C. Xin, M. Yu, Y. Jiang, Z. Zhang, and X. Li. Analyzing key objectives in human-to-robot retargeting for dexterous manipulation. *IEEE Robotics and Automation Practice*, 2026.
- [13] Z.-H. Yin, C. Wang, L. Pineda, K. Bodduluri, T. Wu, P. Abbeel, and M. Mukadam. Geometric retargeting: A principled, ultrafast neural hand retargeting algorithm. In *2025 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 17376–17382. IEEE, 2025.
- [14] Y. Chen, C. Wang, Y. Yang, and C. K. Liu. Object-centric dexterous manipulation from human motion data. *arXiv preprint arXiv:2411.04005*, 2024.
- [15] X. Lin, K. Yao, L. Xu, X. Wang, X. Li, Y. Wang, and M. Li. Dexflow: A unified approach for dexterous hand pose retargeting and interaction. *arXiv preprint arXiv:2505.01083*, 2025.

- [16] L. Huang, H. Zhang, Z. Wu, S. Christen, and J. Song. Fungrasp: Functional grasping for diverse dexterous hands. *IEEE Robotics and Automation Letters*, 2025.
- [17] C. Pan, C. Wang, H. Qi, Z. Liu, H. Bharadhwaj, A. Sharma, T. Wu, G. Shi, J. Malik, and F. Hogan. Spider: Scalable physics-informed dexterous retargeting. *arXiv preprint arXiv:2511.09484*, 2025.
- [18] L. Qi, O. Popoola, M. A. Imran, and W. Ahmad. Genhand: generalised human grasp kinematic retargeting. *npj Robotics*, 4(1):19, 2026.
- [19] H. Qi, A. Kumar, R. Calandra, Y. Ma, and J. Malik. In-hand object rotation via rapid motor adaptation. In *Conference on Robot Learning*, pages 1722–1732. PMLR, 2023.
- [20] Y. Yao, U. Yoo, J. Oh, C. G. Atkeson, and J. Ichnowski. Soft robotic dynamic in-hand pen spinning. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, 2025.
- [21] 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.
- [22] T. He, Z. Luo, X. He, W. Xiao, C. Zhang, W. Zhang, K. Kitani, C. Liu, and G. Shi. Omnih2o: Universal and dexterous human-to-humanoid whole-body teleoperation and learning. *arXiv preprint arXiv:2406.08858*, 2024.
- [23] Y. Qin, Y.-H. Wu, S. Liu, H. Jiang, R. Yang, Y. Fu, and X. Wang. Dexmv: Imitation learning for dexterous manipulation from human videos. In *European Conference on Computer Vision*, pages 570–587. Springer, 2022.
- [24] 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.
- [25] 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.
- [26] S. Brahmabhatt, C. Tang, C. D. Twigg, C. C. Kemp, and J. Hays. Contactpose: A dataset of grasps with object contact and hand pose. In *European Conference on Computer Vision*, pages 361–378. Springer, 2020.
- [27] L. Yang, X. Huang, Z. Wu, A. Kanazawa, P. Abbeel, C. Sferrazza, C. K. Liu, R. Duan, and G. Shi. Omniretarget: Interaction-preserving data generation for humanoid whole-body locomanipulation and scene interaction. *arXiv preprint arXiv:2509.26633*, 2025.
- [28] C. Lugaresi, J. Tang, H. Nash, C. McClanahan, E. Uboweja, M. Hays, F. Zhang, C.-L. Chang, M. G. Yong, J. Lee, et al. Mediapipe: A framework for building perception pipelines. *arXiv preprint arXiv:1906.08172*, 2019.
- [29] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.
- [30] K. Zakka. Mink: Python inverse kinematics based on MuJoCo, Feb. 2026. URL <https://github.com/kevinzakka/mink>.
- [31] J. Wang, Q. Zhang, Y.-W. Chao, B. Wen, X. Guo, and Y. Xiang. Ho-cap: A capture system and dataset for 3d reconstruction and pose tracking of hand-object interaction. *Advances in Neural Information Processing Systems*, 38, 2026.

