EgoStudy: How Does Scaling Egocentric Data Affect VLA and WAM Pretraining?

Supplementary Material

A. Models and Optimization

A.1. Model Instantiations

Unless a backbone ablation specifies otherwise, the VLA uses StarVLA [1] with Qwen3-VL-4B-Instruct and a GR00T-style action diffusion transformer [2]. The action expert conditions on the last-layer vision–language tokens and a geometric state vector. The model receives the available camera views as separate images and predicts an action chunk using flow matching.

The default WAM uses the joint variant of Fast-WAM [3] with a pretrained Wan2.1-I2V-14B-480P video backbone. Its action transformer has 40 blocks and a hidden width of 512. The video-backbone ablation replaces Wan2.1 with Wan2.2-TI2V-5B. The action tokens attend to the video latent tokens, and training supervises both video and action generation from the initial camera observation, language instruction, and geometric state. The WAM packs the main view and two wrist views into a 576×256 canvas. It uses blank tiles for unavailable wrist views. The reference configuration uses nine video frames, including the conditioning frame, aligned with the action timeline.

A.2. Shared Action and State

During pretraining, both models predict $H = 32$ actions. For each hand, an action contains the pose of the wrist and the poses of the thumb, index, middle, ring, and little fingertips. Each pose uses a position in meters and a 6D rotation representation [4] formed by the first two columns of its rotation matrix, yielding $2 \times 6 \times 9 = 108$ coordinates per action. This geometry lets the models share action targets across human and robot embodiments. Figure 1 illustrates the wrist and fingertip coordinate conventions.

Let $(p_w(t), R_w(t))$ and $(p_f(t), R_f(t))$ denote the world poses of a wrist and one of its fingertips. We denote the anchor time by 0 and the rotation from initial camera axes to world axes by $R_{c,0}$. The wrist targets at time t are

$$\Delta p_w(t) = R_{c,0}^{\top}\big(p_w(t) - p_w(0)\big), \tag{1}$$

$$\Delta R_w(t) = R_{c,0}^{\top} R_w(t) R_w(0)^{\top} R_{c,0}. \tag{2}$$

The fingertip targets at that time are

$$p_{f|w}(t) = R_w(t)^{\top}\big(p_f(t) - p_w(t)\big), \tag{3}$$

$$R_{f|w}(t) = R_w(t)^{\top} R_f(t). \tag{4}$$

The 117-dimensional input state combines the current camera pose, the two current wrist poses expressed in camera coordinates, and the ten current fingertip poses in their corresponding wrist frames.

A.3. Optimization

Both models use masked action flow matching and AdamW with a cosine learning-rate schedule. The reference pretraining recipe uses a learning rate of 10^{-4}, weight decay of 10^{-2}, and 20,000 updates. The reference WAM assigns unit weights to the video and action flow-matching losses. We match the batch and update budgets for data comparisons and keep each family's architecture and initialization fixed within its ablations.

B. Data Curation and Sampling

B.1. Sources and Episode Format

The public human sources are EgoDex [5], HOI4D [6], HOT3D [7], and VITRA [8]. We supplement these with self-collected egocentric demonstrations. From VITRA we retain Ego4D [9], EPIC-KITCHENS [10], and Something-Something V2 [11].

Bimanual XHand supplies tabletop demonstrations such as block assembly, organizing objects on a desk, placing objects into storage, and picking and placing fruit. It also includes multi-step manipulation, tasks involving marked blocks, appliance operation, and other activities that require bimanual manipulation. RoboCasa-GR1 contributes a second robot embodiment and simulated kitchen trajectories [12].

Table 1 lists the durations of the filtered sources and breaks down the VITRA subtotal. We sum the durations of retained clips from their frame counts and frame rates before applying the study's splits and human data scaling.

