GEO2DEX: LEARNING MULTI-EMBODIMENT DEXTEROUS RETARGETING FROM HUMAN VIDEOS VIA FEED-FORWARD 3D RECONSTRUCTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Learning from human manipulation videos offers a promising path toward scalable robot learning by reducing dependence on costly robot demonstrations. Accurate retargeting is central to this goal, as the morphological gap between human and robotic hands prevents direct reuse of demonstrated motions. In this paper, we present Geo2Dex, a novel two-stage framework that couples feed-forward 3D reconstruction with embodiment-conditioned trajectory prediction to learn retargeting from paired human–robot demonstrations. Geo2Dex retains reconstructed hand motion and latent scene geometry in a shared interaction representation, which guides a structure-conditioned slot decoder to generate wrist and joint trajectories across robotic hands. Comprehensive evaluations on HRDexDB and DexCap demonstrate marked improvements in reconstruction accuracy, retargeting performance, and simulation replay success. Using these trajectories for fine-tuning also substantially improves vision–language–action policy performance in simulation and on real robots.

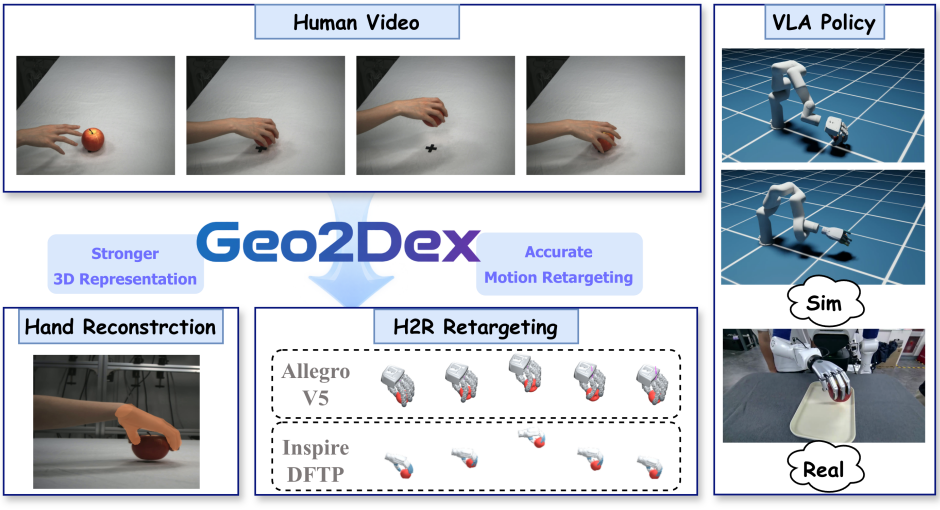

Figure 1: Overview of Geo2Dex. Enhanced spatial geometric information improves hand-motion representation and robot-action retargeting, thereby improving the performance of VLA policies trained with the generated data.

1 INTRODUCTION

The cost of collecting robot demonstrations motivates scaling robot learning with human manipulation videos recorded without operating a robot (Wang et al., 2024). A common approach reconstructs 3D hand motion from these videos and retargets it into robot action trajectories, turning human demonstrations into action supervision for policy training (Paliwal et al., 2026; Ren et al., 2026; Zheng et al., 2026).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

Reconstructing motion for this purpose requires recovering the hand's position relative to the object as well as its articulation. Image-based hand reconstruction methods estimate pose and shape (Pavlakos et al., 2024), but monocular scale–depth ambiguity limits absolute localization. HaWoR and Dyn-HaMR combine camera tracking with temporal modeling to improve global motion recovery (Zhang et al., 2025; Yu et al., 2025). Residual localization errors nevertheless displace the hand relative to the object, providing an inaccurate spatial reference for retargeting.

The recovered motion must then be adapted to the robot's geometry and actuation while preserving the demonstrated interaction. Kinematic retargeting performs this adaptation by matching fingertips or link vectors (Qin et al., 2023; Yin et al., 2025), but these objectives do not constrain hand–object contact. To improve interaction fidelity, contact-aware optimization and reinforcement learning incorporate contact or object-motion objectives (Ren et al., 2026; Mandi et al., 2025), requiring demonstration-specific optimization or simulation training. Learning from paired human–robot videos offers a reusable action mapping, as in Human2Robot (Xie et al., 2025). However, these methods lack an accurate understanding of spatial geometry, and its fixed set of output controls does not directly accommodate hands with different joint counts and functions.

A shared retargeting model therefore needs both scene-aware human motion and an action representation that accommodates each hand's structure. Feed-forward visual geometry models such as VGGT offer spatial context for this purpose by estimating camera and scene geometry, with temporal extensions capturing dynamic scenes (Wang et al., 2025b;a; Yang et al., 2026). Hand3R applies scene-level modeling to joint hand–scene reconstruction by combining a hand expert with a scene foundation model (Hu et al., 2026). Using these geometric features for retargeting further requires mapping the recovered motion and interaction context to embodiment-specific control variables.

In this paper, we introduce **Geo2Dex**, a unified framework for learning multi-embodiment dexterous retargeting from paired human–robot demonstrations. Its *Scene-Grounded Hand Reconstruction* (SGHR) module uses hand queries and camera tokens to aggregate local and global visual geometry, providing spatial cues for absolute hand localization. The module reconstructs 3D hand motion and retains latent hand and scene features to provide both motion and interaction context for retargeting.

To condition robot actions on the evolving interaction, *Multimodal Spatiotemporal Context Encoding* jointly models the reconstructed hand motion, retained visual features, robot state, and optional object pose. VideoFormer summarizes this context into an interaction memory. To accommodate different action spaces within one model, *Embodiment-Conditioned Slot Trajectory Decoding* assigns a structure-conditioned query to the wrist and each actuated joint. Joint attributes identify the action variables, while masks accommodate varying joint counts, allowing shared prediction heads to generate trajectories tailored to each hand.

Comprehensive evaluations on HRDexDB, DexCap, and real robots assess reconstruction, retargeting, and downstream vision–language–action (VLA) policy learning. On HRDexDB, Geo2Dex reduces absolute hand reconstruction error by 53.3% relative to Dyn-HaMR and raises overall replay success by 41.3% relative to DexCap. Its generated data also substantially improves real-robot policy success over robot-only fine-tuning.

In summary, our contributions are as follows:

- We introduce Geo2Dex, a novel framework that carries explicit 3D hand motion and latent visual geometry from reconstruction into human-to-robot trajectory prediction.
- An embodiment-conditioned slot trajectory decoder represents the target hand's joint structure to share action prediction across dexterous hands with different action spaces.
- Comprehensive evaluations demonstrate improved motion accuracy and task success across embodiments, with Geo2Dex-generated supervision substantially improving VLA policies in simulation and on real robots.

2 Related Work

2.1 Hand Reconstruction

Hand reconstruction is a fundamental component of human video understanding. The introduction of the MANO parametric hand model Romero et al. (2017) provides a differentiable representation

of the human hand, enabling end-to-end training. HaMeR Pavlakos et al. (2024) improves robustness by scaling up both the training data and model capacity, while WiLoR Potamias et al. (2024) combines convolutional detection with Transformer-based regression for accurate hand localization and reconstruction. More recently, temporal information is exploited to estimate more coherent hand motion from videos. Deformer Fu et al. (2023) fuses predictions from adjacent frames to refine the current hand pose. HaWoR Zhang et al. (2025) combines hand reconstruction with SLAM-based camera tracking to recover hand motion in world coordinates. Dyn-HaMR Yu et al. (2025) further handles moving cameras through a multi-stage optimization pipeline that jointly refines hand and camera motion. Despite achieving reasonably accurate 2D projections, these methods may still yield inaccurate hand trajectories in 3D space. This limitation stems from the inherent depth ambiguity of monocular videos and the coupling between camera and hand motions.

2.2 Dexterous Manipulation via Human Demonstration

Due to the high cost of collecting data for dexterous manipulation, researchers have turned to human demonstrations for robot policy learning. One straightforward strategy treats the human hand as a heterogeneous end-effector and uses human demonstrations for pretraining Kareer et al. (2025); Li et al. (2025b). Researchers have further explored aligning human and robot motions in a shared latent action space to reduce the embodiment gap and facilitate knowledge transfer Bauer et al. (2026); Jiang et al. (2025); Chen et al. (2024). Such implicit alignment lacks a reasonable and interpretable mapping between human and robot motions, limiting the full utilization of human data. Recently, methods such as EgoMimic Kareer et al. (2024) and EgoScale Zheng et al. (2026) have adopted a more direct paradigm by retargeting human hand motions from egocentric videos into actions executable by the target robot. This explicit action-level supervision provides a direct bridge between human observations and robot control, enabling a more effective use of large-scale human video data for policy learning.

2.3 Human-to-Robot Retargeting

Dexterous retargeting algorithms establish a correspondence between human hand poses and the target robot's actions. Some methods formulate human-to-robot retargeting as a kinematic optimization problem Yin et al. (2025); Qin et al. (2023); Ren et al. (2026), but their hand-crafted objectives capture embodiment-level correspondences, neglecting object geometry and environmental constraints. To address this, some methods improve the physical plausibility of dexterous grasping by training reinforcement learning policies in simulation Mandi et al. (2025); Li et al. (2025a); Xu et al. (2025). This compromises fidelity to the manipulation dynamics exhibited in real-world human demonstrations. Human-conditioned approaches translate human motions or videos into robot actions through generative prediction, contrastive learning, or shared skill representations Bharadhwaj et al. (2024); Xie et al. (2025); Pace et al. (2026). These methods lack grounding in the underlying 3D geometry.

