EgoSteer: A Full-Stack System Towards Steerable Dexterous Manipulation from Egocentric Videos

Yifan Zhong[1,2*], Zhang Chen[1,2*], Tianrui Guan[1,2*], Fanlian Zeng[2,3*], Yuyao Ye[1,2], Tianjia He[2], Ka Nam Lui[1,2], Jiayi Li[1,2], Tingrui Zhang[1,2], Ruilin Yan[1,2], Xinhao Ji[1,2], Guangyu Zhao[1,2], Wenjie Lou[1,2], Jiayuan Zhang[1,2], Yuanpei Chen[1,2†], Yaodong Yang[1,2†]

[1]Institute for AI, PKU, [2]PKU-PsiBot Joint Lab, [3]UPenn.

Abstract: Steerability is a defining capability of generalist robot policies, yet remains largely absent in dexterous-hand systems for lack of large-scale, language-aligned, and action-accurate demonstration data. To address this bottleneck, we present a full-stack system that scales dexterous VLA pre-training from egocentric human videos and enables data-efficient real-robot post-training. It integrates **EgoSmith**, a data pipeline that curates in-the-wild egocentric videos into 9.6 K hours of high-quality pre-training data with 9× higher throughput and better accuracy than prior SOTA; a unified robot stack for teleoperation and human-in-the-loop correction; and **EgoSteer**, a world-model-enhanced VLA trained on optimized infrastructure. Human-data pre-training equips EgoSteer with language-guided manipulation priors, which are grounded through robot post-training and improved by DAgger refinement. Empirically, EgoSteer robustly executes free-form instructions across 40+ diverse tasks, demonstrating failure recovery, dexterity, and generalization. The pre-trained model also few-shot adapts to complex long-horizon tasks, including box folding, on two embodiments with 75+% success. We open-source the system, data, and model at `egosteer.github.io`.

Keywords: Steerable Dexterous Manipulation, VLA Models, Egocentric Videos

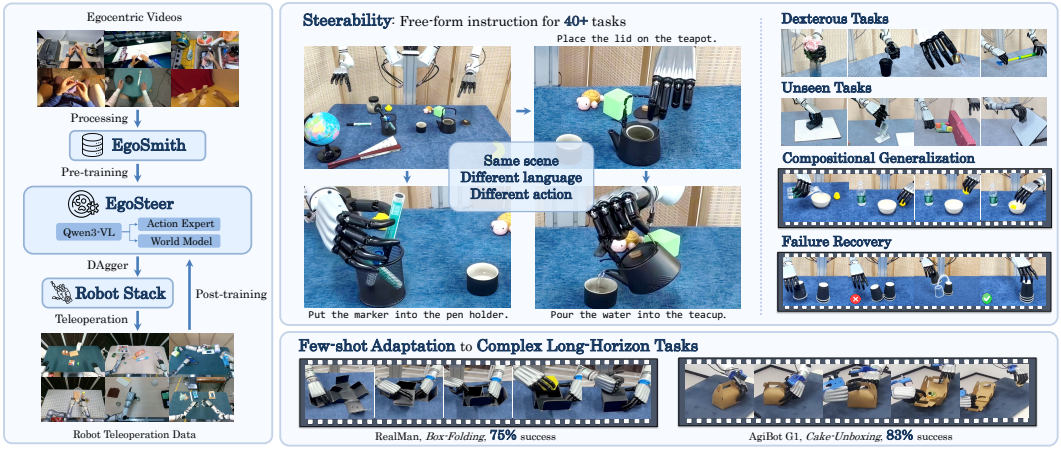


Figure 1: Our **full-stack system** integrates EgoSmith, Robot Stack, and EgoSteer to learn from large-scale egocentric human videos and facilitate data-efficient real-robot post-training, enabling steerable dexterous manipulation across over 40 tasks alongside few-shot adaptation to complex, long-horizon tasks.

1 Introduction

A central goal of general-purpose embodied intelligence is to enable robots to perform diverse manipulation tasks from open-ended human intent. Despite rapid progress in embodied foundation models [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14], most systems still require task-specific fine-tuning, while the few that follow free-form language are largely limited to grippers [15, 16]. Dual-

*Equal contribution.
†Corresponding author emails: yuanpei.chen312@gmail.com and yaodong.yang@pku.edu.cn.

dexterous-hand robots provide a more expressive embodiment, with greater actuation capacity and fine-grained interaction potential for general-purpose manipulation. Yet on this more capable but more challenging platform, steerable dexterous manipulation remains largely unrealized.

The key bottleneck lies in data and system scalability. While language-guided manipulation demands large-scale, high-quality data, collecting such demonstrations directly on dexterous robots is exceptionally difficult, particularly for a specific embodiment. Egocentric human videos [17, 18] offer a scalable alternative, as human hand manipulations contain rich interaction knowledge and are spontaneously generated at a massive scale. However, raw egocentric videos are noisy and lack reliable language and action annotations. Without systematic curation, these unstructured videos provide unstable supervision and can degrade downstream robot policies. Even with high-quality human data, the system must employ high-capacity models trained with effective objectives on scalable infrastructure, and ground the learned priors to the target robot. Failing to address any of these co-dependent components prevents the realization of general language-following manipulation.

In this paper, we close this gap by proposing and open-sourcing a full-stack system for steerable dexterous manipulation. Our system begins with **EgoSmith**, an egocentric data pipeline that curates in-the-wild egocentric videos into clean, fully-annotated training data. By integrating pre-filtering, 4D motion estimation, language labeling, and post-filtering, EgoSmith aligns RGB-D images, world-space hand trajectories, camera parameters, and textual instructions, achieving a $9\times$ throughput increase and more precise, comprehensive annotations than prior open-source SOTA [19]. Using EgoSmith, we curate a 9.6K-hour pre-training corpus across 12 egocentric datasets. To ground these human-hand interaction priors onto physical embodiments, we design a unified robot stack for teleoperation, model inference, and human-in-the-loop correction. By mapping the operator's relative motions onto the intervened robot states, this stack enables seamless expert intervention for efficient DAgger [20] refinement from arbitrary deployment states, effectively correcting policy failures. Using this framework, we collect 187 hours of high-quality teleoperation data across 193 semantically-diverse tasks. Finally, we introduce **EgoSteer**, a novel world-model-enhanced VLA trained on an optimized infrastructure. By integrating a world-model expert that predicts action-induced future states in the DINOv3 [21] latent space, EgoSteer enhances the VLM backbone's action imagination and modality alignment, enabling steerable and fine-grained manipulation. To facilitate human-robot transfer, EgoSteer employs a unified action space of wrist poses and fingertip keypoints [22, 23], coupled with training-time RTC [24] to eliminate real-world execution pauses.

Empirically, through large-scale egocentric pre-training, diverse real-robot post-training, and efficient DAgger refinement, EgoSteer robustly follows free-form instructions to execute over 40 tasks with a 75% average success rate, exhibiting fine-grained dexterity, failure recovery, and generalization. Furthermore, systematic evaluations confirm the significance of each component, including egocentric pre-training data scale and quality, the world-model objective, training-time RTC, and DAgger refinement, enabling EgoSteer to consistently outperform strong baselines such as $\pi_{0.5}$ [2] and Being-H0.5 [8]. Additionally, the pre-trained manipulation priors also enable few-shot adaptation to challenging long-horizon tasks, such as box folding and cake unboxing, across multiple embodiments, achieving a 75+% success rate. Conversely, our from-scratch baseline and sample-efficient imitation learning methods, including Diffusion Policy [25] and IMLE [26], fail entirely, illustrating the inherent difficulty of these tasks and the efficacy of our curated pre-training priors. To summarize, our contributions are five-fold:

- **EgoSmith**, an egocentric data pipeline that curates a 9.6K-hour fully-annotated pre-training corpus across 12 datasets, achieving a $9\times$ throughput and better accuracy over prior SOTA.
- **A unified robot stack** integrating teleoperation, inference, and seamless human-in-the-loop DAgger correction, that collects 187 hours of data across 193 dexterous tasks.
- **EgoSteer**, a world-model-enhanced VLA alongside an optimized training infrastructure.
- **Extensive real-robot evaluations** demonstrating that EgoSteer robustly performs free-form language-following across 40 tasks and few-shot adapts to complex long-horizon tasks.
- **Open-source release** of our complete system, datasets, and model checkpoints.

2 Related Work

Generalist robot policies. Generalist manipulation policies based on foundation models have recently emerged towards general-purpose manipulation [2, 27, 28, 29, 30]. However, early efforts remain largely confined to simple tasks and rely heavily on single-task fine-tuning, struggling to follow free-form language instructions. Although recent works [15, 16] have made breakthroughs in generalization, they rely heavily on massive real-robot multi-task datasets, or requires intensive computation and complex optimization to sustain real-time control and are limited to grippers.

Scaling with egocentric human videos. Egocentric videos [17, 18, 31, 32, 33, 34, 35, 36, 37] (approximately 116K hours) and data processing tools [19, 38] offer a scalable source for learning dexterous manipulation. Existing policies [6, 7, 8, 22, 23, 39] leverage these videos through large-scale pre-training or cross-embodiment co-training. By aligning human hands and robot action spaces through diverse approaches, these methods effectively transfer human priors to the robot domain. Notably, EgoScale [6] reveals a log-linear scaling law in pre-training and introduces a mid-training stage to align human and robot space. Nevertheless, converting massive human videos into training signals remains highly inefficient, and current policies still struggle with steerable control.

Human-in-the-loop post-training. Human-in-the-loop post-training is key to elevating the performance ceiling and resolving out-of-distribution failures with high sample efficiency. Recent paradigms leverage online reinforcement learning with human copilots [40], compliant residual feedback for contact-rich tasks [41], or hand-arm intervention frameworks for dexterous VLAs [42, 43]. However, these approaches still struggle with real-time, high-frequency corrections in high-DoF joint spaces and require prohibitive trajectory labeling labor.

3 EgoSmith: Curating Egocentric Videos into Grounded Dexterous Priors

While egocentric data are rich in fine-grained interactions and highly scalable for general embodied learning, they are typically monocular RGB videos suffering from camera jitter, frequent occlusions, and a lack of annotations. This section presents **EgoSmith** (Figure 2), an efficient automated pipeline that transforms massive raw videos into fully-annotated training samples to enable effective learning.

To achieve raw data cleaning, labeling, and quality control, EgoSmith employs a four-stage pipeline. The **first** stage, **pre-filtering**, uses simple yet effective heuristics to discard locomotion segments and hand misidentifications that degrade downstream 4D motion estimation. Specifically, we filter out active displacement by computing average optical flow over a 128-point grid, leveraging its strong correlation with human locomotion in egocentric videos. We then eliminate frames with severe occlusions or bystander interference by applying geometric criteria to the hand counts, scales, and coordinates detected by YOLO [44, 45], preserving only clearly visible egocentric manipulations.

Building upon the state-of-the-art method HaWoR [19], the **second** stage, **4D motion estimation**, reconstructs camera extrinsics, depth, and world-space hand trajectories. Since HaWoR lacks depth estimation and relies on DROID-SLAM [46] for tracking, which is computationally expensive and subject to drift under rapid head movements and drastic scene changes, we propose an improved, more robust and efficient scheme. We leverage DPVO [47] for more stable, *metric-free* camera tracking and keyframe depth estimation, and Any4D [48] for frame-wise, *metric-scale* depth prediction. Aligning their scale ratio recovers more accurate metric-scale camera trajectories, which we use to transform camera-frame hand motions into more consistent world-space trajectories. By leveraging DPVO, which is significantly faster than DROID-SLAM, and optimizing I/O and batching, the pipeline achieves a $9\times$ throughput speedup over HaWoR, facilitating large-scale processing.

The **third** stage, **language labeling**, performs multi-granularity language annotation, which is crucial for enabling free-form instruction following. We first employ Qwen3.5-VL-Plus [49] to filter out segments lacking meaningful hand-object manipulation, discarding an additional 3.5% of clips that passed the heuristic rules but lacked active operations. For the remaining clips, the model generates coarse-to-fine, five-level language instructions. This hierarchical annotation simultaneously provides task-level semantic grounding and action-level spatiotemporal grounding, enabling downstream models to learn and respond to instructions across varying levels of abstraction.

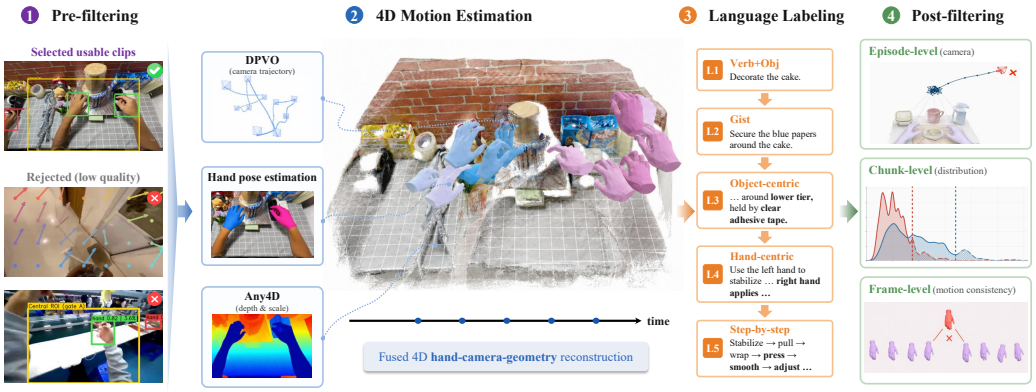


Figure 2: Overview of EgoSmith. Integrating pre-filtering, 4D motion estimation, language labeling, and post-filtering, EgoSmith efficiently curates in-the-wild egocentric videos into clean and annotated training samples.

The **fourth** stage, **post-filtering**, performs multi-level quality control on the generated data. First, at the episode level, we compute camera translation distributions to discard outliers, while applying hard rotation thresholds to drop episodes with excessive head motions. Second, at the chunk level, we transform wrist poses into its middle camera frame, project finger keypoints into frame-wise wrist frames, and discard spatial outliers across wrist and finger coordinates. Finally, at the frame level, we compute frame-to-frame deltas of camera, wrist, and finger motions, filtering out abrupt jumps with hard thresholds. This coarse-to-fine filtering systematically eliminates unreliable clips caused by action jumps, inaccurate metric scales, or head tracking drift, ensuring high corpus quality.