A Appendix

A.1 Retargeting Details

Implementation detail for E_{reg} . In the implementation, the regularization term in the final refinement is expanded as

$$E_{\text{reg}}(q; q_{t-1}^{r,*}) = \lambda_{\text{reg}} \|q - q_{t-1}^{r,*}\|_2^2 + \lambda_{\text{base,pos}} \|q_{\text{pos}}^{\text{base}}\|_2^2 + \lambda_{\text{base,rot}} \|q_{\text{rot}}^{\text{base}}\|_2^2. \quad (9)$$

The first term enforces temporal smoothness in the optimizer coordinates, while the last two terms are a lightweight base-pose prior for floating-base retargeting. Here, $q_{\text{pos}}^{\text{base}}$ and $q_{\text{rot}}^{\text{base}}$ denote the translational and rotational components of the robot base pose q^{base} , respectively.

Table 3: Selected retargeting parameter settings.

Parameter	Value	Description
N_o	50	Number of object surface samples in the interaction mesh.
λ_{IM}	500	Weight of the interaction-mesh Laplacian objective.
κ	30	Distance-decay factor for the exponential Laplacian edge weights.
λ_{warm}	1	Bone-direction weight in the initialization stage.
λ_{bone}	0.1	Bone-direction prior weight in the final refinement stage.
λ_{smooth}	2.5	Temporal smoothness weight in the initialization stage.
λ_{reg}	2.5	Temporal smoothness weight inside E_{reg} .
$\lambda_{\text{base,pos}}$	100	Floating-base translation prior weight used in E_{reg} .
$\lambda_{\text{base,rot}}$	1	Floating-base rotation prior weight used in E_{reg} .
τ	0.001 m	Soft penetration tolerance.
b	0.030 m	Hard penetration backstop.
w_s	1.0×10^5	Slack penalty weight for the penetration constraint.

A.2 OmniRetarget Baseline

OmniRetarget [27] is originally designed for humanoid whole-body loco-manipulation and scene interaction. For our dexterous hand-object baseline, we keep its interaction-mesh objective and object surface sampling, but replace the humanoid source keypoints with MediaPipe hand keypoints:

$$V_t^s = [P_t^{\text{mp}}; O_t^s], \quad V_t^r(q) = [P_t^r(q); O_t^r],$$

where $P_t^{\text{mp}} \in \mathbb{R}^{21 \times 3}$ denotes the source MediaPipe hand keypoints and O_t^s, O_t^r are the sampled object points under the source and retargeted object poses.

We make two hand-specific changes. First, the wrist is not fixed as a root anchor; it is included in $P_t^r(q)$ and optimized jointly with the other hand joints. Second, collision handling is applied to the full hand geometry rather than to a sparse humanoid contact set, since dexterous manipulation may contact the object with fingertips, intermediate phalanges, finger sides, or the palm.

A.3 Retargeting Metrics

Let \mathcal{C} denote the in-contact hand links, where each link c associates a source hand point h_c^s , a retargeted robot point h_c^r , the corresponding source and retargeted object points o_c^s and o_c^r , and a parent joint with source and retarget positions $h_{\text{pa}(c)}^s$ and $h_{\text{pa}(c)}^r$ (the wrist for the proximal link). The set \mathcal{C} is obtained directly from the ContactPose contact annotations using ContactPose’s own attribution: per-vertex contact intensity is sigmoid-normalized and thresholded, each contact vertex is assigned to its nearest of the 20 hand bones by point-to-segment distance, and a link is marked in contact when at least 10 vertices fall on it. Contact precision measures the object-relative position error:

$$E_{\text{prec}} = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \|(h_c^r - o_c^r) - (h_c^s - o_c^s)\|_2. \quad (10)$$