3 Method

Geo2Dex conditions trajectory prediction on the demonstrated interaction and the target hand's structure (Fig. 2). After formulating the task in Sec. 3.1, we first temporally align human and robot trajectories (Sec. 3.2). We then present Scene-Grounded Hand Reconstruction (Sec. 3.3) and Multimodal Spatiotemporal Context Encoding (Sec. 3.4) to recover and encode the interaction. Embodiment-Conditioned Slot Trajectory Decoding (Sec. 3.5) maps this representation to robot controls, with training and trajectory generation described in Sec. 3.6.

3.1 Task Formulation

We train on paired human–robot demonstrations of the same manipulation task. For a target embodiment g with N_g actuated joints, the robot trajectory contains a wrist pose $\mathbf{P}_t^{R,g}$ and joint positions $\mathbf{q}_t^g \in \mathbb{R}^{N_g}$ at each step. The wrist pose comprises a 3D translation and a 6D rotation representation. Given an observation window of length L_o, the model predicts an H-step trajectory conditioned on the interaction context and target embodiment,

$$\widehat{\mathcal{A}}_{t:t+H-1}^g = \pi_\Theta\Big(\mathcal{C}_{t-L_o+1:t}^g, \mathbf{s}_{t-L_o+1:t}^{R,g}, \mathcal{E}^g\Big), \qquad \mathcal{A}_t^g = (\mathbf{P}_t^{R,g}, \mathbf{q}_t^g), \tag{1}$$

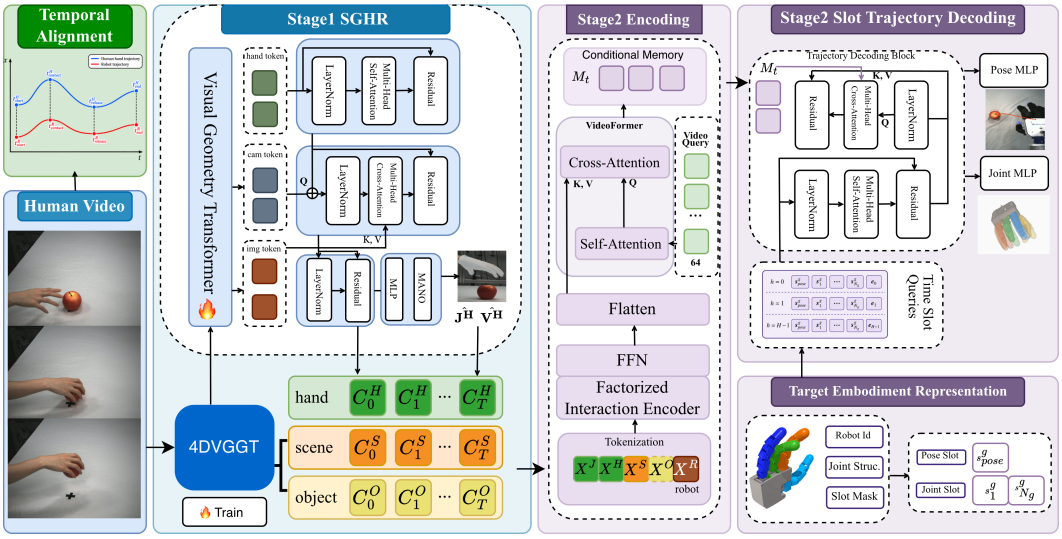

Figure 2: **Overview of Geo2Dex.** SGHR carries explicit 3D hand motion and latent visual geometry into retargeting. A spatiotemporal encoder and VideoFormer summarize the interaction context, which conditions wrist and joint queries associated with the target hand's structure. Shared prediction heads produce trajectories for hands with different action spaces.

where $\mathcal{E}^g$ specifies the identity and actuator attributes of the target end effector. The context retains both the recovered human motion and information about the surrounding interaction,

$$\mathcal{C}_t = (\mathcal{C}_t^H, \mathcal{C}_t^S, \mathcal{C}_t^O), \qquad \mathcal{C}_t^H = \left(\hat{\mathbf{J}}_t^H, \mathbf{z}_t^H\right), \qquad \mathcal{C}_t^S = \mathbf{z}_t^V, \qquad \mathcal{C}_t^O = \mathbf{T}_t^O, \tag{2}$$

Here, $\hat{\mathbf{J}}_t^H$, $\mathbf{z}_t^H$, and $\mathbf{z}_t^V$ are the reconstructed human joints, latent hand features, and scene features produced by the first stage. The robot state $\mathbf{s}_t^{R,g}$ conditions the prediction on the target robot's motion, and object pose $\mathbf{T}_t^O \in SE(3)$ supplies additional spatial context when available.

3.2 Human-Robot Temporal Alignment

Before learning implicit retargeting from human demonstrations to multiple dexterous robotic embodiments, we temporally align each paired human–robot sequence to establish frame-wise correspondences and provide temporally consistent supervision. Specifically, we define four semantic anchors on both the human and robot timelines:

$$\mathcal{T}^{H/R} = \left(t_{\text{start}}^{H/R}, t_{\text{contact}}^{H/R}, t_{\text{release}}^{H/R}, t_{\text{end}}^{H/R}\right), \tag{3}$$

corresponding to the sequence start, contact onset, contact release, and sequence end, respectively. The human contact interval is obtained from the hand–object interaction annotations, whereas the robot contact interval is identified according to the geometric proximity between the robot interaction points and the manipulated object. Details of robot contact detection are provided in the Appendix A.

Given the corresponding anchors, we construct a monotonic piecewise-linear mapping from the human timeline to the robot timeline. For a human timestamp $t^H \in [t_k^H, t_{k+1}^H]$, its corresponding continuous robot timestamp is defined as

$$\phi(t^H) = t_k^R + \frac{t_{k+1}^R - t_k^R}{t_{k+1}^H - t_k^H}\left(t^H - t_k^H\right), \qquad k \in \{0, 1, 2\}. \tag{4}$$

The mapped timestamp is clipped to the valid robot-frame range and rounded to the nearest frame, preserving temporal order despite differences in demonstration duration and execution speed. The aligned sequences are then split into fixed-length observation–action windows to train the implicit human-to-robot retargeting model.

3.3 SCENE-GROUNDED HAND RECONSTRUCTION

To anchor retargeting in the observed scene, SGHR reconstructs hand articulation and absolute 3D location using both learned hand queries and camera tokens. Given a video, the 4DVGGT backbone produces frame-wise features $\mathbf{F}_{t,\ell}^{\mathrm{frm}}$, global features $\mathbf{F}_{t,\ell}^{\mathrm{glb}}$, and camera tokens $\mathbf{c}_t$ Hu et al. (2025). We aggregate selected feature levels and use them as the context for hand decoding,

$$\begin{aligned} \mathbf{F}_t &= \mathrm{Agg}_{\ell\in\mathcal{L}}\, P_\ell\Big([\mathbf{F}_{t,\ell}^{\mathrm{frm}}; \mathbf{F}_{t,\ell}^{\mathrm{glb}}]\Big)\,, \\ \widetilde{\mathbf{Q}}_t &= \mathcal{D}_H([\mathrm{SA}(\mathbf{Q}_H); P_{\mathrm{cam}}(\mathbf{c}_t)], \mathbf{F}_t)\,. \end{aligned} \tag{5}$$

Here, $\mathcal{L}$ indexes the selected backbone levels, P_ℓ projects their concatenated features, and Agg combines them across levels. SA applies self-attention to the learned hand queries $\mathbf{Q}_H$. The cross-attention decoder $\mathcal{D}_H$ refines their concatenation with projected camera tokens, using $\mathbf{F}_t$ as keys and values. This query set retrieves hand-specific and camera-conditioned evidence from the same visual context before pose and translation are regressed.

An MLP head maps the refined tokens to MANO parameters for hand articulation, shape, and placement,

$$\begin{aligned} (\boldsymbol{\theta}_t, \boldsymbol{\beta}_t, \boldsymbol{\tau}_t) &= f_{\mathrm{MANO}}(\widetilde{\mathbf{Q}}_t), \\ (\hat{\mathbf{J}}_t^H, \hat{\mathbf{V}}_t^H) &= \mathrm{MANO}(\boldsymbol{\theta}_t, \boldsymbol{\beta}_t, \boldsymbol{\tau}_t), \end{aligned} \tag{6}$$

where $\boldsymbol{\theta}_t$, $\boldsymbol{\beta}_t$, and $\boldsymbol{\tau}_t$ denote pose, shape, and global translation. The differentiable MANO layer yields the explicit joints and mesh vertices, with absolute geometry supervised in camera coordinates. For retargeting, we retain $(\hat{\mathbf{J}}_t^H, \mathbf{z}_t^H, \mathbf{z}_t^V)$, where $\mathbf{z}_t^H$ contains the refined hand features and $\mathbf{z}_t^V$ the backbone scene features. The joints provide a spatial description of the demonstrated motion, while the latent features preserve visual evidence about its interaction context. Feature selection is detailed in Appendix E.1.

3.4 MULTIMODAL SPATIOTEMPORAL CONTEXT ENCODING

Action prediction depends on how the hand–object interaction evolves relative to the robot's state. We encode the SGHR outputs and robot-state context jointly over an observation window. For each human joint, absolute and wrist-relative positions describe global placement and articulation, while first-order temporal differences describe the motion between frames. Latent hand and scene features retain the visual context of this motion. Robot state and its temporal difference condition the prediction on the robot's configuration and motion, and available object pose supplies an explicit spatial reference.