With EgoSmith, we curate a large-scale egocentric pre-training corpus across 12 raw datasets [35, 50, 34, 33, 51, 36, 52, 31, 18, 17, 32, 37]. To filter out highly repetitive videos, we subsample Egocentric-10K [32] and Egocentric-100K [37]. For Ego4D [31] and EPIC-KITCHENS [52], we apply EgoSmith to their respective VITRA [38] subsets. Ultimately, this pipeline yields a fully-annotated egocentric dataset comprising 9.60K hours, 2.09M episodes, and 1.04B frames.

4 A Unified Robot Stack for Teleoperation and DAgger Post-Training

While EgoSmith's curated egocentric data provides rich manipulation priors, direct transfer to real robots is prevented by the embodiment gap across visual, dynamics, and kinematic degrees of freedom, necessitating real-robot teleoperation to ground these priors onto the target embodiment. This section presents the **Unified Robot Stack** (Figure 3), which shares low-level control and dynamics to simultaneously support teleoperation, policy inference, and human-in-the-loop correction.

For teleoperation, a pair of PsiBot SynGlove-Air gloves and Vive Trackers capture the operator's $SE(3)$ wrist poses and hand joint angles, which respectively drive two robotic arms through inverse kinematics (IK) computed via `mink` [53] and two 6-DoF robotic hands via joint mapping. During policy inference, the trained policy publishes the wrist pose trajectory in the camera frame and the hand keypoints in the wrist frame. These actions share the same arm and hand FK/IK and control nodes with teleoperation, ensuring identical execution dynamics across training and inference.

The primary challenge in enabling human-in-the-loop intervention is preventing sudden state jumps at the handover boundary to ensure a smooth transition. To address this, we propose a *relative motion mapping* scheme. When the operator signals intervention by pressing a foot pedal at step t, the system records the robot end-effector poses $\mathbf{T}_t^{\mathrm{R},i} \in SE(3)$, human wrist poses $\mathbf{T}_t^{\mathrm{H},i} \in SE(3)$, robot hand joint states $\mathbf{q}_t^{\mathrm{R},i} \in \mathbb{R}^6$, and glove states $\mathbf{q}_t^{\mathrm{H},i} \in \mathbb{R}^6$ for each arm/hand index $i \in \{1, 2\}$. Subsequently, at any $t' \geq t$, the operator's relative motions, $\Delta\mathbf{T}_{t\to t'}^{\mathrm{H},i} = (\mathbf{T}_t^{\mathrm{H},i})^{-1}\mathbf{T}_{t'}^{\mathrm{H},i}$, $\Delta\mathbf{q}_{t\to t'}^{\mathrm{H},i} = \mathbf{q}_{t'}^{\mathrm{H},i} - \mathbf{q}_t^{\mathrm{H},i}$, are mapped to the robot, computing the commanded end-effector poses $\mathbf{T}_{t'}^{\mathrm{R},i}$ and hand joint states $\mathbf{q}_{t'}^{\mathrm{R},i}$ as $\mathbf{T}_{t'}^{\mathrm{R},i} = \mathbf{T}_t^{\mathrm{R},i}\Delta\mathbf{T}_{t\to t'}^{\mathrm{H},i}$ and $\mathbf{q}_{t'}^{\mathrm{R},i} = \mathbf{q}_t^{\mathrm{R},i} + \Delta\mathbf{q}_{t\to t'}^{\mathrm{H},i}$. This formulation allows the operator to smoothly take over control by simply mimicking the robot's motion. After correcting failures, the operator hands control back to the policy via another pedal press, resuming inference.

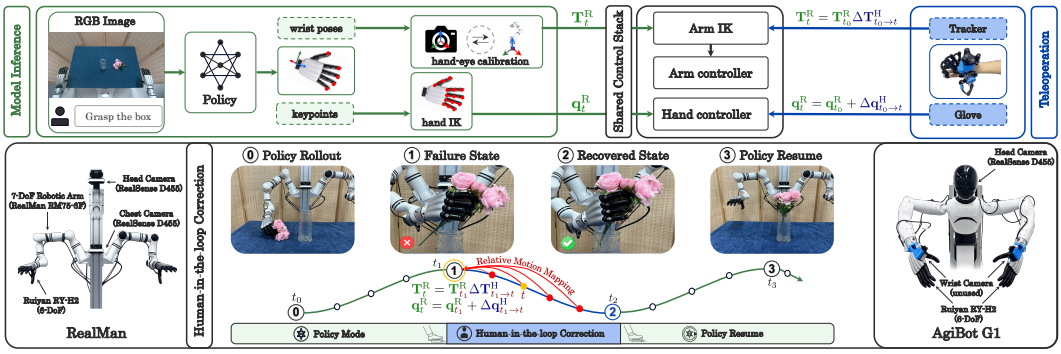


Figure 3: Overview of the Robot Stack. It unifiedly supports teleoperation, policy inference, and human-in-the-loop correction. A relative motion mapping scheme is employed to facilitate seamless transitions during interventions, and the bottom row illustrates the two robotic embodiments utilized in our experiments.

Only these intervention segments are utilized for subsequent training. This design achieves a handover success rate exceeding 85%, enabling efficient collection of corrective demonstrations.

With unified robot stack, we construct a 187-hour robot dataset across 193 tabletop tasks spanning seven categories: Pick-and-Place(PnP)-Easy/Medium/Hard, non-prehensile, reorient, bimanual, and contact-rich operations. These comprise 56 common tasks covering most core manipulation primitives, alongside 137 long-tail tasks to facilitate human-to-robot transfer and grounding. Multi-level language annotations are generated using Qwen3-VL-Flash [54], followed by manual verification. To cover diverse primitives and establish modal alignment, data collection follows a free-form protocol where environments are cluttered, and tablecloths, object instances, and initial configurations are randomized without pre-defined trajectories. This emphasis on natural, human-like execution yields substantial rollout diversity, fostering policy robustness and multi-task generalization.

5 EgoSteer: A World-Model-Enhanced VLA for Steerable Dexterity

To effectively learn language-guided manipulation from human data, teleoperation, and corrections, we propose **EgoSteer**, a flow-based VLA model enhanced by a world-model objective, shown in Figure 4. To ensure robust vision-language understanding while modeling multimodal continuous actions, EgoSteer pairs a Qwen3-VL 2B backbone [54] with a DiT-based [55] action expert, which jointly attends to itself and backbone to generate action chunks via flow-matching [1]. To facilitate human-robot transfer, we design a unified data format and state-action space across both domains. An episode τ of length N is represented as $\tau = \{l, \mathbf{K}, (\mathbf{I}_t, \mathbf{D}_t, \mathbf{T}_t^{w2c}, \mathbf{s}_t^w, \mathbf{a}_t^w)_{t=0}^{N-1}\}$, where l is the instruction, $\mathbf{K} \in \mathbb{R}^{3\times3}$ is the camera intrinsics, t is the timestep, $\mathbf{I}_t \in \mathbb{R}^{H\times W\times3}$ and $\mathbf{D}_t \in \mathbb{R}^{H\times W\times1}$ are the RGB and depth images, $\mathbf{T}_t^{w2c} \in SE(3)$ is the world-to-camera extrinsics, and bimanual world-frame states and actions $\mathbf{s}_t^w, \mathbf{a}_t^w \in \mathbb{R}^{48}$ comprise the 3D wrist translation, 6D wrist rotation, and 15D fingertip keypoints of both hands. Since depth is unused in our model, the training sample at each timestep t becomes $\{l, \mathbf{K}, \mathbf{I}_{t-k+1:t}, \mathbf{s}_{t-k+1:t}^{c_t}, \mathbf{a}_{t:t+h-1}^{c_t}\}$, where k and h denote the history and prediction lengths, and $\mathbf{s}_{t-k+1:t}^{c_t}$ is the state history transformed into the current camera frame c_t via $\mathbf{T}_t^{w2c}$. The relative action chunk $\mathbf{a}_{t:t+h-1}^{c_t}$ is computed in c_t relative to $\mathbf{s}_t^{c_t}$, where wrist motions are relative $SE(3)$ transforms and finger movements are coordinate displacements. For simplicity, we omit the c_t superscript for $\mathbf{s}$ and $\mathbf{a}$ hereafter. Furthermore, to avoid execution pauses during real-robot inference, we implement training-time Real-Time Chunking (RTC) [24] in the action expert. Specifically, we feed a clean action prefix $\mathbf{a}_{\text{pre}} = \mathbf{a}_{t:t+d-1}$ of randomly sampled delay d as ground truth and train the expert solely to denoise the subsequent actions $\tilde{\mathbf{a}}_{\text{suf}} = \tilde{\mathbf{a}}_{t+d:t+h-1}$. During deployment, the robot executes the reserved prefix $\mathbf{a}_{\text{pre}}$ during asynchronous VLA inference, transitioning seamlessly to the new chunk $\mathbf{a}_{\text{suf}}$ without execution gaps. Denote our model by π, we train it using Conditional Flow Matching (CFM) [56] by regressing the linear velocity field of the target suffix $\mathbf{a}_{\text{suf}}$ conditioned on the context $\mathbf{C}_t = \{l, \mathbf{K}, \mathbf{I}_{t-k+1:t}, \mathbf{s}_{t-k+1:t}, \mathbf{a}_{\text{pre}}\}$ with $\mathcal{L}_{\text{CFM}}(\pi) = \mathbb{E}_{t,\eta,\boldsymbol{\epsilon}}\left[\|\pi(\tilde{\mathbf{a}}_{\text{suf}}, \eta, \mathbf{C}_t) - (\mathbf{a}_{\text{suf}} - \boldsymbol{\epsilon})\|^2\right]$, where $\eta \in [0,1]$, $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$, and $\tilde{\mathbf{a}}_{\text{suf}} = (1-\eta)\boldsymbol{\epsilon} + \eta\mathbf{a}_{\text{suf}}$. To expand the effective batch size and improve loss gradient, we sample four random η per sample.

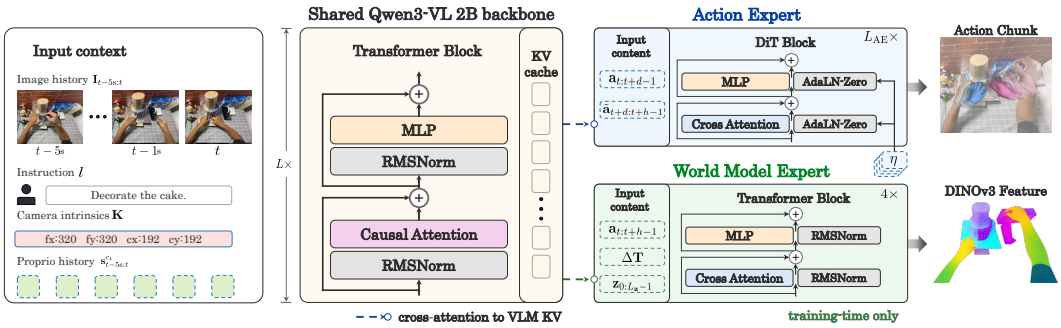


Figure 4: Overview of EgoSteer, a world-model-enhanced VLA model for steerable dexterity. A shared Qwen3-VL backbone extracts KV cache representations from multi-modal inputs. The action expert jointly attends to itself and the backbone to generate action chunks via flow-matching, integrating training-time RTC to eliminate execution pauses. The training-only world model expert predicts future DINOv3 features to improve action accuracy with zero inference overhead.

While VLA excels at vision-language understanding, lacking future imagination limits its action generation accuracy [16]. To address this, we introduce a world-model expert to predict action-induced future DINOv3 features [21]. The expert takes the ground-truth $\mathbf{a}_{t:t+h-1}$, the relative camera motion $\Delta\mathbf{T} = \mathbf{T}_t^{w2c}(\mathbf{T}_{t+h-1}^{w2c})^{-1}$, and learnable query tokens $\mathbf{z}_{0:L_\mathbf{z}-1}$ of length $L_\mathbf{z}$ as inputs to jointly attend to themselves and the backbone. The upsampled output of $\mathbf{z}$ is supervised against future frame $\mathbf{I}_{t+h-1}$ DINOv3 features via regression loss, which provides a more direct and stable supervision signal than generative loss. For robot setups with an additional chest camera, the backbone inputs both cameras' image histories and intrinsics, while the expert receives their relative motions and regresses both future DINOv3 features by adding distinct camera embeddings to $\mathbf{z}$. To focus optimization on the backbone's representation, this module comprises only four Transformer layers attending to the backbone layers at regular intervals, guiding the gradient to primarily shape the backbone. Crucially, the expert is discarded during inference, avoiding computational overhead.

To enable efficient training, we develop an optimized infrastructure. We use Hybrid Sharded Data Parallel (HSDP) [57] to scale batch size and overlap computation with communication, while incorporating mixed-precision training. To enhance GPU utilization, we leverage `torch.compile` for kernel fusion and integrate FlexAttention [58] to optimize attention. To mitigate I/O bottlenecks, WebDataset is employed for sequential streaming instead of random reads, drastically reducing I/O pressure while maintaining training randomness via shuffle buffers, random sample dropping, and randomized shard reading. This pipeline achieves a 44.5% Model FLOPs utilization (MFU) and a throughput of 97 samples/s on an 8-A800 node, scaling near-linearly to 128 GPUs.

6 Experiments

We conduct extensive experiments to answer five core research questions: **Q1**. How well does EgoSteer follow free-form instructions to complete various tasks? **Q2**. Does DAgger efficiently and effectively improve performance? **Q3**. How does the pre-training scale affect downstream performance, and how does EgoSteer compare with other VLA baselines? **Q4**. Are egocentric pre-training data quality, the world-model objective, and training-time RTC essential to strong performance? **Q5**. Can large-scale egocentric pre-training enable few-shot adaptation to complex long-horizon tasks?

6.1 Steerable Multi-Task Manipulation and Generalization

Setup. EgoSteer is pre-trained on the 9.6K-hour egocentric dataset at 384×384 resolution and post-trained on the 187-hour real-robot dataset using head and chest cameras at 640×480 resolution. Next, three DAgger iterations are conducted, collecting 3.7K trajectories across 56 tasks, yielding 8.3 hours of correction data to refine the policy. Finally, the policy is evaluated across 32 seen tasks, 4 compositional generalization tasks, and 4 unseen tasks. Compositional tasks recombine seen primitives into novel sequences, while unseen tasks feature completely novel action semantics. Each task is tested over 10 randomized trials under free-form instructions to measure success rates.