Contact alignment measures the angular deviation between the source and retargeted bone-segment orientations of the in-contact links:

$$E_{\text{align}} = \frac{1}{|\mathcal{C}|} \sum_{c \in \mathcal{C}} \cos^{-1}(\text{clip}(\hat{u}_c^r \cdot \hat{u}_c^s, -1, 1)),$$

$$\hat{u}_c^r = \frac{h_c^r - h_{\text{pa}(c)}^r}{\|h_c^r - h_{\text{pa}(c)}^r\|_2}, \quad \hat{u}_c^s = \frac{h_c^s - h_{\text{pa}(c)}^s}{\|h_c^s - h_{\text{pa}(c)}^s\|_2}.$$
(11)

For penetration, let \mathcal{X}_t^r be sampled robot-hand surface points and let \mathcal{M}_t denote the posed object mesh with signed distance $d_{\mathcal{M}_t}(x)$, positive outside the object. We report the maximum penetration depth and the fraction of frames whose maximum penetration exceeds $\delta = 2$ mm:

$$D_{\text{pen}}^{\max} = \max_t \max_{x \in \mathcal{X}_t^r} [-d_{\mathcal{M}_t}(x)]_+,$$

$$R_{\text{pen}}(\delta) = \frac{1}{N} \sum_{t=1}^N \mathbf{1} \left[\max_{x \in \mathcal{X}_t^r} [-d_{\mathcal{M}_t}(x)]_+ > \delta \right].$$
(12)

A.4 Experimental Setup

Datasets and Tasks. For reference motion retargeting evaluation, we use the ContactPose Dataset [26] to measure hand-object contact precision, contact alignment, and penetration. We evaluate on 25 of the 28 ContactPose grasps, excluding three with deeply concave object geometry (mug, scissors, Utah teapot) that no method reliably handles. For downstream tracking evaluation, we use two trajectory sets. Ho-cap [31] provides 32 grasp-centric hand-object references that test whether retargeted motions remain trackable under relatively stable contact conditions. The 32 trajectories are randomly drawn from Ho-cap’s single-hand interactions to fit our training budget. Our self-collected MoCap Pen-Spin Dataset contains 32 pen-spinning demonstrations recorded with motion capture and the Wuji Glove, with trajectories averaging 12.4 s and containing frequent non-tip contacts and contact-mode transitions. For both datasets, all retargeting methods are evaluated on the same human references and object geometry, and each generated reference set is used to train a separate policy under the identical RL setup. We further deploy the trained policies on the Wuji Hand for cube reorientation and pen spinning.

Baselines and Metrics. We compare **TopoRetarget** with OmniRetarget [27], Mink [30], DexPilot [8], and GeoRT [13] under the same human references and object geometry when applicable. Reference-level metrics include contact precision, contact alignment, maximum penetration, penetration rate, and solve time; full definitions are provided in Appendix A.3. Policy metrics include success rate and object pose tracking error.

A.5 RL Training

We formulate single-hand object manipulation as a finite-horizon MDP. The goal is to track a reference manipulation clip with a right dexterous hand while keeping the manipulated object aligned with the demonstrated motion. A reference clip is a sequence

$$\xi^{\text{ref}} = \{q_k^{\theta, \text{ref}}, q_k^{o, \text{ref}}, p_{1:L,k}^{\text{ref}}\}_{k=0}^{N-1},$$

where $q_k^{\theta, \text{ref}}$ denotes the reference finger joint positions, $q_k^{o, \text{ref}} \in \text{SE}(3)$ denotes the object pose expressed in the base frame, and $p_{1:L,k}^{\text{ref}}$ denotes the reference positions of the tracked hand links. We use the base frame for reference quantities so that the policy learns the relative hand-object interaction instead of a global trajectory.