Modality-specific projections map these inputs into a common token space, with joint-identity, modality-type, and temporal embeddings preserving their distinct roles. Within each encoder layer, spatial attention relates the hand, scene, object, and robot tokens at the same frame; temporal attention then models their evolution across the window. VideoFormer condenses the encoded sequence into an interaction memory by refining a set of learned queries $\mathbf{Q}_M$,

$$\mathbf{M}_t = \mathrm{VideoFormer}(\mathbf{Q}_M, \mathrm{STEnc}(\mathcal{X}_{t-L_o+1:t})) \in \mathbb{R}^{N_M\times D}, \tag{7}$$

where $\mathcal{X}$ is the multimodal token sequence and $\mathbf{Q}_M$ contains N_M queries of dimension D. All wrist and joint queries subsequently attend to this memory, conditioning their predictions on a shared representation of the observed interaction. Appendix B specifies the modality encodings.

3.5 EMBODIMENT-CONDITIONED SLOT TRAJECTORY DECODING

We make the target hand's action structure explicit through one prediction query for the wrist and one for each actuated joint. Each query identifies both the variable to be predicted and its structural role, allowing the same decoder to operate over different sets of actuators. The embodiment description $\mathcal{E}^g$ consists of the hand identity g, actuator attributes $\{\mathbf{a}_j^g\}_{j=1}^{N_g}$, and a validity mask for padded joint slots. Each attribute vector encodes the actuator index, digit membership, within-digit position or actuator type, and embodiment type. These descriptors distinguish, for example, thumb rotation from finger flexion; their construction for each hand is given in Appendix C.

We condition the wrist query on an aggregate description of the hand and each joint query on its own attributes,

$$\begin{aligned} \mathbf{b}^g_{\text{pose}} &= \mathbf{e}^{\text{type}}_{\text{pose}} + \mathbf{e}^{\text{emb}}_g + f_{\text{global}}(\bar{\mathbf{a}}^g), \\ \mathbf{b}^g_j &= \mathbf{e}^{\text{type}}_{\text{joint}} + \mathbf{e}^{\text{emb}}_g + \mathbf{e}^{\text{index}}_j + f_{\text{joint}}(\mathbf{a}^g_j), \qquad j = 1, \dots, N_g, \end{aligned} \tag{8}$$

where $\bar{\mathbf{a}}^g$ is the mean of the valid actuator attributes. The wrist query receives a hand-level structural summary, while the joint queries retain actuator-specific descriptors. Embodiment and index embeddings identify the output variable; its attributes specify the role of that variable within the hand.

Pairing each structural slot with a future-step embedding forms a joint representation of action variables and prediction times,

$$\mathbf{y}^{g,0}_{h,k} = \mathbf{b}^g_k + \mathbf{e}^{\text{future}}_h, \qquad \mathbf{Y}^g = \text{SlotDecoder}\Big(\{\mathbf{y}^{g,0}_{h,k}\}_{h,k}, \mathbf{M}_t\Big), \tag{9}$$

where $h = 0, \dots, H-1$ and k indexes the wrist or an actuated joint. Self-attention over this time–slot grid models dependencies among the wrist, fingers, and future steps. Cross-attention to $\mathbf{M}_t$ conditions the trajectory queries on the observed interaction, with each query retrieving context for its control variable and time step. The refined queries $\mathbf{Y}^g$ are mapped to wrist poses and joint positions by shared heads,

$$\widehat{\mathbf{P}}^{R,g}_{t+h} = f_{\text{pose}}(\mathbf{Y}^g_{h,\text{pose}}) \in \mathbb{R}^9, \qquad \widehat{q}^g_{t+h,j} = f_{\text{qpos}}(\mathbf{Y}^g_{h,j}), \quad j = 1, \dots, N_g. \tag{10}$$

In mixed-embodiment batches, joint slots are padded to the largest joint count in the batch. The mask $m^g_j = \mathbb{I}[j \le N_g]$ excludes padded slots from attention, supervision, and output interpretation. Because the joint-position head acts on individual slots, its output width is independent of the hand's total joint count. The target structure determines which queries are instantiated, while the decoder and prediction heads are shared across embodiments.

3.6 Training and Trajectory Generation

SGHR learns articulation and global localization through root-relative and absolute joint and vertex supervision, supplemented by MANO-parameter losses and pose regularization. The hand decoder and the final backbone blocks are optimized jointly, with the remaining backbone fixed. The reconstruction objective is provided in Appendix D.1.

The retargeting stage learns from the temporally aligned robot trajectories using action supervision and a motion-difference term,

$$\mathcal{L}_{\text{BC}} = \mathcal{L}_{\text{action}} + \lambda_{\text{motion}} \mathcal{L}_{\text{motion}}. \tag{11}$$

Here, $\mathcal{L}_{\text{action}}$ groups the weighted wrist-translation, wrist-rotation, and joint-position losses. Translation and joint positions use smooth-ℓ_1 supervision, and wrist rotation uses geodesic distance on $\text{SO}(3)$. The motion term matches first-order differences of wrist translations and joint positions to those of the reference trajectory, supervising how the controls change over time. Temporal masks exclude invalid future steps, and joint masks exclude padding. Appendix D.2 defines the component losses.

We sample both robotic hands equally during retargeting training and normalize wrist-pose and joint-position targets separately. Behavior cloning is followed by recursive rollout fine-tuning over consecutive windows, updating the joint-position modules to reduce accumulated prediction error. At generation time, successive video windows and robot-state context condition wrist and joint trajectories for the selected embodiment. These trajectories supply downstream VLA supervision (Appendices G and H).

4 Experiments

In experiments, we first introduce the implementation details (Sec. 4.1). Then, we benchmark the proposed **Geo2Dex** against state-of-the-art methods on dynamic human-hand reconstruction (Sec. 4.2) and human-to-robot retargeting (Sec. 4.3), respectively, to demonstrate the superiority of Stage 1 and Stage 2. Moreover, to evaluate the utility of the mapped robot data for downstream manipulation, we conduct downstream VLA policy fine-tuning (Sec. 4.4).

4.1 Implementation Details

We conduct all experiments on HRDexDB Lim et al. (2026), which contains paired human demonstrations and robot manipulation trajectories for multiple dexterous-hand embodiments. For Stage 1, we use the human RGB frames together with their same-frame MANO annotations to supervise the reconstruction of 3D hand joints and mesh vertices. Every third frame is sampled during training. For Stage 2, we use paired demonstrations from Allegro V5 and Inspire DFTP and temporally align each human–robot sequence following Sec. 3.2. The dataset is split into training, validation, and test sets at a ratio of (7:1:2). Furthermore, to evaluate the performance of our method across different datasets, we conduct additional experiments using the DexCap dataset Wang et al. (2024). More details are provided in Appendix E.

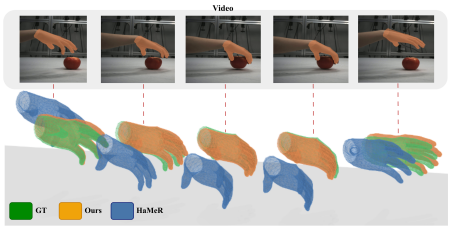

Figure 3: Visualization of hand motion reconstruction.

4.2 Hand Motion Reconstruction Evaluation

Table 1 presents a quantitative comparison between Stage 1 of Geo2Dex and existing human hand reconstruction methods. In particular, WiLoR Potamias et al. (2024), HaMeR Pavlakos et al. (2024) and Dyn-HaMR Yu et al. (2025) are fine-tuned on HRDexDB for a fair comparison. Details of the evaluation metrics are provided in the Appendix F.

Table 1: Quantitative comparison with human hand reconstruction methods on HRDexDB.

Method	Abs-MPJPE ↓	RR-MPJPE ↓	PA-MPJPE ↓	RR-J20 ↑	PA-J20 ↑
WiLoR	51.71	12.84	4.70	45.96%	76.98%
HaMeR	21.11	9.62	3.91	56.34%	80.78%
Dyn-HaMR	17.58	6.03	3.14	70.92%	85.01%
Ours	**8.21**	**4.47**	**2.26**	**78.43%**	**88.92%**

Our method outperforms all competing approaches across all metrics. In particular, it reduces Abs-MPJPE from 17.58 to 8.21 compared with the strongest baseline. This substantial gain is primarily attributed to the 3D spatial geometry representations provided by VGGT, which effectively alleviate depth ambiguity and improve absolute hand localization. The improvements in the other metrics further demonstrate more accurate reconstruction of hand structures and local poses. Fig. 3 presents qualitative visualizations of the hand reconstructions produced by our method.

4.3 Human-to-Robot Retargeting Evaluation

To comprehensively evaluate the quality of the robot trajectories generated by Geo2Dex, we conduct two experiments focusing on kinematic retargeting fidelity and task-level interaction effectiveness. First, we perform an offline trajectory reconstruction evaluation. The wrist position and orientation errors, root-relative fingertip error, and joint-angle error are used to measure the consistency between the generated trajectories and the paired ground-truth robot trajectories in terms of global wrist motion, hand configuration, and joint articulation. Second, we execute the generated trajectories in simulation and evaluate object position and orientation errors, and replay success rates. These metrics assess the object manipulation accuracy, contact stability, and task executability of the generated motions.

Kinematic Retargeting Fidelity. We evaluate the kinematic fidelity of robot trajectories generated by Geo2Dex from human demonstrations. Table 3 compares Geo2Dex

Table 2: Evaluation on DexCap.

EE Pos. (mm)↓	**EE Ori.** (°)↓	**Joint Err.** (°)↓	$\mathbf{E}_{\text{prec}}$ ↓ (cm^2)
79.21	11.98	3.57	6.04



Figure 4: Rendered apple and soap-dispenser grasping sequences. The human-hand motions are ground truth, while the dexterous-hand motions are generated using Geo2Dex.

with Dex-Retargeting Qin et al. (2023), DexCap Wang et al. (2024), and ManipTrans Li et al. (2025a) on Allegro V5 and Inspire DFTP. We report wrist position and orientation errors, root-relative fingertip error, and joint-angle error to assess global wrist motion, local hand configuration, and joint articulation. Geo2Dex achieves the best overall performance across both dexterous-hand embodiments, demonstrating its ability to generate accurate robot trajectories for hands with different kinematic structures. Furthermore, Table 2 presents a broader range of evaluation metrics for our method on the DexCap dataset, demonstrating the superior performance of Geo2Dex across different training datasets. $\mathbf{E}_{\text{prec}}$ measures the spatial discrepancy between the ground-truth and retargeted robot–object contact regions using the Chamfer Distance.