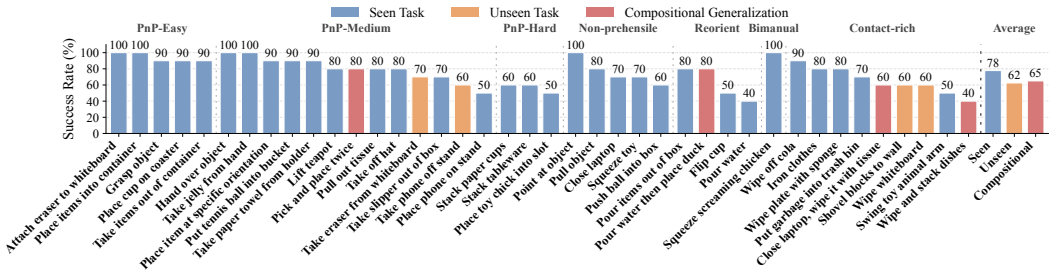


Figure 5: Steerable manipulation performance of EgoSteer across 40 tasks spanning 7 categories. It robustly follows free-form language instructions to achieve an overall success rate of 75%, demonstrating generalization.

Results. As shown in Figure 5, EgoSteer achieves 80+% success rates on 22 tasks and an overall average of 75%. Crucially, in cluttered, randomized layouts, the policy strictly adheres to language instructions regarding target objects, hand selections, and specific actions to execute correct tasks, even for fine-grained manipulation of flat and small objects. Furthermore, EgoSteer exhibits robust failure recovery, executing multiple retries if a previous step fails. It also achieves average success rates of 65% on compositional generalization and 62% on unseen tasks, respectively, confirming that our full-stack system endows EgoSteer with robust, steerable dexterity that covers most common tasks while generalizing effectively to novel environments and tasks.

6.2 Efficacy of DAgger Post-Training

Setup. The model fine-tuned solely on the teleoperation data in Section 6.1 is denoted as EgoSteer-FT, whereas the model refined through three DAgger iterations is referred to as EgoSteer-DG. These models are compared on four dexterous and failure-prone seen tasks, such as "*place phone on stand*". For each task, 10 evaluation trials are conducted using the same settings as in Section 6.1.

Results. As shown in Table 1a, after DAgger iterations totaling 8.3 hours, the average success rate increases from 22.5% to 62.5%. This efficacy stems from the targeted collection of corrective demonstrations addressing deployment failures, achieving a performance leap with minimal data. The refined policy not only exhibits robust failure recovery but also adaptively adjusts actions at critical manipulation bottlenecks. Crucially, these recovery and adjustment capabilities generalize to novel tasks, substantially improving the overall robustness of the DAgger-trained policy.

6.3 Scaling of Pre-Training and Baseline Comparisons

Setup. The EgoSteer models pre-trained on 3K, 6K, and 9.6K hours of egocentric data, alongside a non-pretrained baseline trained from scratch, are post-trained on the real-robot dataset. These models, denoted as EgoSteer-0/3/6/9.6K, are evaluated across 10 tasks. Additionally, the baselines $\pi_{0.5}$ [2] and Being-H0.5 [8] are post-trained on our real-robot dataset and compared across 10 easier tasks.

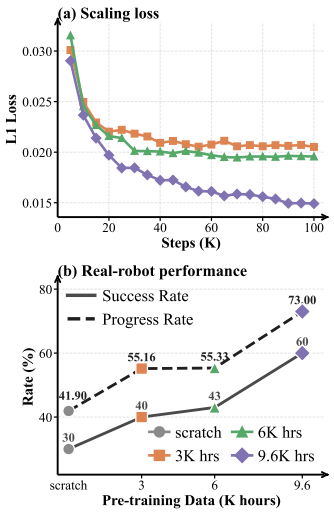


Figure 6: Scaling behavior of pre-training loss and downstream real-robot post-training performance.

Results. As shown by the pre-training loss curves of EgoSteer-3K/6K/9.6K in Figure 6a and the real-robot success and progress rates in Figure 6b, scaling pre-training data drives training loss to lower convergence values while improving real-world execution performance. With expanding pre-training data, the policy exhibits the emergence of failure recovery, enhanced instruction-following, and improved action accuracy, indicating that the model successfully acquires physical common sense for error adjustment and language-guided manipulation priors from increasingly larger datasets curated by EgoSmith. These results reveal that scaling egocentric pre-training is highly beneficial for downstream bimanual manipulation, validating the quality of EgoSmith's annotations, EgoSteer's learning capacity, and the stability of our training infrastructure.

Method	Avg.
EgoSteer-FT	22.5%
EgoSteer-DG	**62.5%**

(a) DAgger ablation.

Method	Avg.
$\pi_{0.5}$ [2]	22%
Being-H0.5 [8]	39%
Ours	**74%**

(b) Baseline comparison.

Method	Avg.
No WM-objective	31%
No training-RTC	39%
Noisy data	33%
Ours	**44%**

(c) Training ablation.

Method	Box-Folding	Cake-Unboxing
DP [25]	0%	0%
IMLE [26]	0%	0%
Ours (scratch)	0%	0%
Ours	**75%**	**83%**

(d) Few-shot adaptation.

Table 1: Extensive experiments to validate the significance of system components and compare with baselines.

The comparison of EgoSteer-9.6K with the baselines is presented in Table 1b, where EgoSteer consistently outperforms both. Specifically, both baselines suffer from inconsistent action representations between pre- and post-training phases, utilize a smaller resolution, and lack deployment optimizations. Consequently, although they can handle basic PnP actions, they exhibit weak instruction-following, limited generalization, and imprecise execution. These performance gains highlight the critical advantage of our unified, full-stack system.

6.4 Ablation Studies

Setup. The model EgoSteer-1K is pre-trained on 1K hours of egocentric data and post-trained on the real-robot dataset. It is compared against three ablated variants across 10 seen tasks: first, *No WM-objective*, which omits the world-model expert during both pre-training and post-training; second, *No training-RTC*, which disables training-time RTC during training and inference; and third, *Noisy data*, which utilizes noisy egocentric data unfiltered by EgoSmith pre-filtering and post-filtering.

Results. As shown in Table 1c, removing any core component leads to a substantial performance decline, validating the necessity of each module. Specifically, the *No WM-objective* variant exhibits a significant reduction in fine-grained manipulation accuracy, confirming that enhancing the backbone's action imagination via the world model is critical for precise action generation. The *No training-RTC* variant introduces severe action pauses and disrupts execution dynamics, causing contact-rich tasks to fail entirely due to continuous jitter. Finally, the *Noisy data* variant fails to converge effectively, leading to severe degradation in both instruction-following and manipulation precision. These ablation results strongly validate the efficacy of our unified full-stack system.

6.5 Few-Shot Adaptation to Complex Long-Horizon Tasks

Setup. We few-shot fine-tune the pre-trained EgoSteer-9.6K on two challenging long-horizon tasks. These include 18-step 40-second "*box folding*" on RealMan using 120 demonstrations, and 9-step 1-minute "*cake unboxing*" on AgiBot-G1 using 200 demonstrations. The adapted policy is compared against strong imitation learning baselines, namely DP [25] and IMLE [26], alongside our non-pretrained ablation, across 24 real-world trials per task under randomized object configurations.

Results. As shown in Table 1d, despite the long-horizon and contact-rich nature of these tasks, and limited demonstrations, EgoSteer-9.6K achieves 75+% success while adapting robustly to spatial randomization. Conversely, the complete failure of DP, IMLE, and our from-scratch variant highlights the difficulty of these tasks, thereby validating that our 9.6 K-hour pre-training provides robust dexterous priors that can be few-shot adapted to novel embodiments and complex tasks.

7 Limitations & Conclusion

This paper presents a full-stack system for steerable dexterous manipulation by integrating EgoSmith, an efficient egocentric video curation pipeline; a unified robot stack for teleoperation, inference, and correction; and EgoSteer, a world-model-enhanced VLA trained on optimized infrastructure. EgoSteer demonstrates robust steerability across 40+ semantically diverse tasks, exhibiting dexterity, failure recovery, and generalization, while achieving few-shot adaptation to long-horizon tasks on multiple embodiments. These results substantiate the efficacy of our full-stack system. Despite these achievements, key limitations remain: first, robotic DoF limitations prevent transferring highly dexterous human knowledge, restricting intricate operations; second, the lack of tactile feedback across datasets, model, and embodiment limits contact-rich performance; and third, the pre-training scale can be expanded to capture broader priors and facilitate unseen task generalization. Addressing these challenges remains a key focus of our future research.

Acknowledgments

We sincerely thank Chengdong Ma, Wenxi Xu, and Shaoyang Guo for their generous help. We also extend our gratitude to our colleagues at PsiBot, including but not limited to Xiaojie Chai, Jianxin Du, Lin Huang, Ruochong Li, Haoyi Su, Tang Li, Yunlong Wang, Hongze Yu, Chaoyang Liu, and Hui Zhang, for their valuable support and helpful discussions.

References

[1] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, L. X. Shi, J. Tanner, Q. Vuong, A. Walling, H. Wang, and U. Zhilinsky. π_0: A vision-language-action flow model for general robot control, 2026. URL https://arxiv.org/abs/2410.24164.

[2] P. Intelligence, K. Black, N. Brown, J. Darpinian, K. Dhabalia, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, M. Y. Galliker, D. Ghosh, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, D. LeBlanc, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, A. Z. Ren, L. X. Shi, L. Smith, J. T. Springenberg, K. Stachowicz, J. Tanner, Q. Vuong, H. Walke, A. Walling, H. Wang, L. Yu, and U. Zhilinsky. $\pi_{0.5}$: a vision-language-action model with open-world generalization, 2025. URL https://arxiv.org/abs/2504.16054.

[3] P. Intelligence, A. Amin, R. Aniceto, A. Balakrishna, K. Black, K. Conley, G. Connors, J. Darpinian, K. Dhabalia, J. DiCarlo, et al. $\pi^*_{0.6}$: a vla that learns from experience. *arXiv preprint arXiv:2511.14759*, 2025.

[4] S. Liu, L. Wu, B. Li, H. Tan, H. Chen, Z. Wang, K. Xu, H. Su, and J. Zhu. Rdt-1b: a diffusion foundation model for bimanual manipulation. In *International Conference on Learning Representations*, volume 2025, pages 29982–30009, 2025.

[5] S. Liu, B. Li, K. Ma, L. Wu, H. Tan, X. Ouyang, H. Su, and J. Zhu. Rdt2: Exploring the scaling limit of umi data towards zero-shot cross-embodiment generalization. *arXiv preprint arXiv:2602.03310*, 2026.

[6] R. Zheng, D. Niu, Y. Xie, J. Wang, M. Xu, Y. Jiang, F. Castañeda, F. Hu, Y. L. Tan, L. Fu, T. Darrell, F. Huang, Y. Zhu, D. Xu, and L. Fan. Egoscale: Scaling dexterous manipulation with diverse egocentric human data, 2026. URL https://arxiv.org/abs/2602.16710.

[7] H. Luo, Y. Feng, W. Zhang, S. Zheng, Y. Wang, H. Yuan, J. Liu, C. Xu, Q. Jin, and Z. Lu. Being-h0: Vision-language-action pretraining from large-scale human videos, 2025. URL https://arxiv.org/abs/2507.15597.

[8] H. Luo, Y. Wang, W. Zhang, S. Zheng, Z. Xi, C. Xu, H. Xu, H. Yuan, C. Zhang, Y. Wang, Y. Feng, and Z. Lu. Being-h0.5: Scaling human-centric robot learning for cross-embodiment generalization, 2026. URL https://arxiv.org/abs/2601.12993.

[9] H. Luo, W. Zhang, Y. Feng, S. Zheng, H. Xu, C. Xu, Z. Xi, Y. Fu, and Z. Lu. Being-h0. 7: A latent world-action model from egocentric videos. *arXiv preprint arXiv:2605.00078*, 2026.

[10] J. Lyu, K. Liu, X. Zhang, H. Liao, Y. Feng, W. Zhu, T. Shen, J. Chen, J. Zhang, Y. Dong, et al. Lda-1b: Scaling latent dynamics action model via universal embodied data ingestion. *arXiv preprint arXiv:2602.12215*, 2026.

[11] L. Li, Q. Zhang, Y. Luo, S. Yang, R. Wang, F. Han, M. Yu, Z. Gao, N. Xue, X. Zhu, Y. Shen, and Y. Xu. Causal world modeling for robot control. *CoRR*, abs/2601.21998, January 2026. URL https://doi.org/10.48550/arXiv.2601.21998.

[12] W. Wu, F. Lu, Y. Wang, S. Yang, S. Liu, F. Wang, Q. Zhu, H. Sun, Y. Wang, S. Ma, et al. A pragmatic vla foundation model. *arXiv preprint arXiv:2601.18692*, 2026.

[13] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, T. Jackson, S. Jesmonth, N. J. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, I. Leal, K.-H. Lee, S. Levine, Y. Lu, U. Malla, D. Manjunath, I. Mordatch, O. Nachum, C. Parada, J. Peralta, E. Perez, K. Pertsch, J. Quiambao, K. Rao, M. Ryoo, G. Salazar, P. Sanketi, K. Sayed, J. Singh, S. Sontakke, A. Stone, C. Tan, H. Tran, V. Vanhoucke, S. Vega, Q. Vuong, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu, and B. Zitkovich. Rt-1: Robotics transformer for real-world control at scale, 2023. URL https://arxiv.org/abs/2212.06817.

[14] B. Zitkovich, T. Yu, S. Xu, P. Xu, T. Xiao, F. Xia, J. Wu, P. Wohlhart, S. Welker, A. Wahid, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pages 2165–2183. PMLR, 2023.

[15] P. Intelligence, B. Ai, A. Amin, R. Aniceto, A. Balakrishna, G. Balke, K. Black, G. Bokinsky, S. Cao, T. Charbonnier, et al. $\pi_{0.7}$: a steerable generalist robotic foundation model with emergent capabilities. *arXiv preprint arXiv:2604.15483*, 2026.

[16] S. Ye, Y. Ge, K. Zheng, S. Gao, S. Yu, G. Kurian, S. Indupuru, Y. L. Tan, C. Zhu, J. Xiang, A. Malik, K. Lee, W. Liang, N. Ranawaka, J. Gu, Y. Xu, G. Wang, F. Hu, A. Narayan, J. Bjorck, J. Wang, G. Kim, D. Niu, R. Zheng, Y. Xie, J. Wu, Q. Wang, R. Julian, D. Xu, Y. Du, Y. Chebotar, S. Reed, J. Kautz, Y. Zhu, L. J. Fan, and J. Jang. World action models are zero-shot policies, 2026. URL https://arxiv.org/abs/2602.15922.

[17] R. Hoque, P. Huang, D. J. Yoon, M. Sivapurapu, and J. Zhang. Egodex: Learning dexterous manipulation from large-scale egocentric video. *arXiv preprint arXiv:2505.11709*, 2025.