We convert the sources to a common LeRobot episode schema. Each episode stores a caption and synchronized RGB frames, camera poses, and hand poses at 20 Hz. Human pose tracks come from annotations or hand reconstruction. Robot pose tracks come from offline forward kinematics using recorded joint states and the corresponding robot kinematic models. The conversion aligns camera and hand axes before writing poses to a common world-coordinate convention. When retiming pose tracks, we interpolate translations linearly and rotations with spherical linear interpolation, and match images to the sampled timestamps.

We retain manipulation clips with usable video and pose tracks and represent partial annotation coverage with coordinate-level validity masks. The masks track the inputs

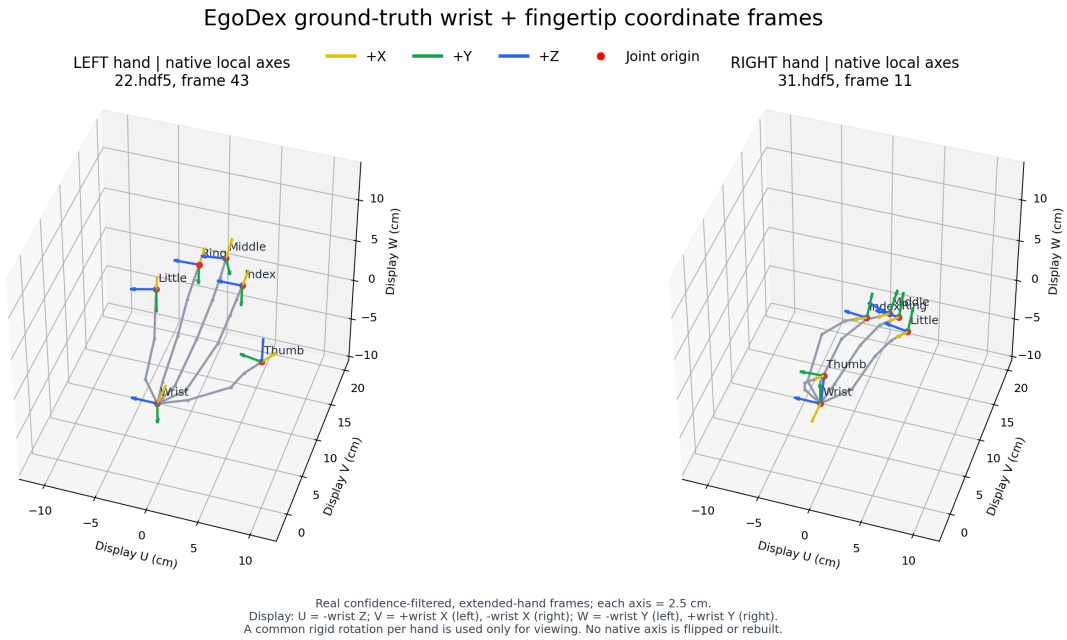

Figure 1. **Shared wrist and fingertip coordinate frames.** Across all datasets, the local coordinate axes of both the wrist and fingertips follow the EgoDex convention. Red points mark frame origins; yellow, green, and blue arrows indicate the local $+X$, $+Y$, and $+Z$ axes, respectively. In the sample images from the dataset, we omit two axes for the sake of display clarity.

Table 1. Post-filtering duration of the pretraining data. The indented rows partition the VITRA subtotal.

Category	Dataset	Hours
Egocentric	EgoDex	633.831
	VITRA (subtotal)	347.217
	Ego4D	272.855
	EPIC-KITCHENS	55.808
	Something-Something V2	18.554
	HOT3D	2.936
	HOI4D	1.525
	Self-collected	867.414
Robot	Bimanual XHand	272.676
	RoboCasa-GR1	80.837
Total		**2,206.436**

to each geometric transform. For example, wrist displacement depends on the reference wrist and camera orientation, while a wrist-local fingertip pose depends on the wrist at the target time. We exclude invalid and padded targets from the action objective. For each source family, affine action normalization uses a fixed, seeded sample of 1% of the full processed episode pool. The statistics pool includes validation and holdout episodes, and the resulting normalizers are shared across data fractions and evaluation splits.