Table 3: Quantitative comparison of robot trajectory reconstruction on Allegro V5 and Inspire DFTP. Results are reported as mean $\pm$ standard deviation.

Embodiment	Method	EE Pos. (mm) $\downarrow$	EE Ori. ($^\circ$) $\downarrow$	Root-rel. Fingertip (mm) $\downarrow$	Joint Error ($^\circ$) $\downarrow$
Allegro V5	Dex-Retargeting	107.07 ± 56.45	21.17 ± 16.01	44.17 ± 54.08	16.03 ± 37.59
	DexCap	69.34 ± 41.60	16.70 ± 11.49	32.34 ± 22.68	9.84 ± 9.28
	ManipTrans	76.72 ± 48.73	18.94 ± 13.84	34.85 ± 31.42	12.91 ± 15.27
	Ours	$\mathbf{51.22 \pm 38.26}$	$\mathbf{13.13 \pm 9.77}$	$\mathbf{29.16 \pm 26.68}$	$\mathbf{6.14 \pm 8.48}$
Inspire DFTP	Dex-Retargeting	131.64 ± 77.62	27.36 ± 16.97	25.25 ± 13.61	21.47 ± 52.26
	DexCap	108.26 ± 53.87	26.18 ± 11.36	$\mathbf{10.81 \pm 8.35}$	13.22 ± 12.97
	ManipTrans	99.37 ± 56.92	27.02 ± 15.80	19.64 ± 14.20	19.56 ± 24.88
	Ours	$\mathbf{84.45 \pm 48.51}$	$\mathbf{22.78 \pm 13.36}$	18.79 ± 15.30	$\mathbf{7.78 \pm 7.16}$

Simulation-based Trajectory Replay. Kinematic accuracy alone does not ensure successful object manipulation. We therefore replay the generated robot trajectories in simulation and evaluate their object-level accuracy and replay success rate. Specifically, since HRDexDB provides scanned 3D object models, together with object 6D pose trajectories, we report the object pose error, and replay success rate, where we consider a trajectory successful if the mean position and rotation errors between the replayed object trajectory and the ground-truth object trajectory satisfy $E_{\text{pos}} < 0.1\,\text{m}$ and $E_{\text{rot}} < 0.5\,\text{rad}$, respectively Paliwal et al. (2026).

For each test sequence, we reconstruct the initial scene in Isaac Sim and execute the generated robot actions. Table 4 presents the physical trajectory errors and task success rates, respectively, of the aforementioned baseline methods and our method during replay. Our method achieves the best object-level accuracy and task success across both hand embodiments. Notably, the overall success rate increases from 48.94% to 69.15%, indicating that improved trajectory generation translates into more reliable physical interaction. As shown in Fig. 4, Geo2Dex retargets human-hand motions to the Inspire and Allegro hands while preserving the overall grasping behavior and object interactions when grasping an apple and a soap.

Table 4: Test-set performance during simulation replay. Object replay metrics are aggregated across both dexterous hands, while task success rates are reported for each embodiment and overall.

Method	Object Replay Metrics		Task Success Rate (%)		
	Pos. RMSE (mm) ↓	Ori. Error (deg) ↓	Allegro V5	Inspire DFTP	Overall
Dex-Retargeting	206.0	66.1	34.04	27.66	30.85
DexCap	103.3	42.7	55.32	42.55	48.94
ManipTrans	149.1	50.3	57.45	38.30	47.87
Ours	**82.5**	**35.6**	**74.47**	**63.83**	**69.15**

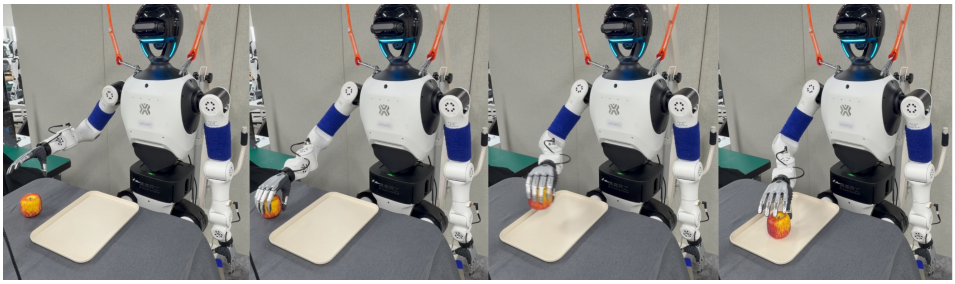

Figure 5: A successful real-world VLA inference trajectory for the apple grasping task.

4.4 Downstream VLA Policy Evaluation

We evaluate the effectiveness of Geo2Dex-generated data for downstream VLA learning with $\pi_{0.5}$ in both simulation and the real world. For each experimental setting, we conduct 10 independent trials. In simulation, we compare three fine-tuning settings on three HRDexDB manipulation tasks: direct fine-tuning with human demonstrations (**Human Data**), robot trajectories generated by Dex-Retargeting (**Dex-Retargeting**), and those generated by Geo2Dex (**Ours**). As shown in Table 5, Geo2Dex-generated data consistently yields the best performance across all tasks and both dexterous hands. Further details are provided in the Appendix G. We further collect paired robot demonstrations for two real-world tasks and compare robot-only fine-tuning (**R**) with mixed training using robot data and human images paired with Geo2Dex-generated robot actions (**H+R**). As shown in Table 6, incorporating Geo2Dex-generated data improves real-world task success. Fig. 5 shows an example of a successful apple-grasping task. Further details are provided in the Appendix H.

Table 5: Success rates (%) under different fine-tuning settings.

Fine-tuning Data	**Hand**	**Apple**	**Box**	**Coffee Tin**
Human Data	Inspire DFTP	30	0	10
	Allegro V5	40	0	30
Dex-Retargeting	Inspire DFTP	40	50	40
	Allegro V5	50	30	40
Ours	Inspire DFTP	**100**	**70**	**70**
	Allegro V5	**100**	**80**	**90**

Table 6: Real-robot task success rates (%). R denotes robot data, and H+R denotes mixed human and robot data.

Data	**Apple**	**Bottle**
R	80	70
H+R	**100**	**100**

5 Conclusion

We present Geo2Dex, a unified framework that translates human demonstration videos into manipulation trajectories for multiple dexterous-hand embodiments. Geo2Dex first leverages 4D visual geometry features to reconstruct accurate 3D human-hand motion. It then encodes the multiple latent codes and uses a slot-based decoder to generate embodiment-specific wrist-pose and joint trajectories. Experiments on HRDexDB demonstrate that our method improves both hand reconstruction accuracy and human-to-robot retargeting fidelity across different robotic hands. Furthermore, the downstream evaluation demonstrates that our method can more effectively convert human demonstrations into useful pretraining data for robot policy learning.

6 AI USE STATEMENT

In this work, we used generative AI tools for **assisting with translation**. We have not used generative AI tools for **helping develop theoretical models or conceptual frameworks, providing critical ingredients for proving mathematical claims and supporting qualitative, proposing or refine hypotheses, designing or providing feedback on research methodology or experiments and supporting qualitative and thematic data analysis, interpret results**, and the rest of the required disclosure tasks are not applicable to this work. Additionally, we used generative AI tools for **editing a research paper to improve readability and creating or editing software code**. We have reviewed all AI-assisted work. We ensure that the AI-modified wording remains consistent with the original meaning and that all code generated or modified by AI has undergone manual review. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Erik Bauer, Elvis Nava, and Robert K. Katzschmann. Latent action diffusion for cross-embodiment manipulation, 2026.

Homanga Bharadhwaj, Debidatta Dwibedi, Abhinav Gupta, Shubham Tulsiani, Carl Doersch, Ted Xiao, Dhruv Shah, Fei Xia, Dorsa Sadigh, and Sean Kirmani. Gen2act: Human video generation in novel scenarios enables generalizable robot manipulation, 2024.

Xiaoyu Chen, Junliang Guo, Tianyu He, Chuheng Zhang, Pushi Zhang, Derek Cathera Yang, Li Zhao, and Jiang Bian. Igor: Image-goal representations are the atomic control units for foundation models in embodied ai, 2024.

Qichen Fu, Xingyu Liu, Ran Xu, Juan Carlos Niebles, and Kris M. Kitani. Deformer: Dynamic fusion transformer for robust hand pose estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 23600–23611, October 2023.

Wendi Hu, Haonan Zhou, Wenhao Hu, and Gaoang Wang. Hand3r: Online 4d hand-scene reconstruction in the wild, 2026.

Yu Hu, Chong Cheng, Sicheng Yu, Xiaoyang Guo, and Hao Wang. Vggt4d: Mining motion cues in visual geometry transformers for 4d scene reconstruction, 2025.

Haoran Jiang, Jin Chen, Qingwen Bu, Li Chen, Modi Shi, Yanjie Zhang, Delong Li, Chuanzhe Suo, Chuang Wang, Zhihui Peng, and Hongyang Li. Wholebodyvla: Towards unified latent vla for whole-body loco-manipulation control, 2025.

Simar Kareer, Dhruv Patel, Ryan Punamiya, Pranay Mathur, Shuo Cheng, Chen Wang, Judy Hoffman, and Danfei Xu. Egomimic: Scaling imitation learning via egocentric video, 2024.