[18] R. Punamiya, S. Kareer, Z. Liu, J. Citron, R.-Z. Qiu, X. Cai, A. Gavryushin, J. Chen, D. Liconti, L. Y. Zhu, et al. Egoverse: An egocentric human dataset for robot learning from around the world. *arXiv preprint arXiv:2604.07607*, 2026.

[19] J. Zhang, J. Deng, C. Ma, and R. A. Potamias. Hawor: World-space hand motion reconstruction from egocentric videos. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 1805–1815, 2025.

[20] S. Ross, G. Gordon, and D. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.

[21] O. Siméoni, H. V. Vo, M. Seitzer, F. Baldassarre, M. Oquab, C. Jose, V. Khalidov, M. Szafraniec, S. Yi, M. Ramamonjisoa, et al. Dinov3. *arXiv preprint arXiv:2508.10104*, 2025.

[22] X. Cai, R.-Z. Qiu, G. Chen, L. Wei, I. Liu, T. Huang, X. Cheng, and X. Wang. In-n-on: Scaling egocentric manipulation with in-the-wild and on-task data. *arXiv preprint arXiv:2511.15704*, 2025.

[23] Y. Fu, N. Chen, J. Zhao, S. Shan, G. Yao, P. Wang, Z. Wang, and S. Zhang. Metis: Multi-source egocentric training for integrated dexterous vision-language-action model. *arXiv preprint arXiv:2511.17366*, 2025.

[24] K. Black, A. Z. Ren, M. Equi, and S. Levine. Training-time action conditioning for efficient real-time chunking. *arXiv preprint arXiv:2512.05964*, 2025.

[25] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

[26] K. Rana, R. Lee, D. Pershouse, and N. Suenderhauf. Imle policy: Fast and sample efficient visuomotor policy learning via implicit maximum likelihood estimation. *arXiv preprint arXiv:2502.12371*, 2025.

[27] Y. Zhong, F. Bai, S. Cai, X. Huang, Z. Chen, X. Zhang, Y. Wang, S. Guo, T. Guan, K. N. Lui, Z. Qi, Y. Liang, Y. Chen, and Y. Yang. A survey on vision-language-action models: An action tokenization perspective, 2025. URL https://arxiv.org/abs/2507.01925.

[28] O. M. Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, T. Kreiman, C. Xu, J. Luo, Y. L. Tan, L. Y. Chen, P. Sanketi, Q. Vuong, T. Xiao, D. Sadigh, C. Finn, and S. Levine. Octo: An open-source generalist robot policy, 2024. URL https://arxiv.org/abs/2405.12213.

[29] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, Q. Vuong, T. Kollar, B. Burchfiel, R. Tedrake, D. Sadigh, S. Levine, P. Liang, and C. Finn. Openvla: An open-source vision-language-action model, 2024. URL https://arxiv.org/abs/2406.09246.

[30] NVIDIA, :, J. Bjorck, F. Castañeda, N. Cherniadev, X. Da, R. Ding, L. J. Fan, Y. Fang, D. Fox, F. Hu, S. Huang, J. Jang, Z. Jiang, J. Kautz, K. Kundalia, L. Lao, Z. Li, Z. Lin, K. Lin, G. Liu, E. Llontop, L. Magne, A. Mandlekar, A. Narayan, S. Nasiriany, S. Reed, Y. L. Tan, G. Wang, Z. Wang, J. Wang, Q. Wang, J. Xiang, Y. Xie, Y. Xu, Z. Xu, S. Ye, Z. Yu, A. Zhang, H. Zhang, Y. Zhao, R. Zheng, and Y. Zhu. Gr00t n1: An open foundation model for generalist humanoid robots, 2025. URL https://arxiv.org/abs/2503.14734.

[31] K. Grauman, A. Westbury, E. Byrne, Z. Chavis, A. Furnari, R. Girdhar, J. Hamburger, H. Jiang, M. Liu, X. Liu, 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.

[32] B. AI. Egocentric-10k, 2025. URL https://huggingface.co/datasets/builddotai/Egocentric-10K.

[33] X. Wang, T. Kwon, M. Rad, B. Pan, I. Chakraborty, S. Andrist, D. Bohus, A. Feniello, B. Tekin, F. V. Frujeri, et al. Holoassist: an egocentric human interaction dataset for interactive ai assistants in the real world. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 20270–20281, 2023.

[34] X. Zhan, L. Yang, Y. Zhao, K. Mao, H. Xu, Z. Lin, K. Li, and C. Lu. Oakink2: A dataset of bimanual hands-object manipulation in complex task completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 445–456, 2024.

[35] Y. Liu, H. Yang, X. Si, L. Liu, Z. Li, Y. Zhang, Y. Liu, and L. Yi. Taco: Benchmarking generalizable bimanual tool-action-object understanding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21740–21751, 2024.

[36] P. Banerjee, S. Shkodrani, P. Moulon, S. Hampali, S. Han, F. Zhang, L. Zhang, J. Fountain, E. Miller, S. Basol, et al. Hot3d: Hand and object tracking in 3d from egocentric multi-view videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7061–7071, 2025.

[37] B. AI. Egocentric-100k, 2025. URL https://huggingface.co/datasets/builddotai/Egocentric-100K.

[38] Q. Li, Y. Deng, Y. Liang, L. Luo, L. Zhou, C. Yao, L. Zeng, Z. Feng, H. Liang, S. Xu, et al. Scalable vision-language-action model pretraining for robotic manipulation with real-life human activity videos. *arXiv preprint arXiv:2510.21571*, 2025.

[39] R. Yang, Q. Yu, Y. Wu, R. Yan, B. Li, A.-C. Cheng, X. Zou, Y. Fang, X. Cheng, R.-Z. Qiu, H. Yin, S. Liu, S. Han, Y. Lu, and X. Wang. Egovla: Learning vision-language-action models from egocentric human videos, 2025. URL https://arxiv.org/abs/2507.12440.

[40] J. Luo, C. Xu, J. Wu, and S. Levine. Precise and dexterous robotic manipulation via human-in-the-loop reinforcement learning. *Science Robotics*, 10(105):eads5033, 2025.

[41] X. Xu, Y. Hou, Z. Liu, and S. Song. Compliant residual dagger: Improving real-world contact-rich manipulation with human corrections. *Advances in Neural Information Processing Systems*, 38:139559–139581, 2026.

[42] Y. Han, Z. Chen, Y. Zhao, C. Xu, Y. Shao, Y. Peng, Y. Mu, and W. Lian. Dexhil: A human-in-the-loop framework for vision-language-action model post-training in dexterous manipulation, 2026. URL https://arxiv.org/abs/2603.09121.

[43] Z. Li, L. Huang, W. Xu, Z. Zhu, N. Lin, X. Ma, X. Sheng, and R. Wen. Hand-in-the-loop: Improving vla policies for dexterous manipulation via seamless hand-arm intervention, 2026. URL https://arxiv.org/abs/2605.15157.

[44] J. Redmon and A. Farhadi. Yolov3: An incremental improvement. *CoRR*, abs/1804.02767, 2018. URL http://arxiv.org/abs/1804.02767.

[45] R. A. Potamias, J. Zhang, J. Deng, and S. Zafeiriou. Wilor: End-to-end 3d hand localization and reconstruction in-the-wild. In *Proceedings of the Computer Vision and Pattern Recognition Conference*, pages 12242–12254, 2025.

[46] Z. Teed and J. Deng. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. *Advances in neural information processing systems*, 34:16558–16569, 2021.

[47] Z. Teed, L. Lipson, and J. Deng. Deep patch visual odometry. *Advances in Neural Information Processing Systems*, 36:39033–39051, 2023.

[48] J. Karhade, N. Keetha, Y. Zhang, T. Gupta, A. Sharma, S. Scherer, and D. Ramanan. Any4d: Unified feed-forward metric 4d reconstruction. *arXiv preprint arXiv:2512.10935*, 2025.

[49] Qwen Team. Qwen3.5: Towards native multimodal agents, February 2026. URL https://qwen.ai/blog?id=qwen3.5.

[50] T. Kwon, B. Tekin, J. Stühmer, F. Bogo, and M. Pollefeys. H2o: Two hands manipulating objects for first person interaction recognition. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10138–10148, 2021.

[51] G. Garcia-Hernando, S. Yuan, S. Baek, and T.-K. Kim. First-person hand action benchmark with rgb-d videos and 3d hand pose annotations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 409–419, 2018.

[52] D. Damen, H. Doughty, G. M. Farinella, S. Fidler, A. Furnari, E. Kazakos, D. Moltisanti, J. Munro, T. Perrett, W. Price, et al. Scaling egocentric vision: The epic-kitchens dataset. In *Proceedings of the European conference on computer vision (ECCV)*, pages 720–736, 2018.

[53] K. Zakka. Mink: Python inverse kinematics based on MuJoCo, Feb. 2026. URL https://github.com/kevinzakka/mink.

[54] S. Bai, Y. Cai, R. Chen, K. Chen, X. Chen, Z. Cheng, L. Deng, W. Ding, C. Gao, C. Ge, W. Ge, Z. Guo, Q. Huang, J. Huang, F. Huang, B. Hui, S. Jiang, Z. Li, M. Li, M. Li, K. Li, Z. Lin, J. Lin, X. Liu, J. Liu, C. Liu, Y. Liu, D. Liu, S. Liu, D. Lu, R. Luo, C. Lv, R. Men, L. Meng, X. Ren, X. Ren, S. Song, Y. Sun, J. Tang, J. Tu, J. Wan, P. Wang, P. Wang, Q. Wang, Y. Wang, T. Xie, Y. Xu, H. Xu, J. Xu, Z. Yang, M. Yang, J. Yang, A. Yang, B. Yu, F. Zhang, H. Zhang, X. Zhang, B. Zheng, H. Zhong, J. Zhou, F. Zhou, J. Zhou, Y. Zhu, and K. Zhu. Qwen3-vl technical report. *arXiv preprint arXiv:2511.21631*, 2025.

[55] W. Peebles and S. Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 4195–4205, 2023.

[56] Y. Lipman, R. T. Chen, H. Ben-Hamu, M. Nickel, and M. Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023.

[57] Y. Zhao, A. Gu, R. Varma, L. Luo, C.-C. Huang, M. Xu, L. Wright, H. Shojanazeri, M. Ott, S. Shleifer, et al. Pytorch fsdp: experiences on scaling fully sharded data parallel. *arXiv preprint arXiv:2304.11277*, 2023.

[58] J. Dong, B. Feng, D. Guessous, Y. Liang, and H. He. Flex attention: A programming model for generating optimized attention kernels. *arXiv preprint arXiv:2412.05496*, 2(3):4, 2024.

[59] J.-Y. Bouguet et al. Pyramidal implementation of the affine lucas kanade feature tracker description of the algorithm. *Intel corporation*, 5(1-10):4, 2001.

[60] R. Hartley and A. Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[61] J. Romero, D. Tzionas, and M. J. Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)*, 36(6), Nov. 2017.

[62] B. Wen, W. Yang, J. Kautz, and S. Birchfield. Foundationpose: Unified 6d pose estimation and tracking of novel objects. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 17868–17879, 2024.

[63] L. Wiedmann, O. Zohar, A. Mahla, X. Wang, R. Li, T. Frere, L. von Werra, A. R. Gosthipaty, and A. Marafioti. Finevision: Open data is all you need. *arXiv preprint arXiv:2510.17269*, 2025.

[64] E. Zhou, J. An, C. Chi, Y. Han, S. Rong, C. Zhang, P. Wang, Z. Wang, T. Huang, L. Sheng, et al. Roborefer: Towards spatial referring with reasoning in vision-language models for robotics. *Advances in Neural Information Processing Systems*, 38:28404–28481, 2026.

[65] H. Li, Z. Wang, Z.-h. Ding, S. Yang, Y. Chen, Y. Tian, X. Hu, T. Wang, D. Lin, F. Zhao, et al. Robointer: A holistic intermediate representation suite towards robotic manipulation. *arXiv preprint arXiv:2602.09973*, 2026.

[66] W. Yuan, J. Duan, V. Blukis, W. Pumacay, R. Krishna, A. Murali, A. Mousavian, and D. Fox. Robopoint: A vision-language model for spatial affordance prediction for robotics. *arXiv preprint arXiv:2406.10721*, 2024.

[67] Y. Tang, L. Zhang, S. Zhang, Y. Zhao, and X. Hao. Roboafford: A dataset and benchmark for enhancing object and spatial affordance learning in robot manipulation. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pages 12706–12713, 2025.

[68] K. Chen, S. Xie, Z. Ma, P. R. Sanketi, and K. Goldberg. Robo2vlm: Visual question answering from large-scale in-the-wild robot manipulation datasets. *arXiv preprint arXiv:2505.15517*, 2025.

[69] Y. Ji, H. Tan, J. Shi, X. Hao, Y. Zhang, H. Zhang, P. Wang, M. Zhao, Y. Mu, P. An, et al. Robobrain: A unified brain model for robotic manipulation from abstract to concrete. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1724–1734, 2025.

Appendix

Table of Contents

A Details of EgoSmith

This section presents the implementation details of EgoSmith (Section A.1) and provides detailed statistics of the curated 9.6K-hour pre-training corpus derived from 12 egocentric human video datasets (Section A.2).

A.1 Implementation Details

A.1.1 Pre-Filtering Heuristics

The pre-filtering stage employs frame-wise heuristics to rapidly discard low-quality segments, such as those containing locomotion, excessive head movement, hand absence, occlusion, or others' hand interference. These five scenarios are handled by two specialized gates: a camera gate for motion-related issues, and a hand gate for visibility anomalies. A contiguous segment is pruned only if it contains at least three consecutive invalid frames; isolated failures are retained as they exert negligible impact on subsequent reconstruction.

The camera gate estimates ego-motion using sparse optical flow. For each frame, a 128-point grid is tracked back to their positions 15 frames earlier via pyramidal Lucas–Kanade [59]. We fit a similarity transform to these correspondences using RANSAC [60]; the frame passes this gate if the translation of the estimated transform is within 10% of the image's longer dimension.

The hand gate uses YOLO [44, 45] to detect hands, retaining a bounding box as valid only if it satisfies three criteria: (a) confidence ≥ 0.30 to reject false positives; (b) area within $[2\%, 50\%]$ of the image, where the 50% upper bound excludes hands abnormally close to the lens, and the 2% lower bound is calibrated on Egocentric-10K/100K [32, 37] subsets, where we manually labeled small bounding boxes to find a threshold that filters out most other people's hands while retaining the operator's; and (c) spatially intersecting with the lower-central region (normalized $[0.075, 0.925]$ horizontally, $[0.075, 1.0]$ vertically). The gate requires ≥ 2 valid detections per frame, which naturally filters out hand absence, occlusion, or cases where only other people's hands are present, while preserving clear bimanual manipulation.