B.2. Episode Segmentation and Captioning

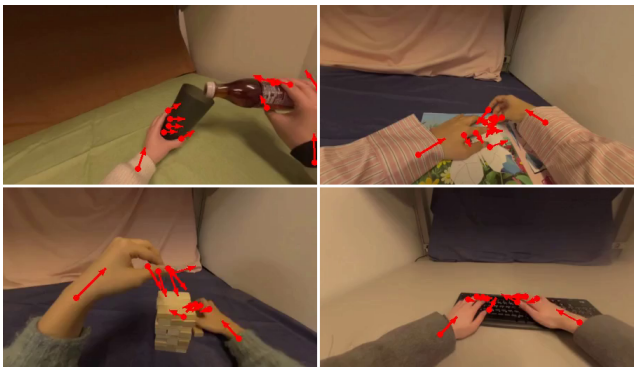

Figure 2. **EgoDex examples.** Red points and arrows denote ground-truth positions and orientations in the shared action space.

EgoDex. Each source RGB video is paired with its ARKit pose file. For temporal segmentation, the video input to the vision–language model is sampled at 4 Hz over at most the first 30 seconds with aligned poses. The model proposes chronological, nonoverlapping clips that cover complete manipulation actions and skip idle motion. The target duration is 3 seconds. A parser clips the proposed boundaries to the available aligned interval and discards clips with empty captions, overlapping intervals, or durations outside 1 to 4 seconds. The action caption describes the hand, visi-

ble object, motion, and outcome. A templated camera sentence standardizes the caption format. The retained times are rounded to source frame indices to extract matching RGB, camera, wrist, and fingertip tracks before conversion to the shared episode schema.



Figure 3. **VITRA examples.** Red points and arrows denote ground-truth positions and orientations in the shared action space.

VITRA. The released hand and camera annotations are aligned with the corresponding source videos, with wrist and fingertip poses recovered from reconstructed joints and MANO rotations. Overlapping annotation windows are merged only when their geometry agrees, without interpolating across gaps. A vision–language model processes 30-second windows at 4 Hz to propose nonoverlapping, semantically complete clips of 2–8 seconds, each with an action sentence and a camera-motion sentence. Geometry checks screen hand depth, anatomical proportions, and temporal stability. Clips must retain at least one continuously valid hand; unavailable hand targets are masked. The retained video and pose tracks are synchronized and exported at 20 Hz.

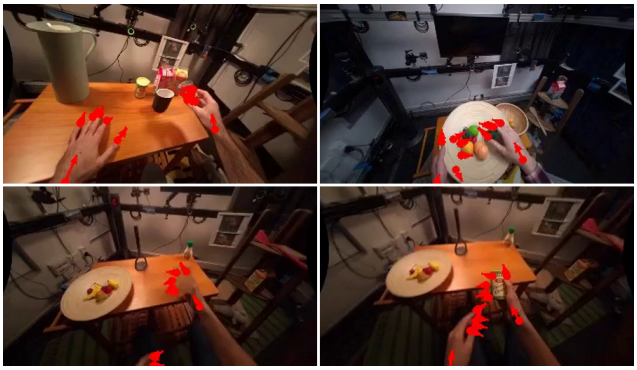

Figure 4. **HOT3D examples.** Red points and arrows denote ground-truth positions and orientations in the shared action space.

HOT3D. Official timestamped images are paired with hand and camera annotations, and MANO reconstruction supplies wrist and fingertip poses. One camera stream per recording is rectified from fisheye to an upright pinhole view, with corresponding updates to the camera axes. The semantic segmenter examines 30-second windows at 4 Hz and proposes complete actions lasting 1–4 seconds, targeting 3 seconds. Each retained clip is separately captioned for its visible manipulation and camera motion. Clips with invalid geometry, discontinuous timing, or frames lacking any valid hand are rejected; missing hand coordinates remain masked. The accepted clips are resampled to the shared 20 Hz episode format.