Simar Kareer, Karl Pertsch, James Darpinian, Judy Hoffman, Danfei Xu, Sergey Levine, Chelsea Finn, and Suraj Nair. Emergence of human to robot transfer in vision-language-action models, 2025.

Kailin Li, Puhao Li, Tengyu Liu, Yuyang Li, and Siyuan Huang. Maniptrans: Efficient dexterous bimanual manipulation transfer via residual learning. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025a.

Qixiu Li, Yu Deng, Yaobo Liang, Lin Luo, Lei Zhou, Chengtang Yao, Lingqi Zeng, Zhiyuan Feng, Huizhi Liang, Sicheng Xu, Yizhong Zhang, Xi Chen, Hao Chen, Lily Sun, Dong Chen, Jiaolong Yang, and Baining Guo. Scalable vision-language-action model pretraining for robotic manipulation with real-life human activity videos, 2025b.

Jongbin Lim, Taeyun Ha, Mingi Choi, Jisoo Kim, Byungjun Kim, Subin Jeon, and Hanbyul Joo. Hrdexdb: A paired human-robot dataset for cross-embodiment dexterous grasping, 2026.

Zhao Mandi, Yifan Hou, Dieter Fox, Yashraj Narang, Ajay Mandlekar, and Shuran Song. Dexmachina: Functional retargeting for bimanual dexterous manipulation, 2025.

Maximus A. Pace, Prithwish Dan, Chuanruo Ning, Atiksh Bhardwaj, Audrey Du, Edward W. Duan, Wei-Chiu Ma, and Kushal Kedia. X-diffusion: Training diffusion policies on cross-embodiment human demonstrations. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2026.

Bhawna Paliwal, Haritheja Etukuru, William Liang, Pieter Abbeel, Nur Muhammad Mahi Shafiullah, and Jitendra Malik. Do as i do: Dexterous manipulation data from everyday human videos, 2026.

Georgios Pavlakos, Dandan Shan, Ilija Radosavovic, Angjoo Kanazawa, David Fouhey, and Jitendra Malik. Reconstructing hands in 3D with transformers. In *CVPR*, 2024.

Rolandos Alexandros Potamias, Jinglei Zhang, Jiankang Deng, and Stefanos Zafeiriou. Wilor: End-to-end 3d hand localization and reconstruction in-the-wild, 2024.

Yuzhe Qin, Wei Yang, Binghao Huang, Karl Van Wyk, Hao Su, Xiaolong Wang, Yu-Wei Chao, and Dieter Fox. Anyteleop: A general vision-based dexterous robot arm-hand teleoperation system. In *Robotics: Science and Systems*, 2023.

Jie Ren, Zhehao Jiang, Yinhong Yang, Haorui Jia, Han Jiang, Ben Li, Yao Yao, Cheng Lin, Qiu Shen, Zhenshan Bing, Xiao-Xiao Long, and Xun Cao. C2dex: Contact-consistent reconstruction and retargeting for dexterous manipulation from monocular video. *arXiv preprint arXiv:2608.07045*, 2026.

Javier Romero, Dimitrios Tzionas, and Michael J. Black. Embodied hands: modeling and capturing hands and bodies together. *ACM Transactions on Graphics*, 36(6):1–17, November 2017. ISSN 1557-7368. doi: 10.1145/3130800.3130883.

Chen Wang, Haochen Shi, Weizhuo Wang, Ruohan Zhang, Li Fei-Fei, and C. Karen Liu. Dexcap: Scalable and portable mocap data collection system for dexterous manipulation. *arXiv preprint arXiv:2403.07788*, 2024.

Haonan Wang, Hanyu Zhou, Haoyue Liu, and Luxin Yan. 4d-vggt: A general foundation model with spatiotemporal awareness for dynamic scene geometry estimation, 2025a.

Jianyuan Wang, Minghao Chen, Nikita Karaev, Andrea Vedaldi, Christian Rupprecht, and David Novotny. Vggt: Visual geometry grounded transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2025b.

Sicheng Xie, Haidong Cao, Zejia Weng, Zhen Xing, Haoran Chen, Shiwei Shen, Jiaqi Leng, Zuxuan Wu, and Yu-Gang Jiang. Human2robot: Learning robot actions from paired human-robot videos, 2025.

Sirui Xu, Yu-Wei Chao, Liuyu Bian, Arsalan Mousavian, Yu-Xiong Wang, Liang-Yan Gui, and Wei Yang. Dexplore: Scalable neural control for dexterous manipulation from reference-scoped exploration, 2025.

Yuxue Yang, Lue Fan, Ziqi Shi, Junran Peng, Feng Wang, and Zhaoxiang Zhang. Neoverse: Enhancing 4d world model with in-the-wild monocular videos. *arXiv preprint arXiv:2601.00393*, 2026.

Zhao-Heng Yin, Changhao Wang, Luis Pineda, Krishna Bodduluri, Tingfan Wu, Pieter Abbeel, and Mustafa Mukadam. Geometric retargeting: A principled, ultrafast neural hand retargeting algorithm, 2025.

Zhengdi Yu, Stefanos Zafeiriou, and Tolga Birdal. Dyn-hamr: Recovering 4d interacting hand motion from a dynamic camera. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2025.

Jinglei Zhang, Jiankang Deng, Chao Ma, and Rolandos Alexandros Potamias. Hawor: World-space hand motion reconstruction from egocentric videos. *arXiv preprint arXiv:2501.02973*, 2025.

Ruijie Zheng, Dantong Niu, Yuqi Xie, Jing Wang, Mengda Xu, Yunfan Jiang, Fernando Castañeda, Fengyuan Hu, You Liang Tan, Letian Fu, Trevor Darrell, Furong Huang, Yuke Zhu, Danfei Xu, and Linxi Fan. Egoscale: Scaling dexterous manipulation with diverse egocentric human data, 2026.

A Robot Contact Detection for Temporal Alignment

For the robot sequence, we determine the contact interval based on the geometric proximity between the robot and the manipulated object. Specifically, we use forward kinematics to compute K robot interaction points, including the wrist and fingertips, from the robot joint configuration. Let $\mathbf{p}_{t,j}^R$ denote the position of the j-th interaction point at frame t, and let $\mathcal{V}_t^O$ denote the object surface transformed according to its pose at frame t. We define the contact state of each interaction point as

$$c_{t,j} = \mathbb{I}\left[\min_{\mathbf{v}\in\mathcal{V}_t^O} \left\|\mathbf{p}_{t,j}^R - \mathbf{v}\right\|_2 < \epsilon\right], \tag{12}$$

where ϵ is a predefined distance threshold.

The first and last frames containing at least one contacting interaction point define the robot contact interval:

$$t_{\text{contact}}^R = \min\left\{t : \sum_{j=1}^{K} c_{t,j} > 0\right\}, \qquad t_{\text{release}}^R = \max\left\{t : \sum_{j=1}^{K} c_{t,j} > 0\right\}. \tag{13}$$

The sequence boundaries define the remaining robot-side anchors, $t_{\text{start}}^R = 1$ and $t_{\text{end}}^R = T_R$.

Notably, our four-anchor temporal alignment assumes a single continuous interaction stage. For multi-stage tasks with multiple contact transitions, the demonstration should first be segmented into interaction-consistent subtasks using an external task-segmentation method, after which our alignment is applied to each segment independently.

B Multimodal Spatiotemporal Context Encoding

Given the structured interaction context $\mathcal{C}_t = (\mathcal{C}_t^H, \mathcal{C}_t^S, \mathcal{C}_t^O)$, we independently project its human-hand, scene, and object components into a common D-dimensional token space. Specifically, the tokens at frame t are constructed as

$$\begin{aligned}
\mathbf{x}_{t,j}^J &= f_J\Big(\Big[\hat{\mathbf{J}}_{t,j}^H, \hat{\mathbf{J}}_{t,j}^H - \hat{\mathbf{J}}_{t,\text{wrist}}^H, \Delta\hat{\mathbf{J}}_{t,j}^H\Big]\Big) + \mathbf{e}_j^{\text{joint}} + \mathbf{e}_J^{\text{type}} + \mathbf{e}_t^{\text{time}} \\
\mathbf{x}_t^H &= f_H\big(\mathbf{z}_t^H\big) + \mathbf{e}_H^{\text{type}} + \mathbf{e}_t^{\text{time}}, \qquad \mathbf{x}_t^S = f_S\big(\mathbf{z}_t^V\big) + \mathbf{e}_S^{\text{type}} + \mathbf{e}_t^{\text{time}} \\
\mathbf{x}_t^O &= f_O\big(\boldsymbol{\xi}\big(\mathbf{T}_t^O\big)\big) + \mathbf{e}_O^{\text{type}} + \mathbf{e}_t^{\text{time}}
\end{aligned} \tag{14}$$

where $\Delta\hat{\mathbf{J}}_{t,j}^H$ denotes the temporal velocity of the j-th human-hand joint and $\boldsymbol{\xi}(\cdot)$ vectorizes the object pose. The function $f_\cdot$ is a modality-specific projection network. The joint-identity embedding $\mathbf{e}_j^{\text{joint}}$, modality-type embedding $\mathbf{e}^{\text{type}}$, and temporal embedding $\mathbf{e}_t^{\text{time}}$ preserve the structural, semantic, and temporal identities of the tokens, respectively.