Together, the two gates yield video segments characterized by stable camera motion and clearly visible bimanual interactions.

A.1.2 4D Motion Estimation

In stage 2, we reconstruct hand motions in a unified, metric world-space coordinate system. Within this pipeline, we adopt the ViT module from HaWoR [19] as an off-the-shelf camera-frame hand reconstructor. Specifically, hands are detected and cropped across frames, and the ViT processes these inputs in temporal windows to regress the frame-wise MANO [61] pose parameters $\boldsymbol{\theta}_t \in \mathbb{R}^{51}$, shape parameters $\boldsymbol{\beta}_t \in \mathbb{R}^{10}$, and the camera-relative root translation $\mathbf{t}_t \in \mathbb{R}^3$. Although this camera-space hand reconstruction is highly reliable, placing these reconstructions in world space with accurate physical dimensions requires a robust, metric camera trajectory. This is a primary limitation of the original HaWoR pipeline, as its dependency on DROID-SLAM [46] can accumulate drift under egocentric conditions with rapid head movements or textureless environments. To address this, we design a robust pipeline to recover a metric, temporally consistent camera trajectory in world space, onto which the hand reconstructions are subsequently mapped.

We replace DROID-SLAM with DPVO [47] to estimate the camera trajectory. DPVO is more robust in long-range egocentric scenarios and incurs much lower computational cost, outputting up-to-scale camera poses $\hat{\mathbf{T}}_t = (\mathbf{R}_t, \hat{\mathbf{p}}_t) \in SE(3)$ along with focal length, where the hat notation $\hat{\cdot}$ denotes up-to-scale quantities. We then anchor the trajectory to the physical scale using metric depth estimates from Any4D [48] as a reference. To ensure temporal coherence across the entire sequence, we perform a cross-chunk alignment on the local Any4D depth windows, yielding a temporally consistent metric depth sequence.

To recover the metric scale factor s for DPVO trajectory, we compute the median ratio of the aligned Any4D depth $\mathbf{D}_t^{\text{Any4D}}$ to the DPVO depth $\hat{\mathbf{D}}_t^{\text{DPVO}}$ over the background pixels across all frames:

$$s = \text{median}_{t,(u,v)\in\mathcal{B}_t} \frac{\mathbf{D}_t^{\text{Any4D}}(u,v)}{\hat{\mathbf{D}}_t^{\text{DPVO}}(u,v)}, \tag{1}$$

where $\mathcal{B}_t$ is the valid background region in frame t, defined by excluding the hand regions projected from our reconstructed 3D hand mesh. We then calibrate the entire trajectory via $\mathbf{p}_t = s\hat{\mathbf{p}}_t$, yielding a metric camera trajectory.

Given the metric camera poses, we transform the camera-space hand vertices and joints, denoted generally as 3D coordinates $\mathbf{x}_t^{\text{cam}} \in \mathbb{R}^3$, into world space via $\mathbf{x}_t^{\text{world}} = \mathbf{R}_t^\top(\mathbf{x}_t^{\text{cam}} - \mathbf{p}_t)$. The output for each video segment consists of the world-space bimanual states and actions, frame-wise camera intrinsics and extrinsics $(\mathbf{K}_t, \mathbf{T}_t)$, MANO parameters, and the Any4D metric scene depth.

Concerning efficiency, stage 2 represents the primary computational bottleneck of our entire pipeline, with the majority of the overhead stemming from camera trajectory estimation. This highlights another advantage of EgoSmith: DPVO that we use is far more lightweight than the dense DROID-SLAM, substantially reducing this major cost. Building on this, we further optimize the throughput of this stage through parallelized batching and asynchronous I/O pipelining.

While the original HaWoR processes only a single 16-frame temporal window at a time, we group multiple windows into a single batch for parallel forward passes. Furthermore, we overlap CPU-based frame decoding and cropping with GPU-based model inference to prevent GPU idling. Benchmarks on an 8×A800 server using 8 video segments of 2K frames each show that our pipeline achieves an overall speedup of approximately 9× compared to HaWoR.

We further quantitatively benchmark the *accuracy* of our 4D motion estimation pipeline against HaWoR [19] on high-quality annotated subsets of TACO [35], H2O [50], OakInk-v2 [34], and EgoVerse [18]. To jointly assess camera-trajectory and hand-motion accuracy, we adopt four complementary metrics (all in mm):

Table 2: Benchmark results of 4D motion estimation.

Method	RPE ↓	ATE ↓	WA-MPJPE ↓	W-MPJPE ↓
HaWoR [19]	5.17	9.44	38.7	106.9
EgoSmith (Ours)	**2.42**	**7.60**	**25.9**	**86.0**

- **Relative Pose Error (RPE)**: Quantifies local tracking drift and frame-to-frame jitter over a fixed temporal interval. Because it is computed without any global alignment, this metric is highly sensitive to metric-scale inaccuracies.
- **Absolute Trajectory Error (ATE)**: Assesses the overall camera trajectory shape and long-term drift. The estimated trajectory is aligned to the ground truth via a global $\text{Sim}(3)$ transform before evaluation, making this metric insensitive to absolute scale errors.
- **World-Aligned Mean Per Joint Position Error (WA-MPJPE)**: Measures hand joint errors while accounting for global hand placement. The hand joints are aligned to the ground truth using a single $\text{Sim}(3)$ transform over each 100-frame segment, rather than performing per-frame local alignment.
- **World Mean Per Joint Position Error (W-MPJPE)**: Serves as the strictest metric for world-space hand tracking. It rigidly aligns only the first frame of each 100-frame segment via an SE(3) transform, thereby heavily penalizing absolute scale errors, temporal drift, and orientation misalignment across subsequent frames.

As shown in Table 2, EgoSmith substantially outperforms HaWoR across all four metrics. Our pipeline reduces RPE by over 50%, from 5.17 to 2.42 mm, and lowers ATE from 9.44 to 7.60 mm, indicating that the DPVO-based trajectory estimation yields superior local consistency and structural robustness. EgoSmith further improves WA-MPJPE from 38.7 to 25.9 mm and W-MPJPE from 106.9 to 86.0 mm, showing that our Any4D-based metric scaling, cross-window scale alignment, and global re-anchoring effectively mitigate scale distortion and long-term drift, ensuring physically plausible world-space hand tracking over extended sequences.

A.1.3 Language Labeling Prompt

Below, we present the prompt template designed for Qwen3.5-VL-Plus [49] to generate multi-granularity language annotations for egocentric human videos.

Egocentric Human Video Language Labeling Prompt

```
# Video Annotation Task: General Egocentric Hand Action Description

## Objective
Provide a comprehensive 5-level language description for pre-cropped egocentric videos of human hand-
     object interaction. These videos may show daily activities, tool use, object handling, cooking,
      cleaning, repair, assembly, organization, inspection, or other practical manipulation tasks.

## Context
- **Perspective:** Ego-view focused on the hands and manipulated objects.
- **Purpose:** Training data for robotic systems.
- **Core Focus:** The visible hand action, the manipulated object, contact points, grasp type,
     spatial relationships, and the physical sequence needed to complete the task.

---

## Task Instructions

Analyze the video clip and perform the following steps:

### Step 1: Content Filtering (Hand-Action Status Check)
- **Mark "status": "Invalid"** if the video shows:
    - Walking, camera transition, or scene scanning with no meaningful hand-object interaction.
    - Passive observation with no active manipulation.
    - Hands resting, hanging, or only briefly entering the frame without acting on an object.
    - Non-manipulation activities such as talking, reading, waiting, or looking around.
    - A task where the main hand action cannot be identified because of severe occlusion, blur, or
     ambiguity.
- **Mark "status": "Valid"** if the video shows active hand-object manipulation, such as picking,
     placing, opening, closing, pouring, wiping, folding, pressing, turning, cutting, fastening,
     arranging, inserting, removing, scanning a barcode/card/object, or operating an object.
- If the main action is identifiable but partially occluded, mark `"status":"Valid"`.

### Step 2: Multi-Level Hand Action Instructions (For "Valid" only)

- **Level 1 (Verb + Object):** Core task. Max 5 words. Example: "Open the drawer."
- **Level 2 (Gist):** Concise summary of the hand action. Max 15 words.
- **Level 3 (Object-Centric):** Describe the manipulated object, relevant parts, state, and spatial
     features. Max 30 words.
- **Level 4 (Hand-Centric):** Specify left/right hand roles, grasp style, contact points, and
     coordination. Max 50 words.
- **Level 5 (Dense Sequence):** Step-by-step physical breakdown. Max 100 words.
    - Use spatial anchors such as "near the left edge," "above the bowl," "inside the slot," or "
     against the surface."
    - Describe motion trajectories such as "lift upward," "slide forward," "rotate clockwise," "press
      downward," or "pull toward the body."
    - Describe outcome state such as "fully seated," "opened," "aligned," "placed flat," "wiped clean
     ," or "released."

---

## Strict Formatting & Quality Requirements

- **Verb-first imperative:** Start with an action verb. **NO subjects**.
- **Definite object references:** Use "the" for visible objects; avoid "a" or "an" in the generated
     instructions.
- **No transitional words:** Omit "then", "next", "afterwards".
- **Action Precision:** Use specific physical verbs such as "Grip," "Lift," "Place," "Open," "Close,"
      "Pour," "Fold," "Wipe," "Press," "Turn," "Insert," "Remove," "Align," "Slide," "Scoop," "Cut,"
      "Peel," "Tighten," or "Release" when possible.
- **Avoid over-specialization:** Do not infer hidden intent, brand names, object identities, or
     materials unless clearly visible.

---

## Output Format
Return a single JSON object. No markdown code fences. No extra text.
For `"status":"Invalid"`, return empty strings for all five levels.

{
  "status": "Valid/Invalid",
```

```
  "language_instructions": {
    "level1": "<verb and object>",
    "level2": "<concise summary>",
    "level3": "<object-focused description>",
    "level4": "<hand/object interaction details>",
    "level5": "<dense physical step-by-step>"
  }
}
```

A.1.4 Post-Filtering Criteria

In this stage, we perform quality control on the reconstructed outputs to filter out reconstruction anomalies and problematic segments, ensuring overall data quality. This evaluation is conducted from coarse to fine across three granularities: entire episodes, chunk windows, and adjacent frames.

Episode-level checks assess the overall camera motion. We compute the statistics of camera extrinsics, both translation and rotation, for each episode and compare them against the distribution of other episodes within the same dataset, discarding those that deviate significantly. Because reasonable camera motion magnitudes vary across datasets due to different devices, scenes, and manipulation styles, we employ a dataset-specific IQR criterion rather than a universal threshold. Specifically, an episode is classified as an outlier if its statistics fall outside the range $[Q_1 - 2.5\mathrm{IQR}, Q_3 + 2.5\mathrm{IQR}]$. This step filters out camera tracking drift, as well as segments dominated by walking or looking around instead of manipulation.

Chunk-level checks evaluate whether hands fall within physically reasonable spatial boundaries in a standardized egocentric coordinate frame. Directly comparing absolute hand coordinates is problematic, since they are coupled with camera and body movements. Specifically, within a sliding window spanning approximately the past 5 seconds and the future 30 frames, we transform all hand states and actions into the current camera frame. Under this canonical system, wrist positions are defined relative to the camera, and finger joints relative to the wrist. Within this coordinate system, we evaluate both distributional outliers and absolute physical limits. First, outliers in wrist translation/rotation and finger positions are identified using the same IQR criterion based on the respective dataset's distribution. Second, we enforce a universal physical ceiling of 1.5 meters on each coordinate axis for the hands, as a human hand cannot physically reach further than this distance from the head. If any sliding chunk window within an episode violates either the IQR outlier threshold or the 1.5-meter physical limit, the entire episode is discarded.

Frame-level checks identify sudden jumps between adjacent frames. We compute the frame-to-frame changes in camera translation and rotation, wrist translation and rotation, and finger translation. Unlike the previous levels, we do not rely on dataset-specific distributions here, as the physical speed of human hands and heads has a universal limit. Any movement exceeding this limit is attributed to reconstruction artifacts rather than valid motion. We therefore apply fixed physical thresholds: camera translation ≤ 0.20 m/frame, wrist and finger translation ≤ 0.30 m/frame, camera rotation $\leq 28^\circ$/frame, and wrist rotation $\leq 41^\circ$/frame. An episode is discarded if any of its frames violate these thresholds.

Collectively, these three levels of checks filter out problematic segments caused by head tracking drift, inaccurate motion reconstruction, and motion discontinuities.

A.2 Curated Dataset Statistics

The final pre-training corpus is constructed by utilizing EgoSmith to process 12 raw egocentric datasets, ultimately yielding a standardized, 9.6K-hour dataset. This curated corpus comprises world-space bimanual states and actions, frame-wise camera intrinsics and extrinsics, metric scene depth, and coarse-to-fine language annotations. In this section, we detail the scale, source composition, and semantic diversity of this dataset. The primary objective is not merely scaling up dataset volume; instead, the core advantage lies in data quality. Every sample is richly annotated and filtered through a rigorous quality-control pipeline, guaranteeing high-quality, modality-aligned manipula-

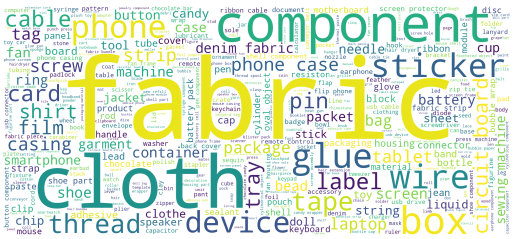
tion knowledge to assist the model in learning steerable dexterous manipulation. The dataset further offers broad coverage of manipulation tasks, objects, and multi-granularity language descriptions.

Scale and Source Composition. Figure 7a presents the duration contribution and proportion of each source dataset, while also indicating whether each annotation is natively provided or reconstructed by our pipeline. The source datasets are highly complementary, covering a wide range of devices, scenes, and manipulation styles. Although a small number of large-scale collections dominate the total duration, the breadth of sources contributes substantial diversity.