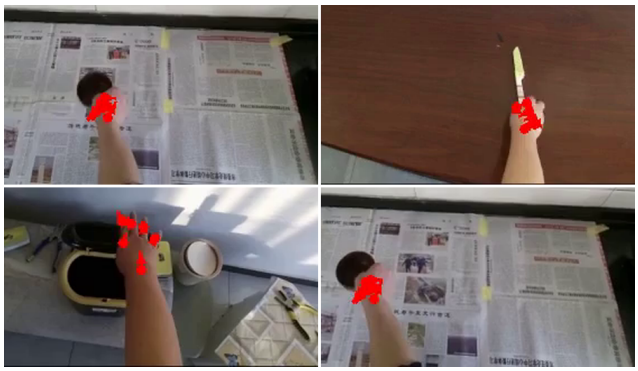

Figure 5. **HOI4D examples.** Red points and arrows denote ground-truth positions and orientations of the annotated hand in the shared action space.

HOI4D. Released MANO parameters reconstruct the annotated hands in camera coordinates, and the recorded camera trajectory maps their wrist and fingertip poses into the common world frame. Only available hand annotations contribute supervision; an unannotated second hand remains masked. The same semantic segmentation procedure as HOT3D produces 1–4-second action clips, followed by separate action and camera-motion captions from the retained video. Quality checks reject clips with implausible hand geometry, unstable hand or camera motion, or frames without a valid hand. Positions and rotations are interpolated only within contiguous valid tracks before synchronized video and pose export at 20 Hz.



Figure 6. **Self-collected examples.** Red points and arrows denote estimated positions and orientations in the shared action space.

Self-collected egodata. The self-collected source provides presegmented egocentric clips and their language annotations. Hand pose tracks are estimated from the RGB recordings using Dyn-HaMR [13]. Each clip is aligned with these estimates and the corresponding camera trajectory using the stored interval and framewise image alignment. The conversion retains the existing clip boundaries and captions, while excluding missing inputs, empty captions, unreliable pose tracks, and ambiguous frame correspondences. Video rotation metadata is applied before excluding portrait recordings. Fixed local-frame transformations align wrist and fingertip axes with the shared convention. RGB frames and pose tracks are resampled together to 20 Hz, using linear interpolation for positions and spherical linear interpolation for rotations.

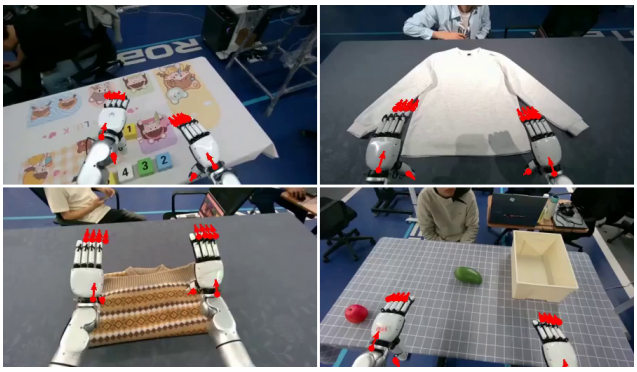

Figure 7. **Bimanual XHand examples.** Red points and arrows denote ground-truth positions and orientations in the shared action space.

Bimanual XHand. The segmenter finds stable extrema in the two gripper signals using thresholds of -0.2 and 0.2. Candidate clips span minimum–maximum–minimum cycles longer than 2 seconds. Nearby boundaries are merged and long intervals are subdivided. A motion-based fallback handles task groups without usable gripper boundaries. Each candidate clip is captioned from a montage of the main and two wrist-camera views sampled at 2 Hz, with the original task instruction supplied when available. The vision–language model writes an action sentence describing the hands, objects, and spatial relations, followed by a sentence about the main-camera view. Responses that fail the caption format check are retried. Forward kinematics converts the synchronized joint records to wrist and fingertip poses and the main-camera trajectory. Each retained segment is exported with its three RGB streams, geometry, and caption at 20 Hz.