The robot state $s_t^{R,g}$ is encoded in the same token space as additional policy-state tokens:

$$\mathbf{x}_t^R = f_R\Big(\Big[\mathbf{s}_t^{R,g}, \Delta\mathbf{s}_t^{R,g}\Big]\Big) + \mathbf{e}_R^{\text{type}} + \mathbf{e}_t^{\text{time}} \tag{15}$$

The complete token set at frame t is $\mathcal{X}_t = \Big(\big\{\mathbf{x}_{t,j}^J\big\}_{j=1}^{N_H}, \mathbf{x}_t^H, \mathbf{x}_t^S, \mathbf{x}_t^O, \mathbf{x}_t^R\Big)$. Importantly, the different modalities are retained as separate tokens rather than concatenated into a single feature vector.

C Robot Hand Construction

We represent the target embodiment g by

$$\mathcal{E}^g = \Big(g, \big\{\mathbf{a}_j^g\big\}_{j=1}^{N_{max}}, \big\{m_j^g\big\}_{j=1}^{N_{max}}\Big), \quad N_{\max} = \max_{g\in\mathcal{G}} N_g, \quad m_j^g = \begin{cases} 1, & j \le N_g, \\ 0, & j > N_g, \end{cases} \tag{16}$$

we associate its j-th actuator with a time-invariant structural descriptor $\mathbf{a}_j^g$. In our implementation, $\mathbf{a}_j^g$ is stored as the j-th joint attribute and is deterministically constructed from the actuator index,

its digit membership, its within-digit position or actuator type, and the embodiment type. It does not represent the instantaneous joint position or the target action value.

Let $\iota_j = (j-1)/(N_g-1)$ denote the normalized actuator index. For the Allegro V5 hand, the descriptor is defined by

$$\begin{aligned} a^g_{j,1} &= \iota_j, \\ (a^g_{j,2}, \dots, a^g_{j,5}) &= \mathbf{e}^{(4)}_{\lfloor (j-1)/4 \rfloor + 1}, \\ (a^g_{j,7}, \dots, a^g_{j,10}) &= \mathbf{e}^{(4)}_{(j-1) \bmod 4 + 1}, \\ a^g_{j,13} &= 1, \end{aligned}$$

where $\mathbf{e}^{(d)}_k$ denotes the k-th basis vector in $\mathbb{R}^d$, and all unspecified dimensions are set to zero. Thus, the four groups of four actuators are assigned to the four digits, while the second one-hot code identifies the actuator's relative position within its digit.

For the Inspire DFTP hand, the six actuators are assigned to digits according to $(1, 1, 2, 3, 4, 5)$, where the first two actuators belong to the thumb. Its descriptor is defined as

$$\begin{aligned} a^g_{j,1} &= \iota_j, \\ (a^g_{j,2}, \dots, a^g_{j,6}) &= \mathbf{e}^{(5)}_{f(j)}, \qquad f = (1, 1, 2, 3, 4, 5), \\ (a^g_{j,7}, a^g_{j,8}) &= \begin{cases} (0,1), & j = 2, \\ (1,0), & j \neq 2, \end{cases} \\ a^g_{j,14} &= 1, \end{aligned}$$

with the remaining dimensions equal to zero. Here, the two-dimensional code distinguishes the thumb-rotation actuator from the flexion actuators.

D Detailed Training Objectives

D.1 Stage 1

We train the hand reconstruction module using both geometric and MANO-parameter supervision. The overall objective is defined as

$$\mathcal{L}_{\text{stage1}} = \lambda_1 \mathcal{L}_{\text{geo}} + \lambda_2 \mathcal{L}_{\text{mano}} + \lambda_3 \mathcal{L}_{\text{pose-reg}}. \tag{17}$$

Here, the geometric loss is defined as

$$\mathcal{L}_{\text{geo}} = \mathcal{L}^{\text{rel}}_{\text{joint}} + \mathcal{L}^{\text{abs}}_{\text{joint}} + \mathcal{L}^{\text{rel}}_{\text{vert}} + \mathcal{L}^{\text{abs}}_{\text{vert}}, \tag{18}$$

where the root-relative joint and vertex losses supervise hand articulation and mesh geometry independently of global translation, while the absolute-coordinate losses constrain the global hand position in the camera coordinate system. The MANO parameter loss is defined as

$$\mathcal{L}_{\text{mano}} = \mathcal{L}_{\text{pose}} + \mathcal{L}_{\text{shape}} + \mathcal{L}_{\text{trans}}, \tag{19}$$

which directly supervises the predicted MANO pose, shape, and translation parameters. The pose-regularization loss $\mathcal{L}_{\text{pose-reg}}$ further regularizes the predicted hand pose.

D.2 Stage 2

We provide the detailed definitions of the loss terms of Stage 2 training, where

$$\mathcal{L}_{\text{action}} = \mathcal{L}_{\text{pos}} + \mathcal{L}_{\text{rot}} + \mathcal{L}_{\text{qpos}}. \tag{20}$$

Wrist Pose Losses. The wrist-translation and rotation losses are defined as

$$\mathcal{L}_{\text{pos}} = \frac{1}{Z_H} \sum_{h=0}^{H-1} u_{t+h}\, \rho\big(\widehat{\mathbf{x}}^g_{t+h}, \mathbf{x}^g_{t+h}\big)\,, \tag{21}$$

$$\mathcal{L}_{\text{rot}} = \frac{1}{Z_H} \sum_{h=0}^{H-1} u_{t+h}\, d_{\text{geo}}\big(\mathcal{R}(\widehat{\mathbf{r}}^g_{t+h}), \mathcal{R}(\mathbf{r}^g_{t+h})\big)\,, \tag{22}$$

where ρ denotes the smooth-ℓ_1 loss, $\mathcal{R}(\cdot)$ converts a 6D rotation representation to a rotation matrix, d_{geo} is the geodesic distance on $\mathrm{SO}(3)$, and u_{t+h} indicates whether the corresponding future action is valid.

Joint-Position Loss. The joint-position loss is masked by both the temporal action mask and the embodiment-specific joint mask:

$$\mathcal{L}_{\text{qpos}} = \frac{1}{Z_Q} \sum_{h=0}^{H-1} \sum_{j=1}^{N_{\max}} u_{t+h} m_j^g \, \rho\Big(\widehat{q}_{t+h,j}^{\,g}, q_{t+h,j}^{g}\Big), \tag{23}$$

where

$$Z_H = \sum_h u_{t+h}, \qquad Z_Q = \sum_{h,j} u_{t+h} m_j^g \tag{24}$$

are the corresponding normalization factors. Therefore, padded joints do not contribute to the optimization.

First-Order Motion Loss. To encourage temporally coherent action chunks, we additionally supervise the first-order motion:

$$\mathcal{L}_{\text{motion}} = \frac{1}{Z_\Delta} \sum_{h=1}^{H-1} u_{t+h} u_{t+h-1} \Bigg[\left\| \Delta \widehat{\mathbf{x}}_{t+h}^{g} - \Delta \mathbf{x}_{t+h}^{g} \right\|_1 + \frac{\sum_{j=1}^{N_g} m_j^g \left| \Delta \widehat{q}_{t+h,j}^{\,g} - \Delta q_{t+h,j}^{g} \right|}{\sum_{j=1}^{N_g} m_j^g} \Bigg], \tag{25}$$

where

$$\Delta \mathbf{x}_{t+h} = \mathbf{x}_{t+h} - \mathbf{x}_{t+h-1}, \qquad \Delta q_{t+h,j} = q_{t+h,j} - q_{t+h-1,j}. \tag{26}$$

This term discourages static predictions and improves the temporal consistency of the predicted wrist and joint trajectories.

E Implementation Details

E.1 SGHR Feature selection

We employ the 4DVGGT Visual Geometry Transformer to extract visual features and camera tokens via interleaved frame-wise and global attention. Instead of relying solely on the final-layer, we collect features from the 5th, 12th, 18th, and 24th Transformer blocks Hu et al. (2025). At each level, frame-wise and global features are concatenated, independently projected, and aggregated across levels to form a hierarchical visual context capturing local appearance and global geometry. Camera tokens extracted from the 24th block are also projected into the hand feature space. We initialize the backbone with pretrained 4DVGGT weights, while only the last two frame-wise and global attention blocks are fine-tuned Yang et al. (2026).

E.2 Training Details

For Stage 1, we sample every third frame during training. The hand decoder uses 32 learnable queries and four self-/cross-attention blocks to aggregate visual tokens. We jointly optimize the decoder and the last two blocks of the pretrained visual geometry encoder using AdamW, with learning rates of 5×10^{-5}, respectively, while freezing the remaining backbone. For Stage 2, the aligned trajectories are segmented into 8-frame observation windows with 16-step action horizons. The interaction encoder, VideoFormer, and slot trajectory decoder comprise four, three, and four Transformer layers, respectively, with a hidden dimension of 512. We first train the complete retargeting model via behavior cloning for 10000 iterations using AdamW with a learning rate of

10^{-4}. The two robotic embodiments are sampled equally, with their wrist-pose and joint-position targets normalized separately. To reduce error accumulation during open-loop deployment, we subsequently perform recursive rollout fine-tuning for 1,000 iterations over 40 consecutive windows. During this stage, only the qpos prediction modules are updated with a learning rate of 10^{-5}.

E.3 DexCap Data Preprocessing

DexCap employs a bimanual robotic embodiment consisting of two Franka Emika arms, each equipped with a 16-DoF LEAP dexterous hand. Since DexCap does not provide object meshes or pose annotations, we extract object point clouds from its depth observations. Specifically, we annotate target objects in 2D and back-project the corresponding depth pixels into 3D. We then combine the resulting object point clouds with robot geometry obtained through forward kinematics to determine robot–object contact regions in 3D space.

F Hand Reconstruction Metrics

We evaluate hand reconstruction using the 21 MANO joints, including the wrist joint. Let $\hat{\mathbf{x}}_{t,j} \in \mathbb{R}^3$ and $\mathbf{x}_{t,j} \in \mathbb{R}^3$ denote the predicted and ground-truth positions of joint j at frame t, respectively. All positional errors are reported in millimeters and averaged over all evaluated frame–joint pairs.