A.5.1 State and action

The simulator state contains the hand joint state, actuator state, contact state, and object pose and velocity. The policy does not observe the full state. At each control step t , the active reference index

is k_t . The action a_t is a residual joint-position command for the n_θ right-hand finger joints. The low-level target is centered at the current reference pose,

$$q_t^{\theta, \text{tar}} = q_{k_t}^{\theta, \text{ref}} + a_t.$$

This residual parameterization lets the policy correct tracking errors while using the demonstration as a strong prior.

A.5.2 Observation

The policy observation has three parts. First, proprioception includes the current finger joint positions, joint velocities, and the previous action. Second, the object observation represents the current object pose in the base frame with a small set of axis points attached to the object. This avoids direct dependence on a particular quaternion parameterization while still encoding position and orientation. Third, the reference observation contains the current reference joint pose, object axis points, and tracked hand-link positions, plus short lookahead reference features from future frames. In our implementation we use lookahead offsets of 1, 3, and 5 control steps. During training we add small noise to proprioception and the observed object pose to improve robustness.

A.5.3 Reference initialization and sampling

We use reference initialization at every reset. A start frame k_0 is sampled uniformly from the reference trajectory, and the simulator is reset to the corresponding reference hand pose and object pose. The reference index then advances with the simulator control steps, $k_{t+1} = k_t + 1$. Uniform sampling over the trajectory exposes the policy to all phases of the manipulation and avoids overfitting to the beginning of the clip. We optionally apply small reset noise to the object pose around the sampled reference state.

A.5.4 Reward

The reward is dominated by tracking terms for the object, hand links, and joints. For an error e and scale σ , we use a Gaussian tracking kernel

$$\psi(e; \sigma) = \exp(-\|e/\sigma\|^2).$$

The object term tracks a set of axis points attached to the object, which jointly encode its base-frame position and orientation,

$$r_{\text{obj}} = \psi\left(\frac{1}{6} \sum_{m=1}^6 \|u_{m,t} - u_{m,k_t}^{\text{ref}}\|_2; \sigma_{\text{obj}}\right),$$

where $u_{m,t}$ and u_{m,k_t}^{ref} are current and reference object axis points in the base frame. The hand-link term tracks the positions of the palm and finger links,

$$r_{\text{link}} = \frac{1}{L} \sum_{\ell=1}^L \psi(\|p_{\ell,t} - p_{\ell,k_t}^{\text{ref}}\|_2; \sigma_{\text{link}}),$$

and the joint term tracks the normalized finger joint error,

$$r_{\text{joint}} = \frac{1}{n_\theta} \sum_{j=1}^{n_\theta} \psi\left(\frac{q_{j,t}^\theta - q_{j,k_t}^{\theta, \text{ref}}}{q_j^{\text{max}} - q_j^{\text{min}}}; \sigma_{\text{joint}}\right).$$

The total reward is a weighted sum:

$$r_t = w_{\text{obj}} r_{\text{obj}} + w_{\text{link}} r_{\text{link}} + w_{\text{joint}} r_{\text{joint}} + w_{\text{smooth}} r_{\text{smooth}},$$

where $r_{\text{smooth}} = \|a_t - a_{t-1}\|_2^2 + \|a_t - 2a_{t-1} + a_{t-2}\|_2^2$. In practice, the largest weight is assigned to object tracking, followed by link, joint, and smoothness terms.

The reward weights and scales, together with the termination conditions, are summarized in Table 4.

Table 4: Reference-tracking MDP. We define $\psi(e; \sigma) = \exp(-\|e/\sigma\|^2)$.