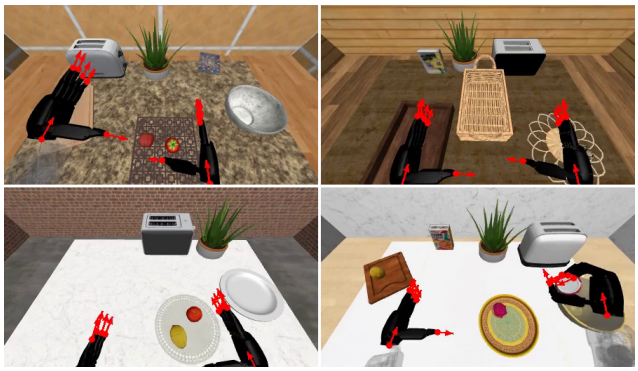

Figure 8. **RoboCasa-GR1 examples.** Red points and arrows denote ground-truth positions and orientations in the shared action space.

RoboCasa-GR1. Official GR1 teleoperation trajectories are replayed in the simulator to obtain synchronized main and wrist-camera views and wrist and fingertip poses through forward kinematics. For six composite task types, a vision–language model separates the complete pick-and-place phase from subsequent door or drawer closing; other trajectories remain intact. Candidate boundaries are checked against object displacement and fixture joint states. All views and pose tracks are split at the same frame without duplicating or dropping source frames. Each resulting episode receives a fresh caption describing the visible robotic manipulation and main-camera motion, while preserving both hands and the original 20 Hz frame sequence.

B.3. Illustrating Episode Completeness

The completeness ablation uses the full mixture of five human sources in both the VLA and WAM comparisons. Within each model family, the two conditions share the batch size, number of training updates, robot sampling probabilities, and evaluation episodes. The unsplit condition retains the original recording as one trajectory. The curated condition segments recordings at action boundaries and retains clips with a clear goal and observable completion. This curation filters out partial events and intervals with ambiguous or unrelated motion, so the comparison

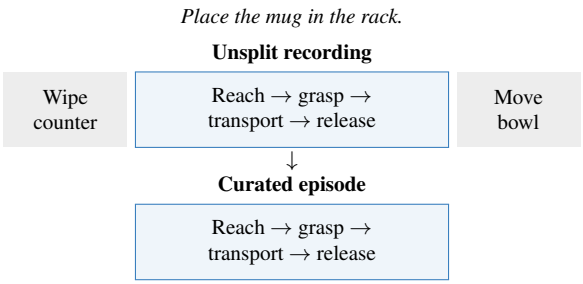

Figure 9. **How does episode curation align trajectories with instructions?** The curated interval in blue spans the reach, grasp, transport, and release described by the instruction. The unsplit recording also includes the preceding and following operations.

measures the joint effect of segmentation and filtering.

Consider a recording in which a person wipes a counter, places a mug in a rack, and then moves a bowl. For the instruction *place the mug in the rack*, the curated interval spans the reach, grasp, transport to the rack, and release. Sampling a fixed number of targets across the entire recording can allocate some to adjacent operations and skip a brief grasp or release. Figure 9 shows the retained interval.

B.4. Temporal Sampling

A training example samples a fixed number of targets from each retained episode. The temporal sampler controls which pose frames enter a training example for a given episode. For a fixed stride s and anchor index u, fixed-step sampling uses indices $u + ks$ for $k = 0, \dots, H$. We pad targets that fall beyond the episode boundary and mask them in the loss. For an episode with L pose frames, event-level sampling uses indices

$$t_k = \text{round}\left(\frac{k(L-1)}{H}\right), \qquad k = 0, \dots, H. \tag{5}$$

The first pose is the anchor, and the remaining poses form the action targets. This sampler assigns the same number of targets to events of different durations and is the default for both model families. It corresponds to the uniform temporal sampling setting in the training configuration. In the reference WAM, the video frames correspond to every fourth pose index along this sampled timeline.