Absolute MPJPE (Abs-MPJPE). Abs-MPJPE measures the mean Euclidean joint error directly in the common camera/world coordinate system:

$$\text{Abs-MPJPE} = \frac{1}{TK} \sum_{t=1}^{T} \sum_{j=1}^{K} \|\hat{\mathbf{x}}_{t,j} - \mathbf{x}_{t,j}\|_2 , \tag{27}$$

where $K = 21$. This metric evaluates both the reconstructed hand pose and its absolute 3D location, and is therefore sensitive to global translation and scale errors.

Root-Relative MPJPE (RR-MPJPE). For root-relative evaluation, the wrist joint, indexed by r, is independently subtracted from the predicted and ground-truth joints:

$$\hat{\mathbf{x}}^{\text{RR}}_{t,j} = \hat{\mathbf{x}}_{t,j} - \hat{\mathbf{x}}_{t,r}, \qquad \mathbf{x}^{\text{RR}}_{t,j} = \mathbf{x}_{t,j} - \mathbf{x}_{t,r}. \tag{28}$$

RR-MPJPE is then defined as

$$\text{RR-MPJPE} = \frac{1}{TK} \sum_{t=1}^{T} \sum_{j=1}^{K} \left\|\hat{\mathbf{x}}^{\text{RR}}_{t,j} - \mathbf{x}^{\text{RR}}_{t,j}\right\|_2 . \tag{29}$$

It removes the error in global hand translation while retaining errors in hand orientation, scale, and articulation.

Procrustes-Aligned MPJPE (PA-MPJPE). PA-MPJPE is computed after independently aligning the predicted joints to the ground truth in each frame using a similarity transformation:

$$(s^*_t, \mathbf{R}^*_t, \mathbf{t}^*_t) = \arg\min_{s,\mathbf{R},\mathbf{t}} \sum_{j=1}^{K} \|s\mathbf{R}\hat{\mathbf{x}}_{t,j} + \mathbf{t} - \mathbf{x}_{t,j}\|_2^2 , \tag{30}$$

where s is a uniform scale, $\mathbf{R} \in \text{SO}(3)$ is a rotation, and $\mathbf{t}$ is a translation. The metric is

$$\text{PA-MPJPE} = \frac{1}{TK} \sum_{t=1}^{T} \sum_{j=1}^{K} \|s^*_t\mathbf{R}^*_t\hat{\mathbf{x}}_{t,j} + \mathbf{t}^*_t - \mathbf{x}_{t,j}\|_2 . \tag{31}$$

By removing global translation, rotation, and scale discrepancies, PA-MPJPE mainly reflects errors in the articulated hand configuration and relative joint structure.

RR-J20 and PA-J20. J20 denotes the normalized area under the Percentage of Correct Keypoints (PCK) curve over joint-error thresholds from 0 to 20 mm. Given per-joint errors $\{e_i\}_{i=1}^{M}$, we define

$$\mathrm{PCK}(\tau) = \frac{1}{M}\sum_{i=1}^{M}\mathbb{I}[e_i \le \tau], \tag{32}$$

and

$$\mathrm{J20} = \frac{1}{20}\int_0^{20}\mathrm{PCK}(\tau)\,d\tau = \frac{1}{M}\sum_{i=1}^{M}\max\left(0, 1-\frac{e_i}{20}\right). \tag{33}$$

RR-J20 uses the root-relative joint errors employed by RR-MPJPE, whereas PA-J20 uses the Procrustes-aligned errors employed by PA-MPJPE. J20 ranges from 0 to 1 and assigns larger scores to predictions whose joint errors are consistently small, rather than only counting whether they fall below a single threshold. Thus, lower values are better for the three MPJPE metrics, while higher values are better for RR-J20 and PA-J20.

G Downstream VLA Policy Fine-tuning in Simulator

To evaluate how different sources of manipulation data affect downstream vision-language-action (VLA) policy learning, we fine-tune a pretrained $\pi_{0.5}$ model using three data settings: Human Data, robot data generated by Dex-Retargeting, and robot data generated by our Geo2Dex method. Experiments are conducted with two dexterous-hand embodiments, Inspire DFTP and Allegro V5, across three grasping scenarios: Apple, Box, and Coffee Tin. For each embodiment, an independent policy is trained under each data setting using the same model initialization and training budget. Fig. 6 presents qualitative results from open-loop inference.

G.1 Policy Inputs and Action Representation

At timestep t, the VLA policy takes an egocentric RGB observation $\mathbf{o}_t^{\mathrm{img}}$, the current joint state $\mathbf{o}_t^{\mathrm{state}}$, and a language instruction ℓ as input, and predicts a sequence of future actions:

$$\pi_\theta\left(\mathbf{o}_t^{\mathrm{img}}, \mathbf{o}_t^{\mathrm{state}}, \ell\right) \to \mathbf{a}_{t:t+H-1}. \tag{34}$$

The action horizon is set to $H = 10$. All trajectories are temporally synchronized and sampled at 5 Hz, while the RGB observations are resized to 320×240. A single policy is trained jointly on all three grasping tasks, with the language instruction specifying the target object.

G.2 Fine-tuning with Human Data

Under the **Human Data** setting, the policy is trained using egocentric human manipulation demonstrations. Each trajectory contains synchronized RGB observations, human hand joint states, and a task instruction. To convert human hand motions into the action space of the target dexterous hand, we align the fingertip positions of the human hand and the dexterous hand through forward and inverse kinematics. Specifically, we first compute the 3D fingertip positions of the human hand using forward kinematics, and then solve for the target dexterous-hand joint configuration via inverse kinematics such that its fingertips are aligned with the corresponding human fingertips. The resulting dexterous-hand joint sequence is used as action supervision.

Specifically, let $\mathbf{q}_t^{\mathrm{dex}}$ denote the dexterous-hand joint configuration obtained after fingertip alignment. The action target at timestep t is defined as the subsequent H dexterous-hand joint configurations:

$$\mathbf{a}_t^{\mathrm{human}} = \left[\mathbf{q}_{t+1}^{\mathrm{dex}}, \mathbf{q}_{t+2}^{\mathrm{dex}}, \ldots, \mathbf{q}_{t+H}^{\mathrm{dex}}\right].$$

We use Human Data as a baseline to evaluate the effectiveness of fine-tuning the downstream policy with human demonstrations retargeted through kinematic fingertip alignment.

G.3 Fine-tuning with Generated Robot Data

The robot training trajectories are generated by two different methods: Dex-Retargeting and Geo2Dex. These trajectories are not generated by the simulator. Instead, each method first produces joint-space trajectories for the target dexterous hand from the corresponding human manipulation data. The generated trajectories are subsequently replayed in Isaac Sim to obtain synchronized egocentric RGB observations and robot states. Isaac Sim is therefore used only for trajectory replay, visual observation rendering, and downstream policy evaluation.

For a robot data generation method $m \in \{\text{Dex-Retargeting}, \text{Geo2Dex}\}$, the robot state at timestep t is represented as

$$\mathbf{o}_t^{\text{state},m} = \left[\mathbf{q}_t^{\text{arm},m}, \mathbf{q}_t^{\text{hand},m}\right]. \tag{35}$$

The corresponding action target is constructed from the future robot joint configurations generated by method m:

$$\mathbf{a}_t^m = \left[\mathbf{q}_{t+1}^m, \mathbf{q}_{t+2}^m, \ldots, \mathbf{q}_{t+H}^m\right]. \tag{36}$$

For Inspire DFTP, the state and action spaces are both 12-dimensional, consisting of six arm joints and six hand joints. For Allegro V5, they are both 22-dimensional, consisting of six arm joints and sixteen hand joints.

The Dex-Retargeting and Geo2Dex datasets use identical object models, camera parameters, trajectory sampling rates, and language instructions. Consequently, the principal difference between the two settings lies in the method used to generate the robot trajectories. This controlled setup allows us to directly examine how the quality of generated robot data affects downstream VLA policy learning.

G.4 Fine-tuning Configuration

All policies are initialized from the same pretrained $\pi_{0.5}$ checkpoint and independently fine-tuned using LoRA. LoRA adapters are applied to both the 2B-parameter PaliGemma vision-language backbone and the 300M-parameter action expert, while the remaining pretrained parameters are kept frozen.

Following the commonly adopted downstream fine-tuning setting for OpenPI models, each policy is trained for 30,000 optimization steps with a batch size of 2. Training is distributed across two devices using fully sharded data parallelism, and exponential moving averaging is disabled. State and action normalization statistics are computed independently for every combination of hand embodiment and training data setting to account for differences in action dimensionality and joint distributions.

H Downstream VLA Policy Fine-tuning in Real-world

In this section, we provide details of the real-world VLA policy evaluation experiments, including data collection and mixed training with human and robot data. Notably, for the human-only setting, we fine-tune $\pi_{0.5}$ using the same experimental configuration as in the simulation experiments.

H.1 Data Collection

To collect paired human manipulation and robot data in the real world, we equipped the setup with a Tien Kung 3.0 humanoid robot equipped with a BrainCo Revo 2 bionic dexterous hand, and two Orbbec Gemini 336 stereo vision depth cameras. Specifically, we set up two identical workspaces, Pick up an apple and a bottle and place them on the tray, for the human operator and the robot, placing two cameras in each workspace. The two Orbbec Gemini 336 stereo vision depth cameras are suitably positioned to simultaneously capture the human and robot motions. During data collection, we first teleoperated the robot and recorded temporal anchors for subsequent alignment. We then collected human manipulation demonstrations, ensuring that the hand trajectories and grasping poses are similar to those in the paired robot demonstrations. For each task, we collected 50 paired demonstrations. Fig. 7 shows images from one collected trajectory. Moreover, the object pose annotations are generated using SAM3D.