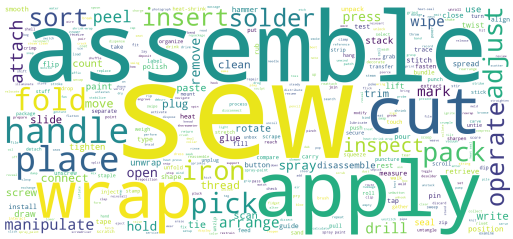
Task and Semantic Diversity. To characterize the manipulation span of our dataset, we extract (verb, object) tuples from the L1 verb–object annotations and analyze the distributions of action verbs and manipulated objects. The dataset covers 8969 distinct object nouns and 623 action verbs (Figures 7b and 7c). Figure 7d illustrates the top 50 most frequent "verb + object" atomic tasks. Common tasks concentrate on fundamental manipulation skills, aligning with the natural distribution of daily hand–object interactions. At the same time, the distribution exhibits a prominent long tail: a large number of low-frequency yet semantically diverse tasks and objects provides broad coverage, preventing the dataset from being biased toward a few dominant actions. This combination of common fundamental skills, long-tail task diversity, and multi-granularity language annotations provides downstream models with abundant training samples, broad task coverage, and rich dexterous manipulation knowledge.

Dataset	Hours	Percentage (%)	Episodes	Hand	Depth	Camera	Language
Egocentric-100K [37]	8,049	83.8	1,795,731	✓	✓	✓	✓
EgoVerse [18]	690	7.2	35,175		✓		✓
EgoDex [17]	370	3.9	147,588		✓		✓
Egocentric-10K [37]	288	3.0	194,915	✓	✓	✓	✓
Ego4D [31]	138	1.4	74,505	✓	✓	✓	✓
Epic-Kitchens [52]	49	0.5	26,454	✓	✓	✓	✓
HoloAssist [33]	11.5	0.1	11,426		✓		✓
HOT3D [36]	4.5	0.05	1,105		✓		
TACO [35]	3.0	0.03	1,558				✓
OakInk-v2 [34]	1.7	0.02	891		✓		
H2O [50]	1.0	0.01	935				
FPHA [51]	0.5	0.01	578	✓	✓		✓
Total	**9,606**	100	**2,290,861**				

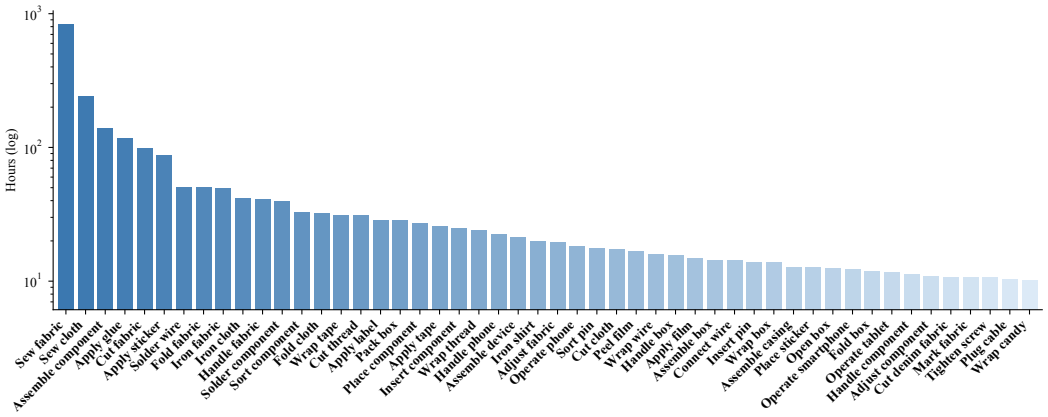
(a) Dataset composition and annotation sources



(b) Word cloud of manipulated objects



(c) Word cloud of action verbs



(d) Top-50 "verb + object" atomic tasks

Figure 7: Curated dataset statistics. (a) Source composition of the 12 egocentric datasets, detailing duration, percentage, episode count, and per-annotation origin. Checkmarks denote annotations generated by our pipeline, while blank entries indicate natively provided annotations. For FPHA, the checkmark under *Hand* specifically denotes the additionally annotated second hand. (b)–(d) Task and semantic diversity: word clouds of the manipulated objects and action verbs, and the most frequent verb–object atomic tasks, exhibiting long-tailed coverage over diverse manipulation skills.

B Details of the Robot Stack

This section details the implementation of the Robot Stack (Section B.1), alongside the collection protocols and statistics of the 187-hour real-robot dataset (Section B.2).

B.1 Implementation Details

B.1.1 Hardware Setup

The two physical embodiments utilized in our work are illustrated in Figure 3. The primary platform, referred to as the RealMan embodiment, consists of two 7-DoF RealMan RM75-6F robotic arms and two 6-DoF Ruiyan RY-H2 dexterous hands. It is equipped with one head-mounted and one chest-mounted Intel RealSense D455 camera, providing dual egocentric viewpoints to ensure a comprehensive visual field. The AgiBot G1 embodiment is adapted from an AgiBot G1 humanoid robot by replacing its default end-effectors with two 6-DoF Ruiyan RY-H2 hands. It features one head-mounted Intel RealSense D455 camera and two wrist cameras, the latter of which are unused. During experiments, the robot's neck, waist, and mobile base are kept fixed. Both platforms conduct tabletop manipulation on an operational table. The RealMan platform serves as the primary embodiment, on which the entire 187-hour dataset was collected. All experiments are conducted on this setup, with the sole exception of the few-shot adaptation experiment performed on the AgiBot G1.

For both platforms, the human wrist tracker, arm kinematic solvers, and joint control ROS 2 nodes operate at 100 Hz, while the glove, hand solvers, and control ROS 2 nodes run at 80 Hz. This high-frequency loop ensures near-zero latency, enabling intuitive bimanual teleoperation and the collection of fine-grained manipulation demonstrations. The cameras capture frames at 30 Hz. Data is recorded at these native frequencies and subsequently resampled to 30 Hz for training.

B.1.2 Aligning Robot Data with Egocentric Human Data

Robot data collection aims to ground the human manipulation priors learned from large-scale pre-training onto the target physical embodiment. To preserve and transfer this pre-trained knowledge, minimizing the domain gap between robot and human data is essential. Because the robotic palm is slightly longer than a human palm, we translate the robot's wrist coordinate frame axially forward. This alignment ensures that the scale from the wrist to the fingertips matches human anatomical proportions. When converting raw robot data into training data, this coordinate transformation is applied alongside hand-eye calibration to map hand actions into the camera coordinate frame. Correspondingly, during real-time policy inference, the control stack executes the inverse transformation and hand-eye mapping to project predicted actions back to the robot's physical coordinate space.

B.1.3 Hand-Eye Verification and Re-Calibration

In practice, hand-eye calibration is prone to drift due to mechanical maintenance, wear, or collisions, with such discrepancies sometimes only identified *post*-acquisition. To ensure data quality, we introduce an offline verification and automated recalibration pipeline. RGB-D images from the camera are projected into a 3D point cloud, and the robot's 3D mesh is rendered in the same space based on the hand-eye calibration, utilizing their spatial overlap for validation. Upon detecting misalignment, our pipeline runs FoundationPose [62] on the point cloud to estimate the robot hand's 6D pose based on its URDF. Through FK, the calibration matrix is back-calculated for each frame. Averaging these matrices across frames and removing outliers robustly recovers the calibration parameters, preventing data degradation.

B.1.4 Language Labeling Prompt

Below, we present the prompt template designed for Qwen3-VL-Flash [54] to generate multi-granularity language annotations for teleoperated demonstrations.

> **Robot Data Language Labeling Prompt**
>
> ```
> **CRITICAL VISUAL CONTEXT & PRIOR (READ CAREFULLY):**
> ```

```
You are observing two synchronized egocentric videos (Head View: top, Chest View: bottom) of an agent
      performing a manipulation task. This output will be used as language instructions for robotic
     training.
1. **The Agent's Hand:** The moving entity is the agent's bare end-effector. It generally has a grey
     base and black fingers.
2. **COLORED TIPS WARNING:** The tips/pads of the fingers often have GREEN, ORANGE, or RED tape/
     markers on them. THESE ARE PART OF THE FINGERS. They are NOT separate tools.
3. **EMPTY-HANDED PRIOR:** The agent is operating empty-handed. NEVER describe the agent as holding
     or using a 'green-tipped tool', 'hot knife', 'pliers', or any handheld instrument.
4. **ABSOLUTE GROUND TRUTH (TASK ALIGNMENT):** The specific task is **[{task_name}]**. This task name
      is your absolute ground truth for interpreting WHAT is being manipulated (Objects) and HOW it
     is being manipulated (Verbs). You MUST use the exact nouns implied by the task name.
5. **GRAMMAR:** Your description must be in **simple present tense**. Write in fluent English, avoid
     awkward phrasing.

**OBJECTIVE:**
Describe the agent's actions (focusing strictly on hand-object interactions) in **simple present
     tense** by integrating information from both views into a single, unified description at three
     levels of detail. **Since this is for robot training, you MUST completely ignore all task-
     irrelevant items.**

"**CONSTRAINTS:**
1. **Unified Description:** Provide ONE consolidated set of descriptions.
2. **Levels of Detail:**
   - **Level 1 (Gist):** A concise summary of the main action. Should always be a verb+noun phrase (i
     .e Ring a bell) (<20 words).
   - **Level 2 (Descriptive):** Main action + features and spatial layout of the **ACTIVELY
     MANIPULATED OBJECTS ONLY** (<40 words). **DO NOT list or describe any stationary background
     clutter.** In your description, list which hand is performing the action.
   - **Level 3 (Sequential):** The step-by-step temporal flow of key functional actions (<70 words).
     Include essential phases (reach -> manipulate -> release) but OMIT trivial micro-adjustments or
      hovering. List which hand is performing each action.
3. **Zero Subjects (Strict):** **Start every single sentence directly with a verb** (e.g., 'Reach for
     ...', 'Grasp...'). DO NOT use subjects (e.g., 'The person', 'The robot', 'The hand', 'It').
4. **Strict Focus on Interaction (NO CLUTTER):** Focus ONLY on the objects being actively touched,
     moved, or interacted with (and their immediate targets/receptacles). **Completely IGNORE all
     irrelevant background items** (e.g., wipes, boxes, tubes, stands that are not part of the task)
     . Never write phrases like 'other items remain unchanged' or 'in the background'.
5. **Vocabulary Restrictions:**
   - DO NOT output words like 'robot', 'mechanical arm', 'gripper', 'human', or 'finger'.
   - DO NOT output colors of the agent's hand/tips.
6. **Task Vocabulary (Verbs & Nouns):** Your choice of verbs AND target nouns must strictly align
     with the task **[{task_name}]**.
7. **Action Logic & Validation:** Focus on the actual state changes of the objects. Verify actual
     contact using both views.
8. **Object Disambiguation:** Use spatial descriptors (e.g., 'the topmost card') ONLY for task-
     relevant items to distinguish them from each other.
9. **Tense:** Use **Simple Present** tense (e.g., 'reach', 'grasp', 'slide').
10. **Spatial Description:** Use the camera frame as the reference.
   - Use **'upper', 'middle', 'lower'** to describe distance of objects on table. For example, 'the
     upper left of the table' refers to the far side of the table, and 'the lower left' refers to
     the near side.
   - Use 'left' 'right' to describe horizontal relationships\n"
   - Use 'on', 'on top of', 'above', 'below' etc. to describe vertical relationships.


**OUTPUT FORMAT:**
1. [Level 1 Description]
2. [Level 2 Description]
3. [Level 3 Description]


**EXAMPLE (If Task Name is 'Draw_cards'):**
1. Draw playing cards.
2. Slide the top cards from a central deck with left hand to draw them to the lower part of the table
     .
3. Reach toward the central deck with left hand, press down on the topmost card, and slide it
     backward. Return to the deck, press on the next card, and slide it backward to complete the
     draw."
```

B.2 Teleoperation Data Collection

Although pre-training on egocentric human videos equips the model with rich dexterous manipulation priors, direct physical deployment is prevented by the embodiment gap across visual appearance, dynamics, and kinematics. Furthermore, because these pre-training labels are generated by automated depth and hand pose estimation models, their accuracy is constrained by current model

capabilities, potentially introducing systematic biases into the pre-trained policy. Therefore, collecting real-robot teleoperation data aims to correct and ground these priors onto the target embodiment with high sample efficiency. Additionally, due to kinematic constraints on robotic degrees of freedom, the robot cannot perfectly replicate all fine-grained human hand movements. To achieve steerable manipulation, we must collect a highly diverse range of free-form tasks to maximize the coverage of basic manipulation primitives, guiding the model to efficiently transfer cross-domain knowledge and fostering robust compositional generalization and multi-task instruction following.

Based on these considerations, within the kinematic limits of RealMan's degrees of freedom, we design 193 semantically distinct dexterous manipulation tasks. For each task, approximately 300 randomized, diverse demonstration trajectories, totaling around 1 hour, are collected under cluttered scenarios, constructing a high-quality real-robot dataset of 187 hours and 55K trajectories.

These 193 tasks are classified into two major categories:

- **Common Tasks** (56 tasks): Everyday manipulations that are readily achievable within the current robot hardware configuration and sensor limits, yielding high teleoperation success rates.
- **Long-Tail Tasks** (137 tasks): Infrequent and physically challenging manipulations, such as contact-sensitive operations lacking tactile feedback, with lower collection success rates, primarily designed to ensure comprehensive semantic coverage of the dexterous manipulation space.

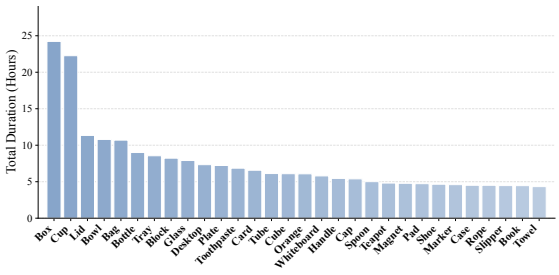
Furthermore, based on motion characteristics and physical interactions, all tasks are categorized into seven classes:

- **PnP-Easy**: Single-step tabletop pick-and-place where objects are easily graspable and the placement space is open.
- **PnP-Medium**: Non-planar or 3D spatial pick-and-place that requires precise operations related to containers, demanding higher control accuracy and spatial perception, such as "*put tennis ball into ball holder*".
- **PnP-Hard**: Multi-step or high-precision pick-and-place sequences, such as "*stack paper cups*".
- **Non-prehensile**: Actions not involving traditional finger grasping, including pushing, pulling, and pressing.
- **Reorient**: Operations involving rotation and reorientation, such as "*pour water*" or "*flip paper cups*".
- **Bimanual**: Bimanual tasks requiring high synchronization and spatial coordination between arms and hands, such as "*plug cable into charger*".
- **Contact-rich**: Operations involving frequent and complex physical contact interactions, requiring physical understanding, such as "*wipe whiteboard*".