B.5. Data Mixture and Splits

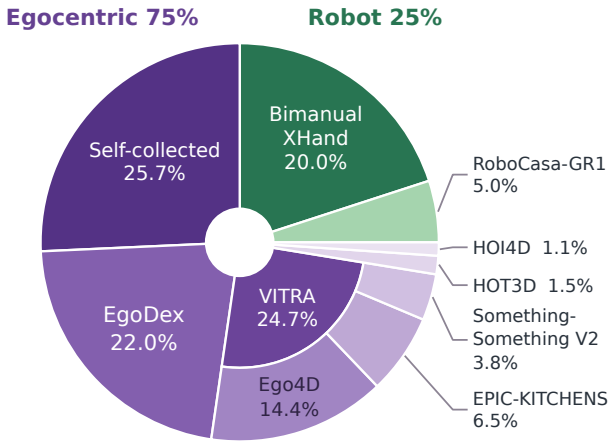

Figure 10. **Pretraining samples allocation.** Egocentric datasets receive 75% of samples, with probabilities proportional to the square root of their filtered durations; VITRA aggregates three subsets. The sampling probabilities of Bimanual XHand (20%) and RoboCasa-GR1 (5%) are fixed across experiments.

Segmentation and filtering reduce approximately 18,000 hours of raw egocentric video to about 2,000 hours of curated clips (1,853 hours in Tab. 1). Including the two robot sources, the pretraining pool totals 2,206 hours.

The reference mixture allocates 75% of samples to human demonstrations, 20% to Bimanual XHand, and 5% to RoboCasa-GR1. Within the human pool, we approximately sample sources in proportion to the square root of their eligible training time. The scale sweep applies the same data fraction to all human sources, forming nested training subsets within each source. We fix the robot training pools and their sampling probabilities. With a fixed global batch size and update count, this keeps the expected number of samples from each robot source constant across scales.

The reference holdout uses a frozen, stratified list of processed episodes with at least 33 frames. Each source family targets 0.1% of its episodes, rounded up and increased as needed to cover each primary label with an eligible episode. EgoDex strata group episodes by task, merging the original training and test names for the same task. Bimanual XHand strata use the task group and recording batch. RoboCasa-GR1 strata use the task and phase. For each processed subset, a fixed-seed permutation assigns approximately 99% of the episodes to training and 1% to validation. The holdout IDs are then excluded from both partitions. Scaling applies only to the human training pools, and both model families use the same fixed evaluation sets across all fractions.

C. Evaluation Protocol

C.1. Controlled Comparisons

Within a model family, each comparison fixes the backbone initialization, input views, action horizon, batch size, optimizer schedule, update count, and evaluation episodes, except for the factor under study. For cross-family comparisons, we share the data pools, sampling proportions, action targets, and evaluation samples while retaining each backbone's native visual preprocessing and training objectives. Temporal-sampling comparisons use the same holdout anchors and target timestamps for all variants, independent of the sampling rule used for training.

C.2. Component-Wise Prediction Error

We compute action errors after denoising. We invert each checkpoint's action normalization and compare predictions with the original target coordinates. For a coordinate set C, the masked mean squared error is

$$E_C = \frac{\sum_{i,t} \sum_{d \in C} m_{itd} (\hat{a}_{itd} - a_{itd})^2}{\sum_{i,t} \sum_{d \in C} m_{itd}}, \tag{6}$$

where i indexes examples, t target steps, and m_{itd} indicates whether coordinate d is valid. We accumulate the error sums and valid-coordinate counts over the entire evaluation set. Mean absolute error uses the same mask and denominator with absolute residuals.

We report results separately for human data, Bimanual XHand, and RoboCasa-GR1. The component groups are wrist translation (6 coordinates), wrist rotation (12), full wrist pose (18), and fingertip pose (90), alongside the full action vector (108). Translation MSE is in squared meters, rotation MSE is in the 6D rotation representation, and fingertip pose MSE combines position and rotation coordinates. The flow-matching validation objective and the WAM video objective serve as training diagnostics. The denoised action error in Eq. (6) provides the common measure of robot validation performance.