Term	Expression / specification	Weight
Object	$\psi(\frac{1}{6} \sum_{m=1}^6 \ u_m - u_m^{\text{ref}}\ _2; 0.04)$	8.0
Link position	$\frac{1}{L} \sum_{\ell=1}^L \psi(\ p_\ell - p_\ell^{\text{ref}}\ _2; 0.025)$	1.0
Joint position	$\frac{1}{n_\theta} \sum_{j=1}^{n_\theta} \psi(q_j^\theta - q_j^{\theta, \text{ref}} / (q_j^{\text{max}} - q_j^{\text{min}}); 0.1)$	1.0
Action smoothness	$\ a_t - a_{t-1}\ _2^2 + \ a_t - 2a_{t-1} + a_{t-2}\ _2^2$	-0.01
Timeout	Episode reaches 20 s	-
Object unstable	Height < 0.06 m, linear velocity > 10 m/s, or angular velocity > 500 rad/s	-
Object position error	$\ p_o - p_o^{\text{ref}}\ _2 > 0.05$ m	-
Object orientation error	Orientation error > 45°	-
Object axis-point error	Any of the 6 object axis-point errors > 0.05 m	-

A.5.5 Domain randomization

To reduce overfitting to a single simulator instance, training randomizes object mass and center of mass, contact friction, actuator gains, joint damping and armature, encoder bias, and hand link inertial properties. We also apply intermittent external force and torque disturbances to the object. These perturbations are sampled at reset or during rollout and are not included in the policy observation.

The full randomization ranges are listed in Table 5.

Table 5: Domain randomization ranges used during tracking-policy training.

Randomized quantity	Range / setting	Mode
Joint-position observation noise	$\mathcal{N}(0, 0.02)$ rad	Step obs.
Joint-velocity observation noise	$\mathcal{N}(0, 0.05)$ rad/s	Step obs.
Object axis-point position noise	$\mathcal{N}(0, 0.002)$ m	Step obs.
Object axis-point orientation noise	$\mathcal{N}(0, 0.01)$ rad	Step obs.
Observation delay	0–2 control steps	Step obs.
Reference reset: finger joints	$U[-0.02, 0.02]$ rad	Reset
Reference reset: object position	$U[-0.005, 0.005]$ m	Reset
Reference reset: object orientation	Axis-angle perturbation, angle $U[-0.03, 0.03]$ rad	Reset
Object COM offset	$[-0.003, 0.003]$ m	Startup
Robot friction scale	$[0.7, 1.3]$	Startup
Robot collision-geometry scale	$[0.9, 1.1]$	Startup
Object mass / inertia scale	$[0.4, 1.6]$	Startup
PD stiffness scale	$[0.75, 1.5]$, log-uniform	Startup
PD damping scale	$[0.5, 2.0]$, log-uniform	Startup
Joint damping scale	$[0.3, 3.0]$, log-uniform	Startup
Joint armature scale	$[0.75, 1.3]$	Startup
Joint friction-loss scale	$[0.5, 2.0]$	Startup
Encoder bias	$[-0.01, 0.01]$ rad	Startup
Robot link inertia scale	$[0.4, 1.5]$	Startup
Robot link mass scale	$[0.4, 1.5]$	Startup
External object force disturbance	$U[-0.25, 0.25]$ N, single-step impulse every 0.6–1.8 s	Rollout
External object torque disturbance	$U[-0.00375, 0.00375]$ N m	Rollout

A.5.6 Policy network and optimization

The policy and value networks and the PPO optimizer use the settings in Table 6.

Table 6: PPO training and network hyperparameters.

Parameter	Value
Parallel environments	4096
Simulation step	0.01 s
Decimation	5
RL control frequency	20 Hz
Reference trajectory frequency	20 Hz
Episode length	20 s (400 control steps)
Rollout length	40 control steps / environment
Samples per PPO iteration	163,840
Actor network	MLP [512, 256, 128], ELU
Critic network	MLP [512, 512, 256, 128], ELU
Observation normalization	Enabled
Action distribution	Softplus Gaussian
Optimizer	Adam
Learning rate	1×10^{-4}
PPO epochs / minibatches	4 / 32
Entropy coefficient	0.001
Discount γ	0.99
GAE λ	0.95