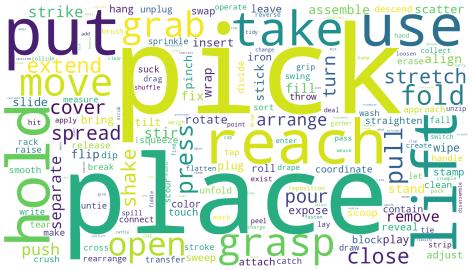
Throughout data collection, strict requirements are enforced on the randomness, diversity, and quality of each trajectory. Specifically, the tabletop features cluttered, unstructured scenes rather than pre-arranged, simplified environments, avoiding task identification from visual inputs alone. This design forces the model to deeply align language instructions with physical actions, encouraging it to learn general task semantics and execution goals rather than memorizing demonstration trajectories. Furthermore, given the highly randomized object configurations, operators are instructed to perform teleoperation in a natural, human-like manner. Consequently, different demonstrations of the same task exhibit substantial variations in execution trajectories, thereby covering a broader distribution of the mapping from human priors to the robot's action space.

As illustrated in Figure 8, we analyze the 187-hour real-robot dataset across several key dimensions, including the verb and noun word clouds with their top 30 frequency distributions, the task duration breakdown across seven manipulation categories, and the detailed duration statistics for the 56 common and 137 long-tail tasks. These statistics reveal a diverse vocabulary of actions and manipulated objects, alongside a balanced distribution across both manipulation categories and task durations. Encompassing rich semantic concepts and manipulation primitives, this dataset effectively grounds pre-trained human manipulation priors onto the RealMan embodiment. Additionally, Figure 9 showcases the dual-view image sequences and language annotations of a teleoperation trajectory, while Figure 10 visualizes sample tasks across each category.

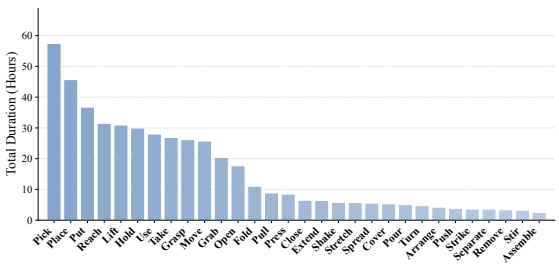
(a) Word cloud of nouns in annotation



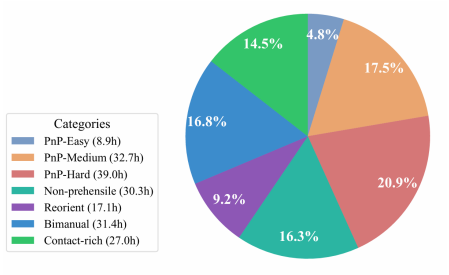
(b) Top 30 noun frequencies



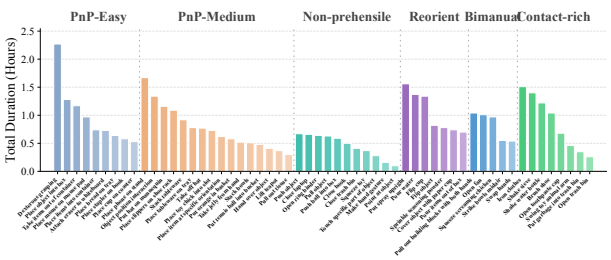
(c) Word cloud of verbs in annotation



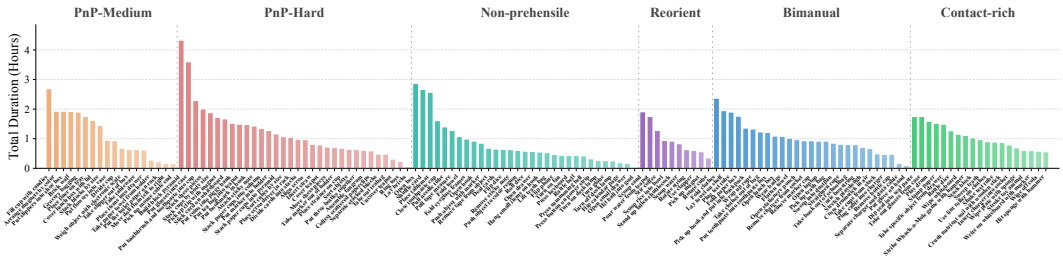
(d) Top 30 verb frequencies



(e) Task duration by seven categories



(f) Common task duration statistics



(g) Long-tail task duration statistics

Figure 8: Teleoperation dataset statistics. (1) Figure 8a - Figure 8d: Word clouds and top-30 frequency distributions of nouns and verbs in language annotations. (2) Figure 8e: Task duration breakdown by seven manipulation categories (PnP-Easy/Medium/Hard, Non-prehensile, Reorient, Bimanual, Contact-rich). (3) Figure 8f and Figure 8g: Detailed duration statistics for common tasks (56 tasks) and long-tail tasks (137 tasks).

L1	Put the white tea bag into the teapot.
L2	First use the right hand to move the black teapot to the center of the table, then use the left hand to pick up the white tea bag from the left and drop it into the teapot.
L3	Reach for the black teapot on the table with your right hand, grasp the handle of the teapot, lift it up and move it to the center of the table, then place it down. Next, reach for the white tea bag on the table with your left hand, grasp the label of the tea bag, lift it up and move it above the teapot, then put the tea bag into the teapot.

Figure 9: Dual-view image sequence and three-level language annotations of a representative trajectory. The frames illustrate a cluttered, randomized setup where the operator executes the task in a natural, human-like manner. The hierarchical language annotations describe the manipulation process from coarse to fine, assisting the model in aligning free-form instructions with physical actions.

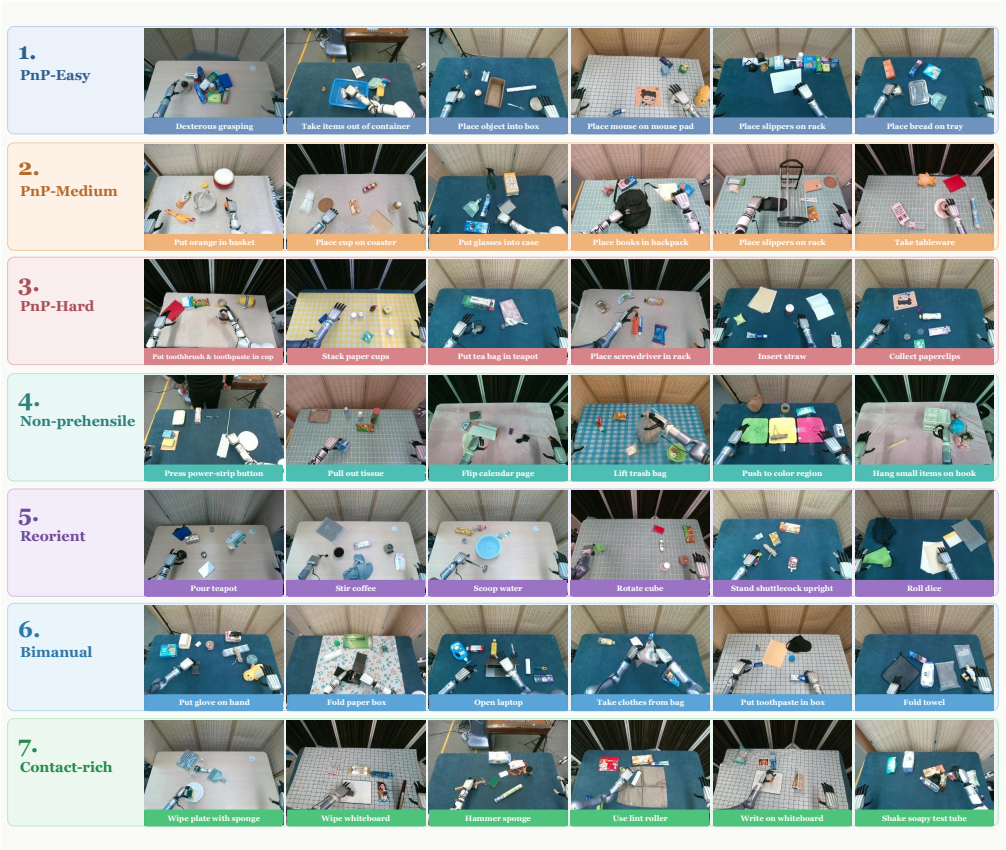


Figure 10: Representative task examples across seven manipulation categories. These diverse tasks are executed in cluttered, randomized scenarios using natural, human-like manipulations, covering a wide range of action semantics and manipulation primitives to facilitate the efficient grounding of pre-trained egocentric human priors onto the physical robot.

Dataset	Samples	Captioning	Visual Question Answering	Multiple-Choice Question	Pointing	Bounding Box	Affordance	Trajectory	Spatial	Planning
FineVision [63]	3.5M	✓	✓	✓		✓				
RefSpatial [64]	2.5M		✓	✓	✓				✓	
RoboInter-VQA [65]	1.6M		✓	✓	✓	✓		✓	✓	✓
RoboPoint [66]	1.3M		✓		✓	✓			✓	
RoboAfford [67]	765K		✓		✓	✓	✓			
Robo2VLM [68]	678K		✓	✓						
ShareRobot [69]	13K		✓			✓	✓	✓		✓

Table 3: VLM datasets used for EgoSteer co-training. This table details the sample sizes and the multi-modal and embodied knowledge domains covered by each dataset. This co-training mixture integrates general vision-language knowledge with interaction understanding and spatial-geometric priors, preserving the model's inherent world knowledge while facilitating its ability to follow free-form instructions to generalize across diverse manipulation tasks.

C Details of EgoSteer

This section first introduces the VLM datasets utilized for co-training with VLA data, *i.e.* egocentric human videos and real-robot data, to preserve EgoSteer's vision-language knowledge and ensure generalization (Section C.1). We then present implementation details of EgoSteer (Section C.2).

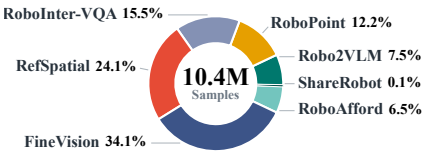


Figure 11: Statistical distribution of samples across VLM co-training datasets.

C.1 VLM Co-Training Data

To preserve general vision-language reasoning capabilities while simultaneously cultivating robust robotic task comprehension, a 10.4M-sample VLM co-training mixture is curated across seven datasets, ranging from open-world perception to embodied interaction grounding, to co-train EgoSteer. This mixture comprises four categories of data:

- **General VLM Pre-Training**: Incorporates FineVision [63] to prevent the catastrophic forgetting of open-world semantic concepts and preserve general visual-language reasoning.
- **Spatial Grounding**: Utilizes RefSpatial [64] and RoboPoint [66] for multi-step spatial referring, 2D visual grounding, and precise coordinate localization.
- **Embodied QA**: Integrates RoboInter-VQA [65], Robo2VLM [68], and ShareRobot [69] to support embodied question answering and temporal reasoning. This preserves the model's capabilities in high-level task planning and causal scene understanding.
- **Affordance Perception**: Adopts RoboAfford [67] for fine-grained manipulation affordance prediction and spatial interaction grounding.

The sample distribution of these datasets is illustrated in Figure 11, with their covered multimodal, embodied knowledge domains detailed in Table 3. We standardize these datasets to comply with the conversational input format of Qwen3-VL. Specifically, we normalize 2D bounding box and point coordinates to Qwen3-VL's native $[0, 1000]$ scale and adopt its standard representation formats. To ensure training stability, we only utilize single-image samples and exclude those with context lengths exceeding our maximum budget. By co-training on this VLM mixture, EgoSteer preserves open-world vision-language understanding and reasoning while enhancing its comprehension of robotics-specific tasks, thereby assisting the policy in following open-ended instructions and generalizing to novel manipulation scenarios.

C.2 Implementation Details

Backbone Input Scheme. EgoSteer treats the image observation history as a temporal video sequence to leverage the native video-processing capabilities of the Qwen3-VL-2B backbone. In practice, this history is downsampled to 6 frames at 1 FPS, covering a 5 s window, with proprioceptive states sampled at the corresponding timestamps. Language instructions and camera intrinsics are formatted as textual inputs, while the proprioceptive state history is encoded by a two-layer MLP and injected as continuous tokens. Because this proprioceptive history correlates strongly with target actions, the model is susceptible to shortcut learning, tending to ignore visual inputs and task

instructions in favor of proprioception. To mitigate this, each frame of the proprioceptive history is replaced by a learnable mask token with a 75% probability during training, forcing the model to attend to the full multimodal context. Additionally, when utilizing dual-camera inputs during real-robot post-training, the chest-camera sequence is randomly dropped with a 50% probability to prevent over-reliance on chest-view observations.

Action Expert. The action expert is built on a DiT architecture, comprising 14 layers with a hidden dimension of 1024 and an intermediate size of 2816. It features 8 attention heads, each with a head dimension $d_{\text{head}} = 128$, totaling approximately 300 M parameters.

The action expert operates on continuous action chunks of length $h = 32$ at a frequency of 30 Hz. Consistent with the VLM backbone, we apply Interleaved MRoPE for positional encoding. During pre-training, we set the delay $d = 0$ to disable training-time RTC, maximizing action supervision signals and learning rich human manipulation priors. During real-robot post-training, the simulated delay d is uniformly sampled as $d \sim \mathcal{U}([0, 5])$ to accommodate varying inference latencies during deployment. For the flow matching timestep $\eta \in [0, 1]$, the prefix action $\mathbf{a}_{\text{pre}}$ is assigned $\eta = 1$, representing no noise and excluding it from the loss computation. Conversely, the target suffix action $\mathbf{a}_{\text{suf}}$ has its timestep sampled from the probability distribution $P(\eta) = \text{Beta}(\frac{s-\eta}{s}; 1.5, 1)$ with $s = 0.999$ [1]. This timestep η is sinusoidally encoded, mapped through a two-layer MLP, and injected into the action expert via AdaLN-Zero.