C.3. RoboCasa Posttraining and Rollouts

We posttrain exclusively on RoboCasa-GR1, initializing both model families from pretrained weights and starting a fresh optimizer for each run. Table 2 summarizes the numerical settings for posttraining and downstream evaluation. Unlike the event-level pretraining sampler, posttraining uses consecutive commands on the control timeline. The action targets are obtained by applying forward kinematics to the recorded joint commands, starting at the command index of the conditioning observation. Both models share these targets and the masked action flow-matching objective. The WAM also retains its auxiliary video loss during posttraining.

Table 2. RoboCasa-GR1 posttraining and evaluation settings shared by the VLA and WAM.

Setting	Value
Posttraining updates	20,000
Peak learning rate	2×10^{-5}
Command sampling frequency	20 Hz
Commands per predicted chunk	32
Action / state dimensions	111 / 120
Rollouts per model and data scale	240

The 111-dimensional posttraining action contains the two end-effector poses, ten fingertip poses, and absolute waist yaw, pitch, and roll in radians. The end-effector targets use the measured wrist pose at the anchor as their reference, and the fingertip targets are expressed in the corresponding target wrist frame. The input state augments the 117-dimensional pretraining state with the three measured waist angles.

During RoboCasa rollouts, both policies use the same controller and neural inverse kinematics decoder. The decoder converts wrist-local fingertip targets to hand joint commands. The arm controller tracks the end-effector targets, and the waist angles are applied as joint setpoints. We fix the task set, initial-state protocol, control frequency, and success criteria across policies. For each checkpoint, we compare its pretraining action MSE on held-out RoboCasa episodes with its success rate after posttraining under this protocol. The success rate is the percentage of the 240 rollouts that satisfy the task's success criterion, aggregated under the same task allocation for every policy.

C.3.1. Neural Inverse Kinematics Decoder

The shared action representation describes dexterous motion using wrist and fingertip poses, whereas the robot controller requires embodiment-specific joint commands. We therefore adopt Neural IK [14] to map the predicted fingertip positions to hand joint configurations. For each robot embodiment e, a separate decoder g_{ϕ_e} converts the ten fingertip positions expressed in their corresponding wrist frames into a bimanual joint command:

$$\hat{\mathbf{q}}_t^{(e)} = g_{\phi_e}\left(\hat{\mathbf{X}}_t^{(e)}\right). \tag{7}$$

Neural IK models the multimodal inverse-kinematics distribution using an autoregressive Gaussian mixture model:

$$p(\mathbf{q} \mid \mathbf{X}) = \prod_{j=1}^{d_e} p(q_j \mid \mathbf{X}, \mathbf{q}_{<j}), \tag{8}$$

allowing multiple joint configurations to correspond to the same fingertip target. We use 20 Gaussian components

for each joint. At inference time, we sample 64 candidate configurations and select the candidate whose forward-kinematics fingertip positions have the smallest mean Euclidean distance to the predicted target.

During decoder training, a posttrained policy infers wrist-local fingertip positions from observations in the corresponding robot dataset. These policy predictions serve as decoder inputs, paired with ground-truth hand joint commands at the same target timesteps. Training on policy-generated inputs exposes the decoder to prediction errors encountered at deployment, helping it accommodate imperfect fingertip inputs. The decoder is trained using a conditional negative log-likelihood objective and remains frozen during rollout. Within each robot embodiment, the same decoder is used across all VLA, WAM, and data-scaling settings to ensure a consistent low-level control interface.