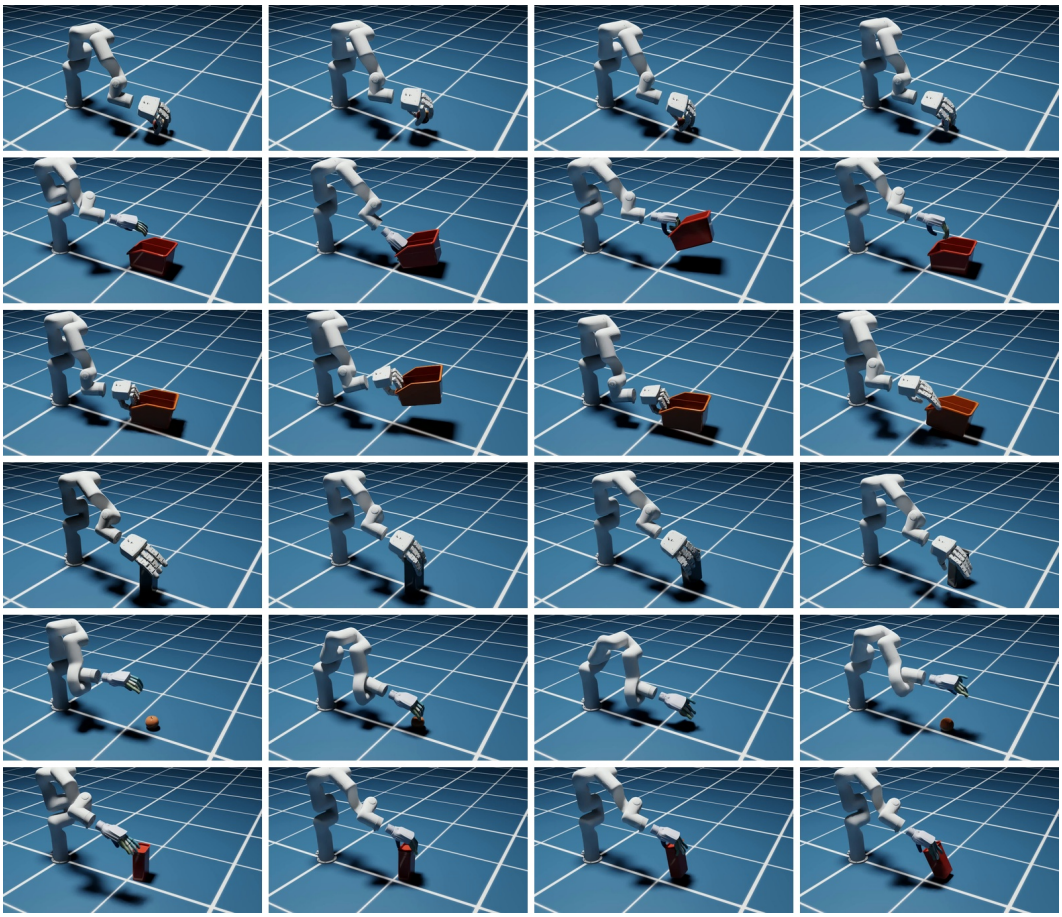

Figure 6: Open-loop rollout visualization of $\pi_0.5$ fine-tuned with Geo2Dex-retargeted actions. Each row shows four representative frames from an open-loop rollout of $\pi_0.5$ on a manipulation task. The policy is fine-tuned using robot actions retargeted from human demonstrations by Geo2Dex.

H.2 Mix-data Fine-tuning

Following the data mixing strategy of $\pi_{0.5}$ Kareer et al. (2025), we fine-tune the policy using both robot demonstrations and human videos. Robot demonstrations are directly represented as observation-action pairs in the robot action space. For each human video, we first extract the human hand motion and then use our Geo2Dex method to retarget it to the target robot. This produces a robot-space action sequence with the same action representation and temporal horizon as the robot demonstrations.

After retargeting, the human and robot data are processed using the same temporal resampling, action normalization, and action-chunk construction procedures. During fine-tuning, batches are sampled from the two datasets according to a fixed mixing ratio. Robot data provide embodiment-specific supervision and anchor the policy to the real robot, while Geo2Dex-retargeted human data provide additional variations in object interaction and manipulation behavior. Geo2Dex is only used to convert human actions into robot actions during data preprocessing and is not required during inference.

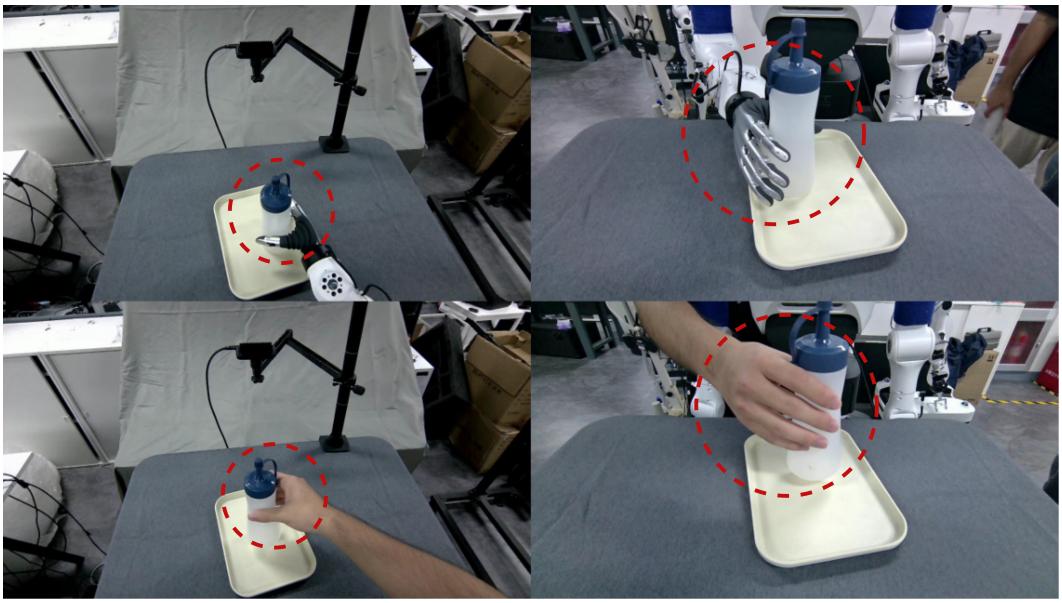

Figure 7: Multi-view comparison of human demonstration and robot execution during the bottle-picking task. The top row shows the robot execution process, while the bottom row shows the human demonstration.

I Ablation Study

I.1 Ablation of the Representation

To assess the importance of the visual representation, we replace the VGGT global feature in our framework with a frozen DINOv3 embedding extracted from the same RGB frames. All remaining inputs, including the hand features, object observations, and robot proprioception, as well as the downstream policy and training protocol, are kept unchanged. This experiment therefore provides a controlled comparison between DINOv3-based appearance features and VGGT-based geometry-aware features.

Table 7: Ablation of the visual representation on Allegro V5 and Inspire DFTP. Only mean errors are reported, and lower is better.

Hand	Visual feature	EE Pos. (mm) ↓	EE Ori. (°) ↓	Root-rel. Fingertip (mm) ↓	Joint Error (°) ↓
Allegro V5	DINOv3	85.93	20.06	37.93	7.40
Allegro V5	VGGT (Ours)	**51.22**	**13.13**	**29.16**	**6.14**
Inspire DFTP	DINOv3	134.32	25.29	31.54	13.46
Inspire DFTP	VGGT (Ours)	**84.45**	**22.78**	**18.79**	**7.78**

As shown in Table 7, the VGGT-based representation consistently outperforms the DINOv3-based alternative across both dexterous hands and all evaluation metrics. On Allegro V5, it reduces the end-effector position, end-effector orientation, root-relative fingertip, and joint-angle errors by 40.4%, 34.5%, 23.1%, and 17.0%, respectively. On Inspire DFTP, the corresponding reductions are 37.1%, 9.9%, 40.4%, and 42.2%. The consistent improvements across two hands with different kinematic structures indicate that the benefit is not specific to a single robot embodiment.

DINOv3 primarily provides strong semantic and appearance representations from individual RGB observations. In contrast, VGGT is designed to encode geometric relationships across frames, providing stronger cues about scene structure and the relative configuration of the human hand, manipulated object, and surrounding environment. These geometry-aware features are particularly useful for transferring human demonstrations to robot end-effector trajectories and dexterous-hand configurations. The results therefore show that, under our task setting, VGGT preserves more task-relevant

geometric and interaction cues than the DINOv3 representation, leading to more accurate robot motion prediction.

I.2 Ablation on Object Context

In this section, we investigate the impact of object latent codes on model trajectory retargeting accuracy. Under the same training configuration, we compare the model performance with and without object contexts. Table 8 reports the results on Allegro V5 and Inspire DFTP.

Overall, the model achieves reasonably strong relocalization performance on both robotic hands even without object contexts. Incorporating object contexts further improves the performance across all evaluation metrics.

Table 8: Ablation on object context on Allegro V5 and Inspire DFTP.

Hand	Object context	EE Pos. (mm) ↓	EE Ori. (°) ↓	Root-rel. Fingertip (mm) ↓	Joint Error (°) ↓
Allegro V5	w/o. object context	63.86	14.20	30.74	7.33
Allegro V5	w/. object context	**51.22**	**13.13**	**29.16**	**6.14**
Inspire DFTP	w/o. object context	96.51	24.08	19.76	8.52
Inspire DFTP	w/. object context	**84.45**	**22.78**	**18.79**	**7.78**