Layer ℓ of the action expert employs a joint attention mechanism, attending to both itself and the VLM backbone's key-value cache from layer $f(\ell) = 2\ell$ after a linear projection. Specifically, let the query, key, and value of the m-th attention head in layer ℓ of the action expert be $\mathbf{Q}^{\text{AE}}_{\ell,m}, \mathbf{K}^{\text{AE}}_{\ell,m}, \mathbf{V}^{\text{AE}}_{\ell,m} \in \mathbb{R}^{h \times d_{\text{head}}}$, and the corresponding key and value of the m-th head in layer $f(\ell)$ of the backbone be $\mathbf{K}^{\text{B}}_{f(\ell),m}, \mathbf{V}^{\text{B}}_{f(\ell),m} \in \mathbb{R}^{N_{\text{B}} \times d_{\text{head}}}$, where N_{B} denotes the backbone's input sequence length. The joint attention at layer ℓ for the m-th head is mathematically formulated as:

$$\text{Softmax}\left(\frac{1}{\sqrt{d_{\text{head}}}}\mathbf{Q}^{\text{AE}}_{\ell,m}\left(\text{concat}[\mathbf{K}^{\text{B}}_{f(\ell),m}\mathbf{W}^{\text{K}}_{\ell}, \mathbf{K}^{\text{AE}}_{\ell,m}]\right)^{\text{T}}\right)\text{concat}[\mathbf{V}^{\text{B}}_{f(\ell),m}\mathbf{W}^{\text{V}}_{\ell}, \mathbf{V}^{\text{AE}}_{\ell,m}], \tag{2}$$

where $\mathbf{W}^{\text{K}}_{\ell}, \mathbf{W}^{\text{V}}_{\ell} \in \mathbb{R}^{d_{\text{head}} \times d_{\text{head}}}$ represent learnable projection matrices. This linear projection on the backbone's key and value representations is designed to align the semantic spaces of the backbone and the action expert.

During real-world deployment, the simulated delay is set to $d = 4$ to cover physical inference latency. To achieve efficient closed-loop control, only the first 12 steps of the predicted 32-step action chunk are retained. Subtracting the 4 prefix steps used for latency conditioning, the robot actually executes 8 new action steps per inference cycle. This high-frequency, asynchronous execution enables highly responsive control, rendering the system robust to dynamic manipulation tasks.

World Model Expert. The world model expert is a lightweight Transformer comprising 4 layers, with its single-layer architecture identical to Qwen3's text layer. It has a hidden dimension of 1024, an intermediate size of 4096, and 8 attention heads with a head dimension $d_{\text{head}} = 128$, totaling approximately 70 M parameters. At layer ℓ, the world-model expert employs a joint attention mechanism, attending to both itself and the VLM backbone's key-value cache from layer $f(\ell) = 7\ell$ after a linear projection, consistent with the formulation of the action expert.

For inputs, the relative camera motion $\Delta\mathbf{T} \in SE(3)$ is flattened into a 16-dimensional vector and encoded into a single continuous token via a two-layer MLP. Let the ground-truth DINOv3 features of the future frame $\mathbf{I}_{t+h-1}$ be $\mathbf{Z} \in \mathbb{R}^{H_v \times W_v \times C_{\text{DINO}}}$. We utilize DINOv3 (ViT-L/16) [21] for feature extraction with an input resolution of 384×384, yielding a spatial resolution of $H_v = W_v = 24$ and a feature dimension of $C_{\text{DINO}} = 1024$. To align with the spatial token-merge format of the backbone, the sequence length $L_{\mathbf{z}}$ of the input query vector $\mathbf{z}$ is configured to match the merged spatial resolution: $L_{\mathbf{z}} = H'_v \times W'_v = \frac{H_v}{2} \times \frac{W_v}{2} = 144$. The world model expert outputs $\hat{\mathbf{Y}} \in \mathbb{R}^{H'_v \times W'_v \times d_{\text{WM}}}$ with a channel dimension $d_{\text{WM}} = 1024$, which is subsequently mapped back to the original DINOv3 spatial resolution through a 2×2 linear upsampling projection layer, yielding the

reconstructed feature map $\hat{\mathbf{Z}} \in \mathbb{R}^{H_v \times W_v \times C_{\text{DINO}}}$. The world model objective is optimized via a mean squared error (MSE) loss:

$$\mathcal{L}_{\text{WM}} = \frac{1}{H_v \cdot W_v} \sum_{u=1}^{H_v} \sum_{v=1}^{W_v} \left\| \mathbf{Z}_{u,v} - \hat{\mathbf{Z}}_{u,v} \right\|_2^2. \tag{3}$$

Compared to direct image prediction in pixel space, regressing DINOv3 features preserves rich semantic information, naturally filtering out lighting variations and background noise to provide more stable gradient guidance for the VLM backbone. Furthermore, the world model expert adopts Interleaved MRoPE positional encoding, consistent with the VLM backbone, which enhances spatial-temporal awareness of multimodal sequences.

Attention Pattern. The Qwen3-VL backbone employs causal attention. Both the action expert and the world-model expert jointly attend to their entire respective sequences and the entire backbone sequence. Crucially, the action expert and the world-model expert do not attend to each other.

Data Processing. Due to significant variations in scale and quality across the 12 egocentric pre-training datasets, a heuristic sampling weight scheme is employed to balance different data sources. Specifically, each dataset i is assigned a subjective quality score $w_i \in [1, 10]$ based on data quality. To mitigate scale discrepancies, this score is scaled by the square root of the total frame count n_i, yielding the final sampling weight $W_i = w_i \cdot n_i^{0.5}$. For data augmentation, ColorJitter is applied to the input images. Furthermore, all action dimensions, except for wrist rotations, are normalized to the range $[-1, 1]$ using their 1st and 99th percentiles estimated from a randomly sampled subset.

Joint Optimization Objective. To balance the primary flow-matching task and the two auxiliary targets, the total training loss $\mathcal{L}_{\text{total}}$ is defined as:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{CFM}} + \mathcal{L}_{\text{WM}} + 0.05\mathcal{L}_{\text{VLM}}, \tag{4}$$

where $\mathcal{L}_{\text{CFM}}$ is the action flow-matching loss, $\mathcal{L}_{\text{WM}}$ is the world-model feature regression loss, and $\mathcal{L}_{\text{VLM}}$ is the autoregressive next-token prediction loss of the VLM. The loss weights are set to align the numerical scales of the three loss terms.

Training Infrastructure. Both the pre-training corpus curated by EgoSmith and the real-robot dataset are stored sequentially as individual episodes in the WebDataset format. During data loading, training batches are randomly sampled from a large shuffle buffer of size $16,384$. This buffer is continuously filled by drawing samples from each dataset proportionally to its pre-defined weight. For each dataset, the data stream is maintained by randomly selecting and streaming WebDataset shards, caching samples via a sliding window, and applying a 20% random retention probability. By combining randomized shard reading, sample dropping, and large-buffer shuffling, this scheme ensures training randomness while leveraging the sequential streaming of WebDataset, thereby drastically reducing I/O pressure. Furthermore, in multi-node distributed training, we explicitly manage Python's garbage collection to effectively mitigate training speed fluctuations and synchronization jitter across the cluster, ensuring high efficiency and stability.

D Experimental Details

This section provides supplementary details for the experimental evaluations presented in Section 6, detailing the model training hyperparameters and per-task success rates. In particular, Table 4 lists the hyperparameter configurations for both the pre-training and post-training phases in the main experiments of EgoSteer in Section 6.1. Table 5 presents the per-task success rates on the four evaluation tasks in Section 6.2 to demonstrate the performance gains from DAgger refinement. For the scaling analysis in Section 6.3, Table 6 lists the training hyperparameters of EgoSteer-3K/6K/9.6K pre-trained models, their respective post-training runs, and the baseline trained from scratch, with their corresponding task-specific success rates reported in Table 7. Additionally, Table 8 compares the detailed success rates of our method against the Being-H0.5 and $\pi_{0.5}$ baselines across ten tasks. Table 10 specifies the training configurations of the ablation experiment from Section 6.4, including the EgoSteer-1K model and its three ablated variants, *No WM-objective*, *No training-RTC*, and *Noisy data*. Their respective performance comparison is provided in Table 9. Finally, Table 11 specifies the hyperparameters for the few-shot fine-tuning of our pre-trained EgoSteer-9.6K on the two challenging, long-horizon tasks evaluated in Section 6.5, namely *Box-Folding* and *Cake-Unboxing*.

Hyperparameter	Pre-Training	Post-Training
Camera setup	Head	Head & Chest
Resolution	384×384	640×480
GPUs	128 A800	96 A800
Gradient accumulation	2	1
Global batch size	4608	384
Training steps	175K	60K
Learning rate (VLM / AE / WM)	1×10^{-4} / 3×10^{-4} / 3×10^{-4}	1×10^{-5} / 3×10^{-5} / 3×10^{-5}
Freeze-VLM steps	5000	0
Warmup steps	2000	2000
Training time	164 h	29 h

Table 4: Training configurations for both the pre-training and post-training phases in the main experiments of EgoSteer in Section 6.1. During pre-training, the VLM backbone is frozen for the first 5,000 steps, during which AE and WM are warmed up for 2,000 steps; once it is unfrozen, VLM is warmed up for 2,000 steps.

Task	EgoSteer-DG	EgoSteer-FT
Stack tableware	**80.0%**	50.0%
Close laptop	**70.0%**	10.0%
Place phone on stand	**50.0%**	0.0%
Flip cup	**50.0%**	30.0%
Average	**62.5%**	22.5%

Table 5: Success rates across four highly dexterous and failure-prone manipulation tasks, comparing EgoSteer-DG against EgoSteer-FT in Section 6.2. Each task is evaluated over 10 randomized trials. Bold values highlight the best performance.

	Pre-Training			Post-Training			
Hyperparameter	EgoSteer-3K	EgoSteer-6K	EgoSteer-9.6K	Scratch	EgoSteer-3K	EgoSteer-6K	EgoSteer-9.6K
Camera setup	Head	Head	Head	Head & Chest	Head & Chest	Head & Chest	Head & Chest
Resolution	384×384	384×384	384×384	640×480	640×480	640×480	640×480
GPUs	64 A800	64 A800	128 A800	64 A800	32 A800	64 A800	64 A800
Gradient accumulation	2	4	2	2	4	2	2
Global batch size	2304	4608	4608	512	512	512	512
Training steps	100K	100K	160K	60K	60K	60K	60K
Learning rate (VLM / AE / WM)	1×10^{-4} / 3×10^{-4} / 3×10^{-4}			1×10^{-5} / 3×10^{-5} / 3×10^{-5}			

Table 6: Training configurations for the pre-training scaling study in Section 6.3.

Task	Pre-Training Data (Hours)			
	Scratch	EgoSteer-3K	EgoSteer-6K	EgoSteer-9.6K
Grasp object	80%	80%	70%	**100%**
Hand over object	70%	80%	80%	**100%**
Place items into container	40%	80%	90%	**100%**
Point at object	20%	40%	60%	**70%**
Place toy chick into slot	30%	0%	20%	**40%**
Pull out tissue	60%	20%	50%	**80%**
Push ball into box	0%	0%	10%	**20%**
Put garbage into trash bin	0%	**60%**	30%	30%
Stack paper cups	0%	20%	10%	**40%**
Stack tableware	0%	**20%**	10%	**20%**
Average	30%	40%	43%	**60%**

Table 7: Per-task success rates for the pre-training scaling study in Section 6.3. Each task is evaluated over 10 randomized trials. Bold values highlight the best performance.

Task	Ours	Being-H0.5	$\pi_{0.5}$
Grasp object	**100%**	80%	80%
Hand over object	**100%**	60%	20%
Place items into container	**100%**	50%	0%
Pour items out of box	**50%**	30%	0%
Place bread on tray	70%	**80%**	40%
Pull out tissue	**80%**	10%	30%
Place item at specific orientation	**30%**	**30%**	**30%**
Attach eraser to whiteboard	**90%**	50%	20%
Put garbage into trash bin	**30%**	0%	0%
Put tennis ball into bucket	**90%**	0%	0%
Average	**74%**	39%	22%

Table 8: Per-task comparison between EgoSteer-9.6K and two VLA baselines, Being-H0.5 and $\pi_{0.5}$, in Section 6.3. All methods are post-trained on our real-robot dataset and evaluated on the same 10 tasks. Each task uses 10 randomized trials. Bold values highlight the best performance.

Task	Ours	No WM-objective	No training-RTC	Noisy data
Grasp object	**60%**	40%	30%	**60%**
Hand over object	**40%**	20%	30%	**40%**
Place items into container	50%	30%	50%	**70%**
Pour items out of box	**60%**	50%	10%	0%
Place bread on tray	**70%**	10%	60%	50%
Pull out tissue	20%	30%	**60%**	10%
Place item at specific orientation	**20%**	10%	10%	**20%**
Attach eraser to whiteboard	50%	**80%**	50%	40%
Put garbage into trash bin	**30%**	0%	0%	0%
Put tennis ball into bucket	40%	40%	**90%**	40%
Average	**44%**	31%	39%	33%

Table 9: Per-task success rates for the ablation study in Section 6.4. EgoSteer-1K is compared with ablated variants. Each task uses 10 randomized trials. Bold values highlight the best performance.

	Pre-Training				Post-Training			
Hyperparameter	Ours	No WM-objective	No training-RTC	Noisy data	Ours	No WM-objective	No training-RTC	Noisy data
Camera setup	Head	Head	Head	Head	Head	Head	Head	Head
Resolution	384×384	384×384	384×384	384×384	384×384	384×384	384×384	384×384
GPUs	64 A800	64 A800	64 A800	64 A800	64 A800	64 A800	64 A800	64 A800
Global batch size	1152	1152	1152	1152	1152	1152	1152	1152
Training steps	30K	80K	30K	20K	60K	60K	60K	60K
Learning rate (VLM / AE / WM)	1×10^{-4} / 3×10^{-4} / 3×10^{-4}				1×10^{-5} / 3×10^{-5} / 3×10^{-5}			

Table 10: Training hyperparameters for the ablation study in Section 6.4. All variants are pre-trained on 1K hours of egocentric data and post-trained on the same real-robot dataset, without DAgger refinement. Pre-training steps are selected at the lowest evaluation L_1 loss for each pre-training run.

Hyperparameter	Box-Folding	Cake-Unboxing
Pre-training checkpoint	EgoSteer-9.6K at 155K steps	
Camera setup	Head	Head
Resolution	384×384	384×384
GPUs	8 A800	8 A800
Gradient accumulation	1	1
Global batch size	144	144
Fine-tuning steps	44K	12K
Learning rate (VLM / AE / WM)	1×10^{-5} / 3×10^{-5} / 3×10^{-5}	
Demonstrations	120	229

Table 11: Few-shot fine-tuning hyperparameters of EgoSteer-9.6K for the two long-horizon dexterous tasks in Section 6.5.