References

[1] StarVLA Community. StarVLA: A lego-like codebase for Vision-Language-Action model developing. *arXiv preprint arXiv:2604.05014*, 2026. 1

[2] NVIDIA, Johan Bjorck, Fernando Castañeda, Nikita Cherniadev, Xingye Da, Runyu Ding, Linxi "Jim" Fan, Yu Fang, Dieter Fox, Fengyuan Hu, Spencer Huang, Joel Jang, Zhenyu Jiang, Jan Kautz, Kaushil Kundalia, Lawrence Lao, Zhiqi Li, Zongyu Lin, Kevin Lin, Guilin Liu, Edith Llontop, Loic Magne, Ajay Mandlekar, Avnish Narayan, Soroush Nasiriany, Scott Reed, You Liang Tan, Guanzhi Wang, Zu Wang, Jing Wang, Qi Wang, Jiannan Xiang, Yuqi Xie, Yinzhen Xu, Zhenjia Xu, Seonghyeon Ye, Zhiding Yu, Ao Zhang, Hao Zhang, Yizhou Zhao, Ruijie Zheng, and Yuke Zhu. GR00T N1: An open foundation model for generalist humanoid robots. In *ArXiv Preprint*, 2025. 1

[3] Tianyuan Yuan, Zibin Dong, Yicheng Liu, and Hang Zhao. Fast-WAM: Do World Action Models need test-time future imagination? *arXiv preprint arXiv:2603.16666*, 2026. 1

[4] Yi Zhou, Connelly Barnes, Jingwan Lu, Jimei Yang, and Hao Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5745–5753, 2019. 1

[5] Ryan Hoque, Peide Huang, David J. Yoon, Mouli Sivapurapu, and Jian Zhang. EgoDex: Learning dexterous manipulation from large-scale egocentric video. *arXiv preprint arXiv:2505.11709*, 2026. 1

[6] Yunze Liu, Yun Liu, Che Jiang, Kangbo Lyu, Weikang Wan, Hao Shen, Boqiang Liang, Zhoujie Fu, He Wang, and Li Yi. HOI4D: A 4d egocentric dataset for category-level human-object interaction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 21013–21022, 2022. 1

[7] Prithviraj Banerjee, Sindi Shkodrani, Pierre Moulon, Shreyas Hampali, Fan Zhang, Jade Fountain, Edward Miller, Selen Basol, Richard Newcombe, Robert Wang, Jakob Julian Engel, and Tomas Hodan. Introducing HOT3D: An egocentric dataset for 3d hand and object tracking, 2024. 1

[8] 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. *arXiv preprint arXiv:2510.21571*, 2025. 1

[9] Kristen Grauman, Andrew Westbury, Eugene Byrne, et al. Ego4D: Around the world in 3,000 hours of egocentric video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18995–19012, 2022. 1

[10] Dima Damen, Hazel Doughty, Giovanni Maria Farinella, Sanja Fidler, Antonino Furnari, Evangelos Kazakos, Davide Moltisanti, Jonathan Munro, Toby Perrett, Will Price, and Michael Wray. The EPIC-KITCHENS dataset: Collection, challenges and baselines. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 43(11):4125–4141, 2021. 1

[11] Raghav Goyal, Samira Ebrahimi Kahou, Vincent Michalski, Joanna Materzyńska, Susanne Westphal, Heuna Kim, Valentin Haenel, Ingo Freund, Peter Yianilos, Moritz Mueller-Freitag, Florian Hoppe, Christian Thurau, Ingo Bax, and Roland Memisevic. The "something something" video database for learning and evaluating visual common sense. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5842–5850, 2017. 1

[12] Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. RoboCasa: Large-scale simulation of everyday tasks for generalist robots, 2024. 1

[13] Zhengdi Yu, Stefanos Zafeiriou, and Tolga Birdal. Dynhamr: Recovering 4d interacting hand motion from a dynamic camera. In *2025 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 27716–27726. IEEE, 2025. 4

[14] Raphael Bensadoun, Shir Gur, Nitsan Blau, and Lior Wolf. Neural inverse kinematic. In *Proceedings of the 39th International Conference on Machine Learning*, pages 1787–1797. PMLR, 2022